



[12] 发明专利申请公开说明书

[21] 申请号 200310118647.6

[43] 公开日 2004 年 6 月 16 日

[11] 公开号 CN 1504902A

[22] 申请日 2003.11.27

[21] 申请号 200310118647.6

[30] 优先权

[32] 2002.12.5 [33] US [31] 10/313,288

[71] 申请人 国际商业机器公司

地址 美国纽约

[72] 发明人 拉维·K·阿瑞米利

本杰曼·L·古德曼

朱蒂·B·卓纳尔

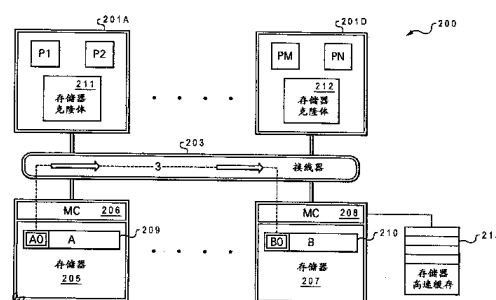
[74] 专利代理机构 中国国际贸易促进委员会专利
商标事务所
代理人 冯赓宣

权利要求书 3 页 说明书 34 页 附图 10 页

[54] 发明名称 在存储器部件之间保持数据一致性的方法和系统

[57] 摘要

一种数据处理系统包含执行指示和更新在存储器子系统内的所有数据线的一致状态的一致性协议，并且响应对包含修改的数据副本的存储器位置的裸写操作，将存储器位置的一致状态从修改(M)改变到无效(I)，而不启动推送数据到主存储器的相应的地址位置。在一致性协议中包含一组对存储器控制器接收的裸写请求的响应，其包括(1)当存储器控制器不能为写数据操作分配缓冲器，并且在该存储器位置的数据不是M状态时，发出重试响应；(2)当存储器控制器能够分配缓冲器时，发出Null响应；(3)当在存储器位置的数据从M变到I状态时，发出组合的Ack_Resend响应。当对特定的裸写操作在存储器克隆体接收到组合的响应时，将存储器位置的保护转移到存储器克隆体。



1、一种数据处理系统，包含：

具有处理器和存储器克隆体的处理器芯片；

具有各存储器部件的存储器子系统；

互连所述存储器子系统和所述处理器芯片的装置；

在所述存储器子系统内保持数据一致性并且还控制数据克隆操作所影响的所述存储器子系统内的所有数据线的一致状态的一致性协议。

2、按照权利要求 1 所述的数据处理系统，其中当对包含修改的数据副本的数据线监听裸写操作时，相应的监听器将数据线的一致状态从修改（M）改变到无效（I），而不启动推送所述数据到主存储器的相应的地址位置。

3、按照权利要求 1 所述的数据处理系统，其中所述一致性协议包括对在存储器控制器接收的裸写操作的一致性响应，所述一致性响应包括：

当在目的地存储器位置的数据线内的当前数据不是 M 状态时，并且存储器控制器不能为与数据写操作相关的数据分配缓冲器时，发出重试响应；以及

当存储器控制器能够分配缓冲器时，发出 Null 响应。

4、按照权利要求 1 所述的数据处理系统，其中所述一致性协议包括对在存储器控制器接收的裸写操作的一致性响应，所述一致性响应包括：

当在存储器位置的当前数据从 M 状态变到 I 状态时，发出组合的 Ack_Resend 响应，其中所述 Ack_Resend 响应指示发生了一致状态变化，但存储器控制器还不能够对写操作分配缓冲器。

5、按照权利要求 4 所述的数据处理系统，其中一致性协议将对其已发出了组合响应的线的保护转移到发出裸写操作的存储器克隆体，其中所述存储器克隆体保持所述保护直到裸写操作接收到 Null

响应。

6、按照权利要求 5 所述的数据处理系统，其中当所述存储器克隆体正保护所述线并且对所有监听的请求发出重试时，所述存储器克隆体监听所有对所述线的请求。

7、按照权利要求 5 所述的数据处理系统，其中当对裸写发出 Null 响应时，对所述线的保护被转移回所述存储器控制器。

8、一种在数据处理系统的存储器部件之间保持数据一致性的方法，所述方法包含：

监视在修改共享存储器地址的内容的所述数据处理系统上的所有的数据操作；

识别涉及裸写操作的数据操作；

对所有其它的为非裸写操作的操作执行标准一致性响应；以及

当监听器监听到裸写操作，存储器部件包含对应于该裸写操作的地址时，发送响应到发出所述裸写操作的存储器克隆体。

9、按照权利要求 8 所述的方法，还包含：

当在该架构上监听到数据克隆操作的读请求时，从最一致存储器位置发送所述克隆操作所需的数据。

10、按照权利要求 8 所述的方法，其中所述发送步骤还包含：

当存储器控制器不能为写数据操作分配缓冲器，并且在所述存储器位置的数据不是 M 状态时，发出重试响应；以及

当存储器控制器能够分配缓冲器时，发出 Null 响应。

11、按照权利要求 8 所述的方法，其中所述发送步骤还包含：

当在存储器位置的数据从 M 状态变到 I 状态时，发出组合的 Ack_Resend 响应，其中所述 Ack_Resend 响应指示发生了一致状态变化，但存储器控制器还不能够对写操作分配缓冲器。

12、按照权利要求 11 所述的方法，还包含：

当接收到所述 ACK_Resend 响应时，将所述存储器位置的保护从存储器控制器转移到发出裸写操作的所述存储器克隆体。

13、按照权利要求 12 所述的方法，还包含：

监听来自数据处理系统的其它部件的所有对存储器位置访问的请求；以及

当所述保护由所述存储器克隆体保持时，对每一监听的请求发出重试响应。

14、按照权利要求 12 所述的方法，其中所述裸写操作被重新发送，并且接收 Null 响应，所述数据地址的所述保护被转移回所述存储器克隆体。

在存储器部件之间保持 数据一致性的方法和系统

技术领域

本发明一般涉及数据处理系统，具体地说，涉及数据处理系统的处理器内的一致性操作。更具体地说，本发明涉及在与数据处理系统的架构上的“裸”写操作有关的数据克隆操作期间保持一致性方法和系统。

背景技术

对于更快速和硬件不太密集的数据处理和数据运算的需要一直是数据处理系统领域中发生的各种改进背后的驱动力。近来的趋势是更快速、更小并且更复杂处理器的出现，以及多处理器结构的实现，所述多处理器结构能够使多个互连的处理器同时执行指定任务的多个部分。除了多处理器结构的实现之外，还开发出了具有便于更有效地访问存储器的分布式存储器系统的系统。另外，实现了基于开关的互连（或者接线器（switch）），以代替传统的总线互连。

分布式存储器使得数据能够被保存在多个独立的存储器模块中，并且增强多处理器结构中的存储器访问。基于开关的互连使处理系统的各个组件彼此直接连接，从而在组件之间提供更快速/更直接的通信和数据传输。

图 1 是图解说明具有分布式存储器和基于开关的互连（接线器）的传统多处理器系统的方框图。如图所示，多处理器数据处理系统 100 包括通过接线器 103 彼此互连，并与其它系统组件互连的多个处理器芯片 101A-101D。其它系统组件包括分布式存储器 105、107（具有相关的存储器控制器 106、108），和输入/输出（I/O）组件 104。其它组件（未示出）也可通过接线器 103 与图解说明的组件互

连。处理器芯片 101A-101D 分别包括顺序标记为 P1-PN 的两个处理器核心（处理器）。除了处理器 P1-PN 之外，处理器芯片 101A-101D 包括和处理器 P1-PN 一起控制数据处理系统 100 内的处理操作的其它组件/逻辑电路。图 1 图解说明了一个这种组件，硬件引擎 111，其功能后面说明。

在如图 1 所示的多处理器数据处理系统中，多个处理器（或者处理器操作）通常可访问一个或多个存储器/存储模块，存储器一般被处理资源共享。由于每个处理资源可独立行动，因此系统内会出现对共享存储资源的竞争。例如，当第一处理器正在访问特定的存储器地址时，第二处理器可能试图对该存储器地址进行写（或读）操作。如果在先的访问正在进行中时，产生后一访问请求，那么必须延迟或阻止后面的请求，直到在先请求完成为止。从而，为了把数据写入特定的存储器位置（或者地址）或者从特定的存储器位置（或地址）读取数据，在读/写操作完全完成之前，处理器必须获得关于该特定存储器地址的锁定。这消除了当系统不知不觉地处理不正确（例如陈旧）数据时，可能发生的错误。

另外，借助更快速、更复杂的多处理器系统，多个数据请求可同时发出，并存在于不同的完成阶段中。除了关心一致性之外，处理器必须确保特定的数据块未被改变操作顺序。例如，如果处理器 P1 要求对位于地址 A 的数据块进行写操作，处理器 P2 必须读取同一数据块，并且如果在程序序列中读操作在写操作之前，那么重要的是保持这两个操作的顺序，以便获得正确的结果。

数据处理系统的标准操作要求处理组件（及其它组件）对数据的访问和移动或处理。数据通常保存在存储器中，并且利用执行程序代码的特定处理器发出的命令，被访问/读取、取回（retrieve）、处理、保存/写入和/或仅仅被移动。

数据移动操作不涉及对数据的数值/内容的改变/修改。相反，数据移动操作把数据从具有第一物理地址的一个存储器位置转移到具有不同物理地址的另一位置。在分布式存储器系统中，数据可从一个存

储模块转移到另一存储模块，不过单一存储器/存储模块内的移动也是可能的。

为了实现目前系统中的任意一种移动，完成下述步骤：（1）处理器引擎发出装入和保存指令，这导致高速缓冲存储器线（cache line）（“CL”）读取内容通过接线器/互连，从处理器芯片被传送给存储器控制器；（2）存储器控制器获得关于目的地存储器位置的锁定；（3）（存储器控制器）向处理器分配锁定目的地存储器位置；（4）通过接线器/互连，把数据从存储器（源地址）发送给处理器芯片（引擎）；（5）通过接线器/互连，把数据从处理器引擎发送给目的地位置的存储器控制器；（6）把数据写入目的地位置；和（7）为其它处理器释放目的地的锁定（lock）。该过程中固有的是即使当利用接线器时，把数据从源存储器位置转移到处理器芯片，随后从处理器芯片转移到目的地存储器位置的固定等待时间。

一般来说，每个装入和保存操作移动一个 8 字节的数据块。为了完成该移动，需要高速缓冲存储器的滚动，转换后备缓冲器（translation look-aside buffer）（TLB）的利用来实现有效读取地址转换，还需要利用处理器及其它硬件资源来接收和转发数据。至少一个处理器系统制造商已引入硬件加速装入线路和存储线路，和 TLB 一起在高速缓冲存储器线上实现字节级的同步操作。

现在利用图 1 图解说明处理器 P1 从存储器中的一个区域/位置（即物理地址）向另一区域/位置的数据移动。如图 1 中及识别路径 1 和 2 的方向箭所示，在数据移动操作中，通过把数据放在沿着数据路径 1 的总线（或者接线器 103）上，数据从存储器 105 中的地址位置 A 移动到处理器芯片 101A。随后通过接线器 103，沿着数据路径 2，数据从处理器芯片 101A 被发送给存储器 107 内的所需地址位置 B。

目前（及在前的）系统利用硬件引擎（即硬件模型）和/或软件编程模型（或者接口）完成上述数据移动操作。

在硬件引擎实现中，利用虚拟地址，硬件引擎 111 控制数据移动操作，并接收正被移动的数据。硬件引擎 111（也称为硬件加速

器)启动锁定获得过程,所述过程在开始数据的移动之前,获得关于源存储器地址和目的地存储器地址的锁定,以避免多个处理器同时访问位于该存储器地址的数据。不是把数据直接发送到处理器,数据被发送给硬件引擎 111。硬件引擎 111 利用高速缓冲存储器线读取,并以流水线方式实现要完成的写入。

就软件编程模型来说,软件把位置 A 和位置 B 通知处理器硬件,处理器硬件随后完成该移动。在该过程中,可利用真实地址(即不是虚拟地址)。因此,可消除前述硬件模型需要的进行虚拟地址-真实地址转换(或历史模式匹配)所需的其它时间。另外在软件模型中,地址可包括偏移量(例如地址 B 可偏移几个字节)。

处理器 P1 为实现数据移动操作而执行的典型伪代码序列如下:

LOCK DST	; 锁定目的地
LOCK SRC	; 锁定源
LD A (Byte 0)	; A _{B0} (4B 或 8B 数量)
ST B (Byte 0)	; B _{B0} (4B/8B)
INC	; 增加字节数
CMP	; 比较,看是否完成
BC	; 如果未完成,则转移
SYNC	; 实现同步
RL LOCK	; 释放锁定

字节数 (B0、B1、B2) 等被递增,直到由地址 A 识别的存储区内保存的所有数据被移动到由地址 B 识别的存储区为止。锁定和释放操作由存储器控制器和总线仲裁器 (arbiter) 执行,总线仲裁器向被授予锁定的请求处理器分配对特定地址的临时访问和控制。

在数据移动操作之后,在处理器 P1 能够重新开始处理其它后续操作之前,处理器 P1 必须接收指示所有数据已被物理移动到存储器位置 B 的完成响应(或信号)。这确保在处理单元之间存在一致性,并且数据一致性被保持。完成信号是对 SYNC 操作的响应,SYNC 操作是在数据移动操作之后,由处理器 P1 在架构 (fabric) 上发布,

以确保所有处理器接收数据移动操作的通知（及确认）。

从而，在图 1 中，处理器 P1 发出的指令启动从位置 A 到位置 B 的数据移动。处理器 P1 发出 SYNC，并且当最后的数据块已被移动到位置 B 时，向处理器 P1 发送指示物理移动已完成的信号。作为响应，处理器 P1 释放对地址 B 的锁定，处理器 P1 能够重新开始处理其它指令。

特别地，由于在能够开始移动操作之前，处理器 P1 必须获得关于位置 B 及位置 A 的锁定，因此完成信号还用信号通知锁定的释放，并使试图访问存储器位置 A 和 B 的其它处理器能够获得这两个存储器位置中任一位置的锁定。

虽然硬件模型和软件硬件分别提供不同的功能好处，但是都存在一些局限。例如，硬件模型和软件模型都具有把数据从存储器（源）装入处理器芯片，随后从处理器芯片装回到存储器（目的地）的固有等待时间。此外，就这两种模型来说，在整体移动完成之前，处理器必须等待，并且在处理器能够重新开始处理后续指令/操作之前，产生来自存储器控制器的完成响应。

于是，本发明认识到需要提供一种更有效的数据移动操作的方法和系统。能够利用对“裸”写操作的具体的存储器克隆响应的一致数据克隆操作的方法和系统理应是受欢迎的改进。本发明提供了这些及其它几个优点。

发明内容

公开了一种数据处理系统，该数据处理系统执行指示和更新在存储器子系统内的所有数据线的一致状态的一致性协议，并且响应对包含修改的数据副本的存储器位置的裸写操作，将存储器位置的一致状态从修改（M）改变到无效（I），而不启动推送数据到主存储器的相应的地址位置。

具体的一组对处理在存储器控制器接收的裸写请求的响应包含在一致性协议中。这些响应包括（1）当存储器控制器不能为写数据

操作分配缓冲器，并且在该存储器位置的数据不是修改（M）状态时，发出重试响应；（2）当存储器控制器能够分配缓冲器时，发出 Null 响应；（3）当在存储器位置的数据从修改状态变到无效状态时，发出组合的 Ack_Resend 响应。

该 Ack_Resend 响应指示发生了一致状态变化，但存储器控制器还不能够对写操作分配缓冲器。当对特定的裸写操作在存储器克隆体接收到组合的响应时，将该存储器位置的保护转移到该存储器克隆体。

附图说明

附加权利要求中陈述了本发明特有的新特征。但是，结合附图，参考例证实施例的下述详细说明，将更好地理解发明本身，以及本发明的优选使用方式、其它目的和优点，其中：

图 1 是根据现有技术图解说明具有用于移动数据的硬件引擎的多处理器数据处理系统的方框图；

图 2 是根据本发明的一个实施例，图解说明配置有存储器克隆体的处理器系统内的例证存储器-存储器克隆操作的方框图；

图 3 是根据本发明的一个实施例，图解说明图 2 的存储器克隆体的组件的方框图；

图 4A 是根据本发明的一个实施例，表示主存储器内的存储位置 X 和 Y 的方框图，所述存储位置 X 和 Y 被用于保存存储器克隆操作的源地址和目的地地址；

图 4B 根据本发明的一个实施例，图解说明图 2 的数据处理系统的（接线器）架构上的存储器地址操作数及标记，包括裸写的流动；

图 5A 是图解说明按照本发明的一个例证实施例工作的数据处理系统内克隆数据的一般过程的流程图；

图 5B 是根据本发明的一个实现，图解说明在数据克隆操作内发出裸写的过程的流程图；

图 5C 是根据本发明的一个实施例，图解说明结构完成状态前后

的过程步骤的流程图;

图 5D 是根据本发明的一个实施例, 图解说明通过发布读取操作, 发布物理移动数据的读取操作的过程的流程图;

图 6A 根据本发明的实现过程中, 一致提供 (sourcing) /保存数据的一个模型, 图解说明具有主存储器、几级高速缓冲存储器和外部系统存储器的分布式存储器子系统;

图 6B 图解说明具有上层金属的存储模块, 所述上层金属简化了相同存储模块内从源到目的地的数据直接克隆, 而不必利用外部接线器;

图 7A 是根据本发明的一个实施例, 用于把多个同时进行的数据克隆操作引向正确的目的地存储器的地址标记的框图;

图 7B 是根据本发明的一个实施例, 当裸写完成, 并且产生结构完成状态时, 存储器克隆体用于跟踪的寄存器的框图;

图 8A 是图解说明按照本发明的一个实施例工作的数据处理系统内的锁定争用过程的方框图;

图 8B 是根据本发明的一个实施例, 图解说明数据克隆操作期间, 保持数据一致性的过程的流程图;

图 9A 根据本发明的一个实施例, 图解说明具有附加模式位的指令, 所述模式位可由软件来回切换, 以指示指令的处理器执行是按照真实寻址模式, 还是按照虚拟寻址模式进行的;

图 9B 根据本发明的一个实施例, 图解说明数据处理系统内的应用代码、OS 和固件层, 以及各层支持的相关类型的地址操作。

具体实施方式

A. 综述

本发明提供与处理器 (或者处理器芯片) 相关的高速存储器克隆体 (cloner), 和利用高速存储器克隆体提供的特征, 完成数据克隆操作的有效方法。存储器克隆体使处理器能够遵循把数据从第一存储器位置移动到另一存储器位置的请求, 继续处理操作, 而不要求要

完成的数据的实际移动。

本发明引入关于移动操作的层次结构完成状态。存储器克隆体提供的功能特征包括裸 (naked) 写操作, 支持裸写操作和直接存储器-存储器数据移动的先进相关一致性操作, 指令集体系结构内的新指令 (例如, 无中断情况下, 借助指令的流水线式发布, 优化的组合指令集), 以及在数据处理的虚拟寻址模式和真实寻址模式之间动态转换的模式位。本发明还提供了数据处理系统的其它新颖操作特征。

本发明利用了目前的处理系统中存在的开关布局以及存储器控制器的功能性。和目前的用于实现移动操作的基于硬件或基于软件的模型 (它要求把数据回送给请求处理器模块, 随后从处理器模块把数据转发给目的) 不同, 本发明实现具有允许直接把数据传送给目的地的附加特征的组合软件模型和硬件模型。最好利用设计有存储器克隆体的处理器芯片实现本发明, 存储器克隆体包括下面说明的各种硬件和软件逻辑单元/组件。

本发明的说明规定了几个新术语, 其中关键的是高速存储器克隆体进行的“克隆”操作。如此处使用的, 克隆操作指的是在高速存储器克隆体内, 在架构上, 以及在存储器位置发生的, 一同实现数据的层次结构完成状态和实际物理移动的所有操作。数据从 A 点被移动到 B 点, 但它是按照和完成数据移动操作的已知方法完全不同的方式进行的。数据“移动”的引用 (reference) 具体指的是从处理器发给高速存储器控制器的指令。在一些情况下, 当具体指的是作为数据克隆操作一部分的数据的物理移动时, 利用术语“移动”。从而, 例如, 物理数据移动的完成被看作是数据克隆操作的一部分。

B. 硬件可选件

现在参见附图, 特别参见图 2, 图 2 图解说明了多处理器、接线器连接的数据处理系统 200, 本发明可在数据处理系统 200 内实现。数据处理系统 200 包括多个处理器模块/芯片, 图中描绘了它们的两个, 即芯片 201A 和 201D。处理器芯片 201A 和 201D 均包含一个或多个处理器 (P1、P2 等)。存储器克隆体 211 至少位于处理芯片之

一内（例如，处理器芯片 201），后面参考图 3 说明存储器克隆体 211。处理器芯片 201A 和 201D 通过接线器 203 彼此互连，并与数据处理系统 200 的其它组件互连。这些其它组件包括分布式存储器模块，图中描绘了它们中的两个，即存储器 205 和 207，存储器 205 和 207 分别具有相应的存储器控制器 206 和 208。与存储器控制器 208 相联系的是存储器高速缓存 213，下面结合裸写操作的说明，描述存储器高速缓存的功能。

在数据克隆操作的实现过程中，通过接线器 203，数据直接从存储器 205 的存储位置 A 被移动到存储器 207 的存储位置 B。从而，数据沿着不包括处理器或处理器模块的直接路径 3 移动。即，被移动的数据不是首先被发送给存储器克隆体 211 或处理器 P1。当正在完成物理移动的时候，数据的实际移动由存储器控制器 206 和 208 分别控制（或者在基于下面描述的模型相关情况下，由高速缓存控制器控制），存储器控制器 206 和 208 还分别控制对存储器 205 和 207 的访问。

这里只是出于举例说明的目的，给出了数据处理系统内的处理器和存储器的例证结构。本领域的技术人员明白本发明的各种功能特征完全适合于包含非分布式存储器和/或单个处理器/处理器芯片的系统结构。于是，这里描述的本发明的功能特征适合于不同结构的数据处理系统，只要数据处理系统包括高速存储器克隆体和/或借助其，可完成这里描述的各种功能特征的类似组件。

高速存储器克隆体

存储器克隆体 211 包括借助其控制和/或启动本发明的处理的硬件和软件组件。具体地说，如图 3 中所示，存储器克隆体 211 包括控制逻辑单元 303 和转换后备缓冲器（TLB）319。存储器克隆体 211 还包括几个寄存器，包括 SRC 地址寄存器、DST 地址寄存器 307、CNT 寄存器 309、层次结构完成寄存器 313 和克隆完成寄存器 317。模式位 315 也包含在存储器克隆体内。在本文献的相关章节描述存储器克隆体的各个图解组件的功能。

特别地，和硬件加速器或类似组件不同，存储器克隆体只接收和发布地址操作。借助每个芯片单一存储器克隆体，可实现本实现。另一方面，每个微处理器可使用相应的存储器克隆体。

TLB 319 包括虚拟地址缓冲器 321 和真实地址缓冲器 323。与处理器 P1、P2 等使用的 I-TLB 和 D-TLB 分离的 TLB 319 被固定，并且与 I-TLB 和 D-TLB 协同工作。缓冲器 321 和 323 由 OS 在启动时加载，并且最好保存由 OS 和进程引用的所有地址的转变，从而不必读取存储器中的 OS 页表。

SRC、DST 和 CNT 寄存器

在图 3 的图解说明的实施例中，存储器克隆体 211 包括源 (SRC) 地址寄存器 305、目的地 (DST) 地址寄存器 307 和计数 (CNT) 寄存器 309。如它们的名字暗示那样，目的地寄存器 307 和源地址寄存器 305 分别保存相对于其移动数据的存储位置的目的地地址和源地址。计数寄存器 309 保存在数据克隆操作中正在被转移的高速缓冲存储器线的数目。

从存储器中用于保存数据克隆操作的目的地地址和源地址的位置 (X 和 Y) 读取目的地地址和源地址。源地址和目的地地址的读取由发出一个或多个指令的处理器 (例如 P1) 触发，所述一个或多个指令一起使存储器克隆体启动数据克隆操作，如下详细所述。

C. 数据克隆操作的一般过程

图 5A 图解说明由利用上述硬件组件的本发明完成的整个过程的几个主要步骤。所述过程始于方框 501，之后，处理器 P1 执行指令，所述指令构成把数据从存储位置 A 克隆到存储位置 B 的请求，如方框 503 所示。存储器克隆体接收数据克隆请求，取回虚拟源地址和目的地地址，查寻对应的真实地址，并启动裸 WR 操作，如方框 505 所示。在架构 (fabric) 上执行裸 WR 操作，存储器克隆体监视层次结构完成状态，如方框 507 所示。在已结构完成克隆操作的指示之后，如方框 509 所示，存储器克隆体用信号通知处理器克隆操作完成，处理器继续处理，好象数据移动已物理完成一样。随后，存储器

克隆体在后台完成实际的数据移动，如方框 511 所示，并且在物理移动数据的时候，存储器克隆体进行必要的高速缓冲存储器线的保护。该过程随后在方框 513 终止。下面参考这里提供的几个其它流程图，展开并说明由图 5A 的各个方框规定的过程。

现在参考图 5B，图中图解说明了图 5A 的完成方框 505 中涉及的几个步骤。该过程始于方框 521，随后转移到方框 523，方框 523 图解说明从存储位置 X 和 Y 取回被请求数据克隆操作的目的地地址和源地址，并将其置于存储器克隆体中的相应寄存器中。计数值（即数据的高速缓冲存储器线的数目）也被置于 CNT 寄存器中，如方框 525 所示。随后如方框 526 所示，完成源令牌（token）和目的令牌操作。之后，裸 CL WR 被放在架构上，如方框 527 所示。每个裸 CL WR 在架构上接收来自存储器控制器的响应。在方框 529，确定响应是否为 NULL。如果响应不是 NULL，那么重试裸 CL WR 操作，如方框 531 所示。但是，当响应为 NULL 时，在存储器克隆体 211 内，裸 CL WR 被标记为完成，如方框 533 所示。在以下各个章节中更详细地说明在图 5B 中图解说明的各个步骤。

移动操作数（operand）和移动地址的取回

为了能够清楚地理解本发明，下面参考小块的程序代码，以及从存储位置 A（具有虚拟地址 A 和真实地址 A1）到另一存储位置 B（具有虚拟地址 B 和真实地址 B1）的数据克隆，说明数据克隆操作的实现。从而，例如，在处理器 P1 执行的导致从存储位置 A 到存储位置 B 的数据克隆的样本程序代码块如下：

```
ST    X           (地址 X 持有虚拟源地址 A)
ST    Y           (地址 Y 持有虚拟目的地地址 B)
ST    CNT         (CNT 是要克隆的数据线的数目)
SYNC
ADD
```

上面表示存储器克隆体从处理器接收的启动克隆操作的样本指令。ADD 指令被用作在完成数据克隆操作之前，不被处理器执行的

样本指令。每当从处理器收到上述直到 SYNC 的一系列指令时，存储器克隆体启动数据克隆操作。在存储器克隆体执行上述一系列指令会导致把虚拟源地址和目的地地址返回给存储器克隆体，还会提供要移动的数据线的数目。在例证实施例中，CNT 的值等于一页存储器内的线（line）的数目，克隆操作被描述成克隆位于地址 A1 的单页数据。

图 4A 用 X 和 Y 存储位置的方框表示，图解说明了存储器 405，存储器 405 可以是存储器子系统内的任意存储器 205、207，数据克隆操作的源地址和目的地地址，A 和 B 存在于所述 X 和 Y 存储位置之中。在一个实施例中，在较早的执行时间，处理器把克隆操作的 A 和 B 地址保存在 X 和 Y 存储位置中。每个存储位置包括 32 位的地址数据和之后的 12 个保留位。根据图解说明的实施例，附加的这 12 位中的头 5 位被数据处理系统的状态机用于选择源或目的地页地址范围内 32 个可能页中的哪一页正被请求/访问。

如图 4A 中所示，X 和 Y 地址是保存 A 和 B 虚拟地址的存储位置，并且当包含在存储请求（ST）中时，向处理器及存储器克隆体指出该请求是关于数据克隆操作（而不是关于常规保存操作）的请求。虚拟地址 A 和 B 对应于数据克隆操作的源和目的地的真实存储器地址 A1 和 B1，并被保存在存储器克隆体 211 的 SRC 地址寄存器 305 和 DST 地址寄存器 307 内。在下面的存储器克隆操作的说明中，A 和 B 指的是地址，所述地址是保存在存储器克隆体内的数据地址，而 A1 和 B1 指的是发布给架构的（即在接线器上发出的）真实存储器地址。A 和 A1 及 B 和 B1 分别代表数据克隆操作的源存储位置和目的地存储位置。

在图解说明的实施例中，当存储器克隆体 211 接收处理器和其后跟随 SYNC 的 ST 命令序列时，TLB 319 分别根据虚拟地址（X 和 Y）查寻真实地址 X1 和 Y1。X1 和 Y1 是专用于保存存储器克隆操作的源地址和目的地地址的存储位置。存储器克隆体 211 通过接线器（即在架构上）向存储器发出操作，所述操作访问相应的位置，并把

目的地地址和源地址返回给存储器克隆体 211。存储器克隆体 211 分别从位置 X1 和 Y1 接收源 (A) 和目的地 (B) 的虚拟地址。提供的实际地址是第一页存储器地址。

存储器克隆体 211 把源地址和目的地地址, 以及从处理器 P1 接收的高速缓冲存储器线计数分别保存在寄存器 305、307、309 中。根据保存在 CNT 寄存器 309 内的值, 存储器克隆体能够产生始于利用 12 个保留位的头 5 个附加位的 SRC 寄存器 305 和 DST 寄存器 307 内的地址, 从 0-31 顺序编号的连续地址。

例如, 对其中把具有 128 字节线的一页 4 K 字节数据从存储器地址 A1 (具有 4K 对准地址) 移动到存储器地址 B1 (也具有 4K 对准地址) 的克隆操作来说, 对应于状态机地址扩展 00000~11111, 为 32 的计数值被保存在 CNT 寄存器 309 中, 所述状态机地址扩展 0000-11111 被附到所述头 5 个位中的源地址上。这些地址扩展可由状态机设置 (即存储器克隆体利用的计数器), 并且识别页内哪些地址块正在被移动。

另外, 除了所有页之外, 本发明的附加特征还能够实现部分存储页的克隆。该特征与例如其中在具有不同大小的高速缓冲存储器线的存储器组件之间进行移动操作的实施例相关。

响应虚拟源地址和虚拟目的地地址的接收, 存储器克隆体 211 实现 (1) 把源地址 (即地址 A) 保存在 SRC 寄存器 305 中, 和 (2) 把目的地地址 (即地址 B) 保存在 DST 寄存器 307 中的功能。存储器克隆体 211 还把从处理器接收的 CNT 值保存在 CNT 寄存器 309 中。保存的源地址和目的地地址是处理器在预先处理中产生的虚拟地址。TLB 319 随后可查寻这些地址, 以便确定存储器中的相应真实地址, 随后使用所述真实地址进行下面说明的数据克隆操作。

D. 令牌操作

现在返回方框 526, 在开始存储器克隆的写入和读取操作之前, 存储器克隆体发出一组令牌 (或者地址占有权 (address tenure)), 在图解说明的实施例中, 所述一组令牌指的是源

(SRC) 令牌和目的地 (DST) 令牌。SRC 令牌是在架构上的操作，该操作查询系统，了解任何其它存储器克隆体目前是否正在利用 SRC 页地址。类似地，DST 令牌也是架构上的操作，该操作查询系统，了解任何其它存储器克隆体目前是否正在利用 DST 页地址。

在发出启动克隆操作的操作之前，由存储器克隆体在架构上发出 SRC 和 DST 令牌。系统中的所有其它存储器克隆体（或者处理器）监听每个存储器克隆体的令牌。每个监听器（snooper）对照当前正在该监听器使用的任意地址，检查令牌的源地址和目的地地址，随后每个监听器发出向发布该令牌的存储器克隆体指出地址是否正被监听器之一使用的应答。令牌操作确保不会有两个存储器克隆体试图对相同位置进行读取/写入操作。令牌操作还确保存储器地址空间对于数据克隆操作可用。

令牌的应用防止了多个存储器克隆体同时把数据写入相同的存储位置。除了防止不同操作对存储位置的多个同时更新之外，令牌操作还有助于避免活锁（livelock），以及确保保持存储器内的一致性（coherency）。本发明还提供确保处理器不会活锁的其它方法，如下所述。

利用令牌地址操作数使存储器克隆体能够在开始一系列写操作之前，接收关于源地址和目的地地址的明确（clear）信号。一旦存储器克隆体收到来自令牌的明确信号，存储器克隆体能够通过发出裸高速缓冲存储器线（CL）写（WR）操作，随后发出 CL 读（RD）操作，开始克隆体操作。

随后根据接收的源地址和目的地地址产生令牌操作，并发出令牌操作，以确保关于访问相应存储器位置的明确响应。在架构上发出 SRC 和 DST 令牌操作，以确定被请求的存储位置是否适合于克隆体（即，当前未被另一处理器或克隆体使用等），并保留可用的地址，直到克隆操作完成为止。一旦 DST 令牌和 SRC 令牌操作返回明确响应，那么存储器克隆体开始通过监听关于访问对应地址空间的其它请求，保护这些地址空间，如下所述。

特别地，在一个实施例中，一旦来自 DST 令牌的响应指示对于该克隆操作来说，目的地地址是明确的（甚至在未收到来自 SRC 令牌的明确响应的情况下），那么允许开始克隆操作。本实施例使数据能够同时来源于相同的源地址，从而允许关于相同源地址的多个同时进行的克隆操作。这种实现的一个主要原因在于和传统的移动操作不同，存储器克隆体控制的克隆操作开始于对目的地地址的一系列裸写操作，如下详细所述。

现在提供可以以令牌的应用为基础的可能的数据提供操作的例子。在本例中，“A”用于代表数据源自的源地址。“B”代表正相对于其完成存储器克隆的目的地的地址，“O”代表试图访问分别对应于地址 A 或 B 的访问位置 A 或 B 的另一过程（例如克隆操作）的存储器地址。当数据正从 A 到 B 时，数据还可同时从 A 到 O。但是，当正在进行数据克隆时，其它任何组合都是不可能的。这些其它组合是：A 到 B 和 O 到 B；A 到 B 和 B 到 O；以及 A 到 B 和 O 到 A。注意，在每种组合中，S 被假定为数据的发源地址。从而，本发明允许多个存储器移动源自相同的存储位置。但是，当目的地地址和被监听的源地址相同时，监听器根据哪个令牌首先被收到，向冲突的 SRC 令牌和 DST 令牌发出重试。

E. 裸写操作

裸写入

现在参见图 5B 的方框 527，本发明在存储器克隆体内引入新的写操作和相关的响应集合。该操作是不具有任何数据占有权的高速缓冲存储器线写入操作（也称为裸写入，因为该操作是不包括数据占有权的地址操作）（因此采用术语“裸”）。裸写入由存储器克隆体发出，以便开始数据克隆操作，并由包含数据将被移动到的目的地存储位置的存储器的存储器控制器接收。存储器控制器产生对裸写入的响应，所述响应被回送给存储器克隆体。

存储器克隆体从而发出不具有任何数据的写入命令（也可称为裸写入），所述写入命令被放在架构上，并为正被移动的数据启动目

的地缓冲器等分配。对应于地址 B，加上其它 31 个连续的页级地址扩展的每一个，存储器克隆体发出始于第一目的地地址的 32 个裸 CL 写入命令。图 4B 图解说明了裸写入和相关响应等的流水线（pipeline）操作。

存储器克隆体按照连续的、流水线方式发出 CL WR。流水线操作过程提供 DMA CL WR (B_0-B_{31})，因为数据被直接写入存储器中。32 个 CL WR 操作独立并在架构上重叠。

对裸 CL 写的响应

图 4B 图解说明高速缓冲存储器线（CL）读取（RD）和写入（WR）操作，以及在架构上传送的对应页的模拟线段（simulated line segment）（即 A_0-A_{31} 和 B_0-B_{31} ）。每个操作接收下面说明的一致性响应（coherency response）。如图所示，在不传送任何实际数据的情况下，发布裸 CL 写入操作。一旦发布了裸 CL WR，为每个裸写入操作产生一致性响应，所述一致性响应表示存储位置 B 是否可以自由接受正被移动的数据。根据特定目的地存储位置的存储器控制器是否能够分配的缓冲器以接收正被移动数据，所述响应可以是 Null（空）或者 Retry（重试）。

在图解说明的实施例中，缓冲器代表目的地存储器 207 的存储器高速缓存 213 的高速缓冲存储器线。在标准存储器操作过程中，发送给存储器的数据首先被保存在存储器高速缓存 213 中，稍后该数据被转移到物理存储器 207 中。从而，存储器控制器检查被用于保存特定裸 CL WR 操作的存储器地址的数据的特定高速缓冲存储器线。术语缓冲器有时可与高速缓冲存储器线交替使用，不过也可在不存在可构成缓冲器的正式存储器高速缓存结构的情况下，实现本发明。

一致性响应被送回存储器克隆体。该响应向存储器克隆体提供是否可在（经历一致性检查和源地址可用性）的时候开始数据传送的指示。当存储器控制器能够为裸 CL WR 分配缓冲器时，分配所述缓冲器，存储器控制器等待该 CL 的数据的接收。除了 Null/Retry 响应之外，还向每个裸 CLWR 提供目的地 ID 标记，如图 4B 中所示。参

考关于图 5D 说明的 CL 操作说明目的地 ID 的利用。

F. 结构完成状态

图 5C 图解说明依据其产生结构完成状态的过程，以及处理器对结构完成状态的响应。所述过程开始于方框 551，存储器克隆体监视关于发出的裸 CL WR 操作的 Null 响应，如方框 553 所示。在方框 555 确定发出的所有裸 CL WR 是否已从存储器控制器收到 Null 响应。当存储器控制器已向所有的裸 CL WR 操作发出 NULL 响应时，认为整个移动“结构完成”，如方框 557 所示，存储器克隆体用信号通知请求处理器，数据克隆操作已完成，即使还没有从存储器子系统读取要移动的数据。所述过程随后终止于方框 559。处理器重新开始执行后续指令（例如例证指令序列中，跟随 SYNC 的 ADD 指令）。

由于不从处理器或存储器克隆体接收数据，因此使结构完成状态的实现成为可能。即，要移动的数据不必被传送给处理器芯片或者存储器克隆体，而是直接从存储位置 A 转移到存储位置 B。一旦系统不再向处理器提供“旧的”目的地数据，处理器接收指示克隆操作已结构完成的指示。

从而，从而处理器的观点来看，即使在物理移动任意数据行之前，克隆操作可能看起来也已完成（取决于根据可用带宽、数据段的大小、重叠移动的数目、以及来回穿过接线器的其它过程等，能够多快完成物理移动）。当获得结构完成状态时，所有的目的地地址缓冲器已被分配，以便接收数据，存储器克隆体已发出触发把数据移动到目的地地址的相应读取操作。从系统同步的观点来看，虽然不是所有数据都已开始移动或者已完成移动，不过仍然通知处理器克隆操作已完成，处理器假定处理器发出的 SYNC 操作已收到 ACK 响应，所述 ACK 响应表示克隆操作的完成。

实现结构完成状态的一个优点在于使处理器免受存储器等待时间和系统布局的影响，因为在实际数据克隆操作完成之前，处理器不必等待。从而，在数据克隆指令被发送给存储器克隆体之后，分配给

数据克隆操作并且在收到 ACK 响应之前防止其处理后续指令的处理器资源被快速释放，以便以极小的延迟继续处理其它操作。

基于寄存器的结构完成状态跟踪

在一个实施例中，实现了基于硬件或软件寄存器的接收的 Null 响应的跟踪。寄存器设置在存储器克隆体 211 内，如图 2 中所示。例如在 CNT 值为 32 的情况下，存储器克隆体 211 配置一个 32 位软件寄存器 313，以便跟踪 32 个裸 CL 写入操作中的哪一个收到 Null 响应。图 7B 图解说明了用于向存储器克隆体提供克隆操作至少部分完成或者结构完成的指示的 32 位寄存器 313。该寄存器用作受存储器克隆体监视的进度条。代替实现 SYNC 操作，存储器克隆体利用寄存器 313 监视/记录已收到哪些 Null 响应。一旦收到了对应编号的裸 CL 写入操作的 Null 响应，那么每个二进制位被置为“1”。根据图解说明的实施例，由置于寄存器 313 的对应二进制位位置中的“1”所示，与二进制位 1、2 和 4 相关的目的地存储器地址的裸 CL 写入操作已完成。

在图解说明的实施例中，通过扫描寄存器的二进制位，了解所有二进制位是否被置为 1（或者是否有任何二进制位未被置为 1），完成结构完成状态的确定。另一种实现涉及对寄存器的各个二进制位中持有的数值进行“或”运算。本实施例中，在对无保持写操作的所有 Null 响应进行“或”运算之后，存储器克隆体用信号把完成状态通知处理器。当所有二进制位数值为 1 时，结构完成状态被确认，存储器克隆体向请求处理器发送指示。随后，整个寄存器 313 被重置为 0。

在图解说明的实施例中，一个 N 位寄存器被用于跟踪哪个保持写操作收到 Null 响应，这里 N 是大到足以覆盖为克隆操作发出的最大数目的写操作的设计参数。但是，在一些情况下，处理器只对了解特定高速缓冲存储器线是否处于结构完成状态感兴趣。对于这些情况，只读取或检查与关心的那些高速缓冲存储器线相关的特定寄存器位置，一旦这些特定的高速缓冲存储器线处于结构完成状态，那么存

存储器克隆体用信号通知处理器重新开始操作。

G. 借助目的地 ID 标记的直接存储器-存储器移动

读取请求

现在参见图 4B, 并参考图 5D 的流程图, 举例说明在裸写入操作之后发出读取操作的过程。该过程始于方框 571, 存储器克隆体监视对裸 CL WR 的 NULL 响应, 如方框 573 所示。在方框 575 确定是否收到 Null 响应。存储器克隆体重试所有未收到 Null 响应的裸 CL WR, 直到收到关于每个裸 CL WR 的 Null 响应为止。如方框 577 所示, 当在存储器克隆体收到 Null 响应时, 立即在架构上向当前要移动的数据段驻留于其中的源存储位置发送相应的 (地址) CL 读取操作。例如, 关于裸 CL WR (B_0) 收到的 Null 响应导致在架构上放置 CL RD (A_0), 依此类推, 如图 4B 中所示。源存储位置的存储器控制器检查源存储器内的特定地址是否可用于 CL 读取操作正在请求的源数据 (即另一过程目前是否没有使用该地址位置或数据)。该检查导致一个 Null 响应 (或 Retry 响应)。

在一个实施例中, 当正被克隆的数据的源地址不可用于 CL RD 操作时, 使该 CL RD 操作排队等待, 直到源地址变成可用时为止。因此, 不需要重试。但是, 对于提供重试而不提供 CL RD 操作的排队的实施例来说, 用信号通知存储器克隆体重试具体的 CL RD 操作。

架构上的目的地 ID 标记

如图 4B 中所示, 和对裸 CL WR 的 Null 响应一起, 目的地 ID 标记由目的地存储器的存储器控制器发出。产生的目的地 ID 标记随后可被附到或者插入 CL RD 操作内 (而不是处理器的 ID, 或者除了处理器的 ID 之外)。根据图解说明的实施例, 借助相应的 CL RD 请求以把目的地 ID 标记放在架构上。目的地 ID 标记是提供给 CL RD 请求以识别读取操作所请求的数据将被返回的位置的路由标记。具体地说, 目的地 ID 标记识别将接收被相关 CL RD 操作移动的数据的存储缓冲器 (分配给裸 CL WR 操作)。

图 7A 图解说明了在架构上发送的读写地址操作 705 以及目的地 ID 标记 701（包括存储器克隆体标记 703）。这两者被用于区分在架构上重叠的多个克隆操作。如图 7A 中所示，地址操作 705 包括 32 位源（SRC）或目的地（DST）页级地址和附加的 12 个保留位，所述 12 个保留位包括被存储器克隆体 211 的控制逻辑单元 303 用于提供页级寻址的 5 位。

与地址操作 705 相关的是目的地 ID 标记 701，目的地 ID 标记 701 包括发出该操作的存储器克隆体的 ID，操作的类型（即 WR、RE、令牌（SRC）或令牌（DST）），计数值（CNT）和发送操作的响应/数据的目的地单元的 ID。如图所示，最初利用 ID 字段中的存储器克隆体地址，发出写操作，如图 7A 的 WR 标记中所示。在 RD 操作中，用实际目的地存储器地址替换 SRC 地址，如图 7A 的 RD 标记中所示。

直接源-目的地移动

为了实现直接存储器-存储器数据移动，而不是经过请求处理器（或者存储器克隆体）的移动，存储器克隆体用目的地存储器位置（B）的真实存储器地址（即目的地 ID）替换 CL RD 操作的标记中的物理处理器 ID。如下所述，这使数据能够直接被发送给存储器位置 B（而不必经过存储器克隆体）。

在目前的系统中，发出读取请求的处理器或处理器芯片的 ID 包含在读取请求内，或者作为标记被提供给读取请求，以便识别数据将被返回给的组件。即，该 ID 指向读操作的源，而不是数据将被移往的最终目的地。

存储器控制器自动把数据发送给在目的地标记中规定的位置。从而，对目前的系统来说，数据被发送给处理器。但是，根据这里描述的实施例，由于路由地址是最终（存储器）目的地的路由地址，因此源存储器控制器必须把数据直接发送给目的地存储器。数据通过接线器，从源存储器被直接转移到目的地存储器。决不会通过处理器或存储器克隆体发送数据，消除了从处理器的数据路由操作。特别地，

在其中在相同物理存储块内移动数据的实施例中，可在不把数据外送给外部接线器构架的情况下，完成数据克隆。

跟踪数据克隆操作的完成

在一个实施例中，为使存储器克隆体了解克隆操作何时完成，提供一个软件激活的克隆完成寄存器，该寄存器跟踪哪些高速缓冲存储器线（或者多少数据部分）已完成克隆操作。由于从发出地址到数据产生其通过接线器到达目的地的路线的中间时间的缘故，装入完成寄存器被用作已到达存储位置 $B_0 \cdots B_n$ 的数据部分 $A_0 \cdots A_n$ 的数目的计数器。在一个实施例中，存储器克隆体根据所有的读取地址操作何时收到指示所有数据正在架构上向目的地存储位置转移的 Null 响应，跟踪实际移动的完成。

在利用软件寄存器的备选实施例中，寄存器包括数目和 CNT 值相同的二进制位。从而，每一位对应于正被移动的数据页的具体段（或者 CL 区组（granule））。克隆完成寄存器可以是存储器克隆体的组成部分，如图 3 中所示，克隆完成寄存器 317 被用于跟踪克隆操作的进展，直到克隆操作的所有数据已被克隆到目的地存储位置为止。

H. 一致性协议（coherency protocol）和操作

完成数据克隆操作时的一个需要考虑的重要因素是数据必须来源于包含该数据的最一致副本的存储位置或高速缓冲存储器。从而，虽然本发明被描述成数据直接来源于存储器，不过本发明的实际应用允许数据来源于高速缓冲存储器/存储器子系统的任意一致位置。图 6B 图解说明了存储器子系统的一种可能结构。

在链接系统 600 的组件的背景下图解说明了接线器 603，系统 600 的组件包括处理器 611、613 和存储器子系统的各个组件。如图所示，存储器子系统指的是分布式主存储器 605、607，处理器（L1）高速缓冲存储器 615、617，低级（L2-Ln）高速缓冲存储器 619、621（也可以是干涉（intervening）高速缓冲存储器），以及任意类似来源。这些存储器组件中的任何一个都可包含数据将被移动

时，数据的最一致副本。特别地，如图 2 中所示及上面所述，存储器控制器 608 包括克隆数据将被移动到其中的存储器高速缓存 213（这里也称为缓冲器）。由于发送给存储器的数据首先被保存在存储器高速缓存 213 中，随后被移动到实际的物理存储器 607，因此存储器高速缓存 213 常包含目的地地址的数据的最一致副本（即呈 M 状态的数据）。

在一些高级系统中，在通过外部（架构）总线 663 连接的不同系统之间共享数据。如同这里所示那样，外部存储器子系统 661 包含与存储器地址 C 相关的存储位置。该存储位置内的数据可代表数据克隆操作的源数据的最一致副本。与外部存储器子系统 661 的连接可通过局域网（LAN）或者甚至通过广域网（WAN）。

可采用常规的一致性协议（例如，修改（M）、独占（E）、共享（S）、无效（I））或者关于一致数据的来源的 MESI 协议；但是，这里利用的一致性协议扩展了常规的协议，以便允许存储器克隆体获得高速缓冲存储器线的所有权，并完成裸 CL WR 操作。

低级高速缓存均具有相应的高速缓存控制器 620、622。当数据直接来源于除分布式主存储器 605、607 之外的位置，例如低级高速缓存 619 时，该高速缓存的相关控制器（高速缓存控制器 620）按照和存储器控制器 606、608 相同的方式控制从该高速缓存的数据转移。

存储器高速缓存控制器对裸写操作的响应

就除了存储器之外，还包括高级和低级高速缓冲存储器的存储器子系统来说，可在高速缓冲存储器之间共享源地址和目的地地址的一致数据，并且任一地址的一致数据可存在于高速缓冲存储器之一中，而不是存在于存储器中。即，存储器子系统起完全相关存储器子系统的作用。对源地址来说，数据总是来源于最相关的存储位置。但是，对于目的地地址来说，一致操作根据标准 MESI 协议变化，如下所述。

当目的地存储位置的存储器控制器收到无保持写操作时，存储

器控制器以三种主要的监听响应之一，应答各个裸写操作。不同的裸写操作的单各响应被转发给存储器克隆体。这三种主要的监听响应包括：

1. **Retry** 响应，该响应表示存储器高速缓存具有呈 M 状态的数据，但是不能转到 I 状态和/或存储器控制器目前不能接受 WR 请求/向 WR 请求分配缓冲器；

2. **Null** 响应，该响应表示存储器控制器可接受 WR 请求，并且相应高速缓冲存储器线的一致状态立即转为 I 状态；和

3. **Ack_Resend** 响应，该响应表示存储器高速缓存内 CL 的一致状态已从 M 状态转变成 I 状态，但是存储器控制器仍然不能接受 WR 请求（即，存储器控制器仍然不能分配缓冲器接收正被移动的数据）。

后一响应（**Ack_Resend**）是使存储器克隆体开始保护 CL 数据（即，向请求访问高速缓冲存储器线的其它组件发送重试）的组合响应。修改的数据从高速缓冲存储器线丢失，因为该高速缓冲存储器线被置为 I 状态，如下所述。存储器控制器稍后分配存储器高速缓存内的地址缓冲器，所述地址缓冲器被保留，直到恰当的读取操作完成为止。

高速缓冲存储器线无效和线的存储器克隆体操作

根据例证的实施例，裸写操作使完全相关的存储器子系统中的所有相应高速缓冲存储器线无效。具体地说，每当存储器克隆体发出针对存储器高速缓存的修改高速缓冲存储器线（即该线处于 MESI 或者其它类似一致性协议的 M 状态）的裸 WR 时，存储器控制器响应对裸写操作的监听，把高速缓冲存储器线的相关状态更新成无效（I）状态。

另外，裸 WR 不会导致存储器高速缓存的“重试/推送”操作。从而，和标准的一致操作不同，当在存储器高速缓存收到裸写操作时，修改后的数据不会从存储器高速缓存推送到存储器。裸写操作立即使当前的修改数据无效。在实际移动操作之后，克隆数据的新的

高速缓冲存储器线被赋予 M 一致状态，随后被用于按照标准一致操作，响应关于位于相应地址空间的数据的后续请求，提供数据源。

当使高速缓冲存储器线无效时，存储器克隆体启动高速缓冲存储器线的保护，并起修改后的监听器的作用。即，存储器克隆体负责完成高速缓冲存储器线的所有一致保护，好像高速缓冲存储器线处于 M 状态一样。例如，如图 5A 的方框 511 所示，如果在实际完成克隆操作之前，另一过程需要该数据（例如保存在 A_0 的数据的读取被监听），那么存储器控制器或者重试或者延迟发送数据，直到实际完成数据的物理移动为止。从而，来自其它组件的关于该高速缓冲存储器线的监听请求被重试，直到数据已被克隆，并且高速缓冲存储器线状态变回 M 状态为止。

图 8B 图解说明了根据本发明一个实施例的关于存储器克隆操作的过程，依据该过程完成一致操作。该过程始于方框 851，之后如方框 853 所示，存储器克隆体发出裸 CL WR。在图解说明的过程中，所有监听器监听该裸 CL WR，如方框 855 所示。具有最高一致状态的监听器（这种情况下为存储器高速缓存）随后把高速缓冲存储器线状态从修改（M）变成无效（I），如方框 857 所示。

特别地，和常规的一致性协议操作不同，在使数据无效之前，监听器不启动向存储器的数据推送。相关的存储器控制器用信号通知存储器克隆体，该存储器克隆体需要向高速缓冲存储器线提供保护。因此，当存储器克隆体被赋予保护高速缓冲存储器线的任务时，该高速缓冲存储器线立即被标记为 I 状态。在高速缓冲存储器线处于 I 状态的情况下，存储器克隆体从而接管使该高速缓冲存储器免受监听的全部责任。

返回图 8B，随后在方框 859（由目的地存储器控制器）确定高速缓冲存储器线用缓冲器是否可用。如果缓冲器不可用，那么发出 Retry 监听响应，如方框 861 所示。存储器克隆体然后重新发送 CL WR，如方框 863 所示。但是，如果缓冲器可用，那么存储器控制器把该缓冲器分配给监听的裸 CL WR，如方框 865 所示。

随后，开始数据克隆过程，如方框 867 所示。当数据克隆过程完成时（如方框 869 所示），持有克隆数据的高速缓冲存储器线的一致状态被改变成 M 状态，如方框 871 所示。随后，终止数据克隆过程，如方框 873 所示。在一种实现中，目的地存储器控制器（MC）可能不具备可用于裸 CL WR 的地址缓冲器，并发出 Ack_Resend 响应，所述 Ack_Resend 响应导致稍后重发裸 CL WR，直到 MC 可接受裸 CL WR，并分配相应的缓冲器为止。

活锁避免

提供一种避免活锁的新方法。该方法包括当裸 WR 在飞行过程中（in flight）时，使修改的高速缓冲存储器线无效，以避免活锁。

图 8A 图解说明了根据本发明的处理当正在完成裸写入，以及随后的数据物理移动时的锁定争用的过程。该过程开始于方框 821，随后前进到方框 823，方框 823 表示处理器 P1 请求从位置 A 到 B 的高速缓冲存储器线移动。在裸 WR 和自数据源的数据的物理移动之前，P1 和/或 P1 启动的过程获得关于该存储位置的锁定。处理器 P2 随后请求访问位于目的地或源地址的高速缓冲存储器线，如方框 825 所示。

在方框 827，（目的地存储器控制器）确定实际移动是否已完成（即 P1 可释放锁定）。如果实际移动已完成，那么向 P2 提供对该存储位置的访问，P2 随后可获得锁定，如方框 831 所示，随后在方框 833 终止该过程。但是，如果移动仍然正在进行中，那么根据正实现的实施例，提供两种路径之一。在第一实施例中，如方框 829 所示，向 P2 返回 Retry 响应，直到 P1 放弃对该高速缓冲存储器线的锁定为止。

在另一实施例中，如果实际移动还没有开始，并且请求与从位置 A 读取数据有关，那么从位置 A 提供数据。这使多个过程能够从相同的源位置获得数据，而不是发出 Retry。但是，当数据正被移动时，在数据已完成移动之前，总是重试对目的地地址的访问请求。

I. 多个同时进行的数据移动和标记标识符

多个存储器克隆体和重叠的克隆操作

完成裸写操作并向 CL RD 请求分配标记的方法的一个关键益处在于借助大量的存储器克隆体，可在系统上实际多个克隆操作。本发明从而允许多个独立的存储器克隆体，每个存储器克隆体可进行与架构上的另一存储器克隆体的另一数据克隆操作重叠的数据克隆操作。特别地，在请求锁定（或者锁定获得）情况下，存储器克隆体的操作使多个存储器克隆体能够发布（issue）同时进行的克隆操作。

在举例说明的实施例中，每个芯片只提供一个存储器克隆体，导致每次从每个芯片只完成一个克隆操作。在多个处理器芯片共用单一存储器克隆体的另一实施例中，存储器克隆体包括确定在指定时间，哪个处理器获得访问的仲裁逻辑单元。仲裁逻辑单元可由 FIFO 队列代替，FIFO 队列能够按照从处理器接收的顺序，保持多个存储器移动操作的完成。其它实施例可提供每个处理器芯片粒度（granularity）增大的存储器克隆体，实现每个芯片的多个存储器克隆操作，每个克隆操作由独立的存储器克隆体控制。

本发明还允许多个存储器克隆体同时工作。存储器可通过令牌操作相互通信，并且每个存储器克隆体把其克隆操作的源地址和目的地地址通知其它存储器克隆体。如果第一存储器克隆体的目的地地址和已进行数据克隆操作的第二存储器克隆体的源地址相同，那么第一存储器克隆体延迟克隆操作，直到第二存储器克隆体完成其实际数据移动为止。

借助目的地 ID 和附加标记，识别多个克隆操作

除了实现直接源-目的地克隆操作之外，当来自多个克隆操作的数据重叠或者正在同时完成时，目的地 ID 标记还被用于唯一地识别架构上的数据占有权。由于每次只有来自单一克隆操作的数据可被发送给任意目的地存储器地址，因此每个目的地地址必须是唯一的。

在另一实现中，一组附加的二进制位被附到图 7A 的数据标记 701 的数据路由部分。这些二进制位（或者克隆 ID 标记）703 唯一地识别来自特定克隆操作的数据和/或识别与克隆操作相关的存储器克

隆体。因此，附加二进制位的实际数目以系统设计员所希望的具体实现为基础。例如，在只具有两个存储器克隆体的最简单实现中，单个二进制位可被用于区分第一克隆操作的数据（与第一存储器克隆体相关）和第二克隆操作的数据（与第二存储器克隆体相关）。

显然，当只把少量的二进制位用于识别不同的数据路由操作时，如果使用的每个标记唯一，那么克隆 ID 标记 703 严格限制了同时进行的克隆操作的数目。

目的地 ID 和克隆 ID 标记的组合

唯一地识别不同克隆操作/数据的另一方式是利用目的地 ID 和克隆 ID 标记的组合。就这种实现来说，由于特定克隆操作的目的地 ID 不可能和另一未决克隆操作的目的地 ID 相同（归因于下面说明的一致性和锁定争用问题），克隆 ID 标记的大小可以相对较小。

如图 7A 中图解所示，标记与各数据克隆操作相关（与之链接、附加在其上，或者以其它方式与数据克隆操作相关）。从而，如果第一数据克隆操作涉及从一页移动 12 个独立的高速缓冲存储器线，那么这 12 个数据克隆操作中的每一个被赋予相同的标记。同时进行的涉及 20 段数据的移动的第二克隆操作例如也使每个数据移动操作由第二标记标记，所述第二标记不同于第一克隆操作的标记，依次类推。

可重新使用的标记标识符

存储器克隆体使用的各高速缓冲存储器线由地址字段内的 12 个保留位的头 5 位确定。由于存在 12 个保留位，因此或多或少的地址都是可能的。在一个实施例中，其它保留位都被用于提供标记。从而，虽然参考独立的克隆标记标识符描述了本发明，不过在高位二进制位被分配给目的地 ID 的情况下，地址字段的低位二进制位可容易地提供所述特征。

在一个实施例中，为了简化大量的存储器克隆操作（例如在大规模多处理器系统中），一旦不再在架构上发送先前的数据，那么就重新使用克隆 ID 标记 703。在一个实施例中，通过使标记足够大，

以致它包含数据移动可能采取的最大间隔，实现标记重新使用。

在举例说明的实施例中，标记被设计成可重新使用的位序列，并选择在标记使用和重新使用期间，避免任意标记冲突所需的二进制位的最小数目（即，确定为设计参数）。该确定涉及处理器数目、重叠克隆操作的可能数目、完成克隆操作的时间长度的考虑。可顺延分配标记，并且当分配该序列中的最后一个标记时，第一标记应可被自由分配给发布的下一克隆操作。从而，在系统级实现标记退役（retirement）和重新使用的过程，以致一旦使第一个发出的标记退役（即，相关的数据克隆操作完成），那么就可重新开始标记编号。

另一实施例提供一种克隆 ID 标记，所述克隆 ID 标记包括在每个克隆操作或存储器克隆体被分配唯一的一个编号的情况下，涵盖同时进行的克隆操作的最大可能数目所需的二进制位。对于任一实施例来说，都不发生克隆 ID 标记的重叠。

可采用确保标记释放，包括何时重新使用标记的几种可能方法。在一个实施例中，为了确保标记可被释放并被重新使用，需要确认。一旦数据克隆操作完成，存储器克隆体从目的地存储器控制器接收释放的确认。

基于标记的冲突的重试

一个实施例引入了基于标记的冲突的重试概念。根据该实施例，标记是可以重新使用的，并且不必是唯一的。从而，当后续克隆操作被赋予“001”的标记号时，具有该标记的第一克隆操作可能仍然正在进行。发生这种情况时，拥有第一克隆操作的第一存储器克隆体监听对后续克隆操作的标记分配（或者接收和对后续克隆操作的标记分配有关的信号）。第一存储器克隆体随后立即向拥有后续克隆操作的第二存储器克隆体的裸写操作发出基于标记的重试。下一存储器克隆体延迟后续克隆操作，直到第一克隆操作完成（即，数据已被移动）为止。

J. 结构化（ARCHITECTED）二进制位和 ST 克隆操作

目前的多数处理器与阻止（hold up）架构上指令的执行的外部

中断一起工作。外部中断特征由硬件位提供，所述硬件位由操作系统（OS）设置。在中断位被断言（assert）或去断言（de-assert）的情况下，OS 设置处理器操作状态。当被断言时，可在指令流的执行过程中的任意时刻发生中断，处理器或应用程序都不能控制何时发生中断。

缺少对外部中断的控制是外部架构上移动操作期间需要考虑的问题。具体地说，移动操作包括处理器发出一系列的指令（例如，6个连续指令）。为了在该系列指令的执行过程中，不发生中断的情况下完成移动操作，在发出实现移动操作的一系列指令之前，处理器必须首先确保对架构的锁定。这意味着每次只有一个处理器可执行移动操作，因为锁定只可提供给一个请求处理器。

根据本发明的一个实施例，实现外部中断（EE）位的断言和去断言的特征被修改，以便允许在处理器上运行的软件断言和去断言中断位。即，用特殊指令对应用程序编码，所述特殊指令可切换（toggle）外部中断（EE）位，从而允许处理器发出特定的指令序列，该指令序列不会被中断。

去断言 EE 位，使得处理器在发出指令序列之前，不必确保对外部架构的锁定。从而，多个处理器能够同时发出它们各自的指令序列。当应用于数据克隆操作时，该特征允许多处理器系统中的多个处理器同时执行克隆操作，而不必分别获得锁定。这还使得每当每个处理器需要完成数据克隆操作时，该处理器就能够开始数据克隆。另外，如下所述，无中断情况下指令的发布，使存储器克隆体能够以流水线方式发出一系列指令。

在举例说明的实施例中，结构化 EE（外部中断）位被用于动态转换处理器的操作状态，以便包括中断或者不包括中断。在共同构成克隆操作的一系列指令之间不存在中断的情况下，在架构上执行该系列指令。应用程序内的程序代码切换 EE 位，以便动态禁用和启用外部中断。该特定指令序列的应用软件超越（over-ridden）OS 选择的中断状态。根据举例说明的实施例，EE 位可被在处理器上运行的应

用程序设置为 1 或 0，每个值对应于具体的中断状态，这取决于处理器的设计以及和 EE 位相关的软件编码值。

从而，本发明提供一种软件编程模型，当外部中断被禁用时，所述软件编程模型能够发出多个指令。就举例说明的实施例来说，共同完成移动或克隆操作的指令序列之前是由如下例证代码序列所示的去断言 EE 位的指令：

EE bit=0

ST A

ST B

ST CNT

EE bit=1

SYNC

在上面举例说明的实施例中，当 EE 位的值为 0 时，外部中断被关闭。指令从处理器流到存储器克隆体。随后，EE 位的值被改为 1，表示处理器状态返回允许外部中断的中断启用状态。之后，在架构上发布 SYNC 操作。

ST CLONE 操作

在一个实施例中，存储器克隆体（或者处理器）把上述指令序列看作代表克隆操作，并自动设置 EE 位，以便防止外部中断打断该指令序列。在另一实施例中，上述指令序列被存储器克隆体接收为组合的基本存储操作。这里，组合操作被称为 Store (ST) CLONE 操作，并用单个 ST CLONE 操作替代上述一系列的三个独立存储操作和一个 SYNC 操作。

ST CLONE 是使存储器克隆体启动克隆操作的多字节存储操作。设置 EE 位使存储器克隆体能够用 ST CLONE 操作替换其后跟随 SYNC 的上述一系列的存储指令。

从而，可用单个 ST CLONE 操作替换 4 个独立操作（即 3 个存储操作及其后的一个 SYNC 操作）。另外，根据本发明的这种实现，SYNC 操作是虚拟的，因为一旦存储器克隆体检测到结构完成状

态，就用信号把数据克隆操作的完成通知给处理器。结构完成状态使处理器表现为好像在存储器克隆操作之后，发布的 SYNC 已收到 ACK 响应。

K. 借助保留位的虚拟/真实地址操作模式

本发明能够实现处理单元的虚拟或真实寻址能力的基于应用的动态选择。在可能影响存储器中数据的位置的各个指令（例如 ST 指令）内，提供一个保留位，所述保留位可由应用软件（即不是 OS）设置，以便把处理器的操作模式选择为虚拟寻址或真实寻址模式。图 9A 图解说明了具有保留位 901 的地址操作 900。保留位 901 能够由运行于处理器上的应用软件动态设置。根据运行于处理器上的应用程序提供的代码，处理器操作模式从虚拟模式变为真实模式，以及从真实模式变成虚拟模式。

保留位 901 表示是需要真实寻址还是需要虚拟寻址，保留位由运行于处理器上的应用软件赋值（1 或 0）。为“0”的默认值可用于表示虚拟寻址，当需要真实寻址时，软件可动态地把该值变为“1”。处理器读取保留位的值，确定特定的地址操作需要何种操作模式。

虚拟或真实寻址模式的选择可由运行于处理器上的特定应用程序确定。当应用程序需要寻求性能的提高，而不是数据的保护时，选择虚拟操作模式，允许应用程序直接把有效地址发送给 OS 和固件。

图 9B 图解说明了典型软件环境的软件层图以及地址操作的相关默认操作模式。如图所示，应用软件（software application）911 按照寻址模式操作，而 OS 913 和固件 913 按照真实寻址模式操作。通过把保留位设置成虚拟寻址模式的预定值，实现提供性能提高的模式的选择。同样地，当要求数据保护时，保留位被设置成指示虚拟寻址模式的值，虚拟数据地址被发送给存储器克隆体 211，存储器克隆体 211 中的 TLB 319 稍后提供相应的真实地址。从而，本发明能够实现性能与数据保护的软件方面的均衡。

虚拟地址模式的处理器操作由存储器克隆体 211 的虚拟-真实地

址转换后备缓冲器 (TLB) 支持。当从处理器接收存储器克隆体操作及虚拟地址时, TLB 被用于把地址从虚拟地址转换成真实 (或物理地址)。随后, 在架构上发出之前, 虚拟地址被转换成真实地址。从 OS 的观点来看, 虚拟寻址模式能够实现用户级特权, 而真实寻址模式则不能。从而, 虚拟寻址模式使数据能够被用户级应用程序以及被 OS 访问。另外, 虚拟寻址模式允许操作系统 (OS) 和用户级应用程序访问存储器克隆体。真实地址操作模式能够实现更快的性能, 因为一旦指令从处理器被发出, 就不需要地址转换。

L. 其它特征、概述和益处

作为数据移动操作的目标的数据来源于实际存储器、处理器高速缓存、低级高速缓存、干涉高速缓存等中的最一致存储位置。从而, 源地址还表示存储器子系统内包含被请求数据的一致副本的正确存储模块。

本发明使多个克隆操作能够在架构上重叠 (或者同时被执行)。为了监视并唯一地区分每个独立克隆操作的完成, 提供一个标记, 所述标记被附到发送给源地址的读取操作的地址标记上。该标记可保存在 M 位寄存器中, 其中每个克隆操作具有置于该寄存器中的不同值, M 是为支持系统上同时进行的可能克隆操作的最大数目而选择的设计参数。

如上所述, 一旦裸 WR 过程完成, 即结构性完成移动。结构完成状态和其它相关特征的实现相当快地使处理器脱离数据移动操作。数据的所有物理移动 (代表存储移动中涉及的相当大部分等待时间) 发生于后台中。处理器能够相当快地重新开始处理指令序列中跟在 SYNC 之后的指令, 因为在产生结构完成状态的裸写过程中不包含任何数据传输阶段。

特别地, 在数据在相同存储模块上的地址之间移动的情况下, 时间益处甚至更显著, 因为数据不必在外部接线器架构上传输。利用存储器芯片上互连存储模块的不同子组件 (例如控制器等) 的上层金属便于这种 “内部” 存储移动。图 6C 中提供了存储模块的这种结

构。从而在其中存在在不同组件之间的互连的接线器实现中，借助存储模块 605 的上层金属 651，也可实现直接内部数据克隆。

本发明提供其它几个可以确认的益处，包括：（1）移动数据不滚动（roll）高速缓冲存储器（L2、L3 等），如同传统处理器启动移动那样；和（2）由于结构完成处理器状态的缘故，执行应用软件也极快地完成。例如，在现有技术中，如下进行 128B CL 移动（LD/ST）：LD/ST: 1 CL RD_x（地址和数据），32 CL RD_y（地址和数据），32 CL WR_y（地址和数据）。该操作实际上是 3 个地址操作和 384 字节的数据事务（transaction）。但是，借助本发明，利用一个裸 CL WR_y（仅地址）和 1 CL RD_x（仅地址）总线事务完成相同的过程。从而，获得显著的性能提高。

本发明利用带有分布式存储器结构的基于接线器的多处理器系统的数个当前可用特征/操作，从处理观点提供更有效的数据移动。例如传统上，MS 控制来自于存储器（高速缓冲存储器线）的数据相对于处理器的发送和接收。MC 被提供地址和源 ID，并利用这两个参数转发所请求的数据。通过用与高速缓冲存储器线读取相关的地址标记中的目的地 ID 替换源 ID，本发明能够实现被移动数据的直接 MC-MC 传输（即发送和接收），而不需要改变传统的 MC 逻辑单元和/或功能。

接线器还使得能够同时进行多个存储器克隆操作，这进一步导致存储队列/缓冲器的有效利用。借助直接接线器连接，数据移动中所涉及的时间也不取决于存储器克隆操作的量。

本发明借助硬件的虚拟化和数个软件可控特征的包含，改进了具有加速器引擎的当前处理器的基于硬件的移动操作。即，在不实际利用传统用于完成移动操作的硬件组件的情况下，观察到并改进了硬件模型的性能优点。

另一例子涉及利用接线器在架构上实现更快的数据移动，因为被移动的高速缓冲存储器线不再必须通过单一点（即进出单一处理器芯片，所述处理器芯片一般接收，并且随后发送被移动的所有数

据)。另外, 由于实际的数据移动不需要相对于单一收集点的传输, 因此利用接线器能够实现(多个)高速缓冲存储器线的并行移动, 导致对更高带宽的使用, 从而快得多地完成所有物理移动。现有系统每次只能完成一次移动。

本发明还能够实现字节、高速缓冲存储器线和页的移动。虽然没有提供和移动何时实际开始有关的任何实际时间, 不过存储器克隆体跟踪该信息, 并保持处理系统的一致性。特别地, 由于该移动不会影响在正在完成实际移动时实现的任何其它过程, 因此处理器资源可以自由完成其它任务, 而不是等到数据从一个存储位置被移动到另一存储位置为止。

虽然已参考具体实施例说明了本发明, 不过所述说明不应被理解为对本发明的限制。当参考本发明的说明时, 对本领域的技术人员来说, 所公开实施例的各种修改, 以及本发明的其它实施例将变得显而易见。于是可认为在不脱离附加权利要求中限定的本发明的精神或范围的情况下, 可做出这样的修改。

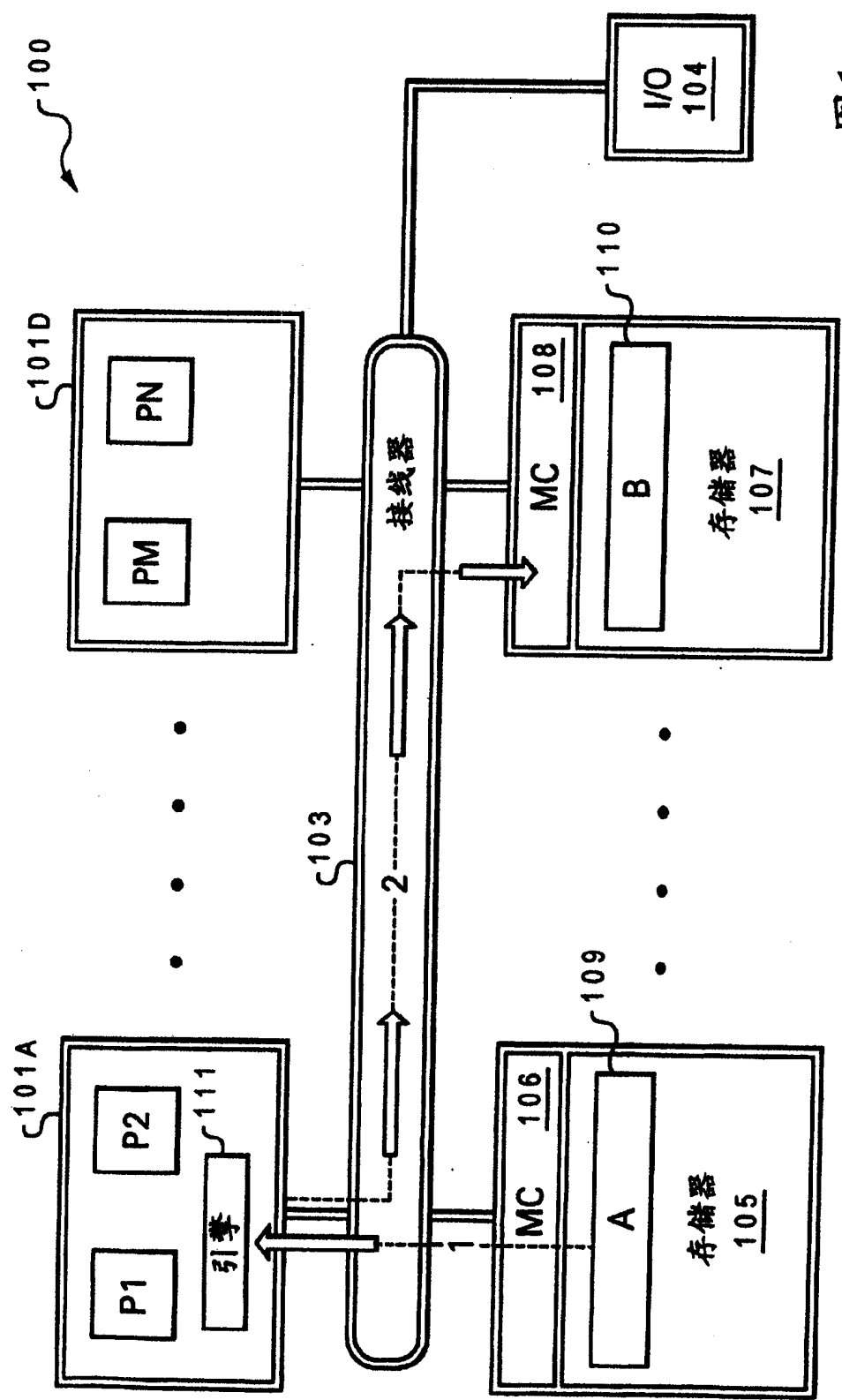


图1
现有技术

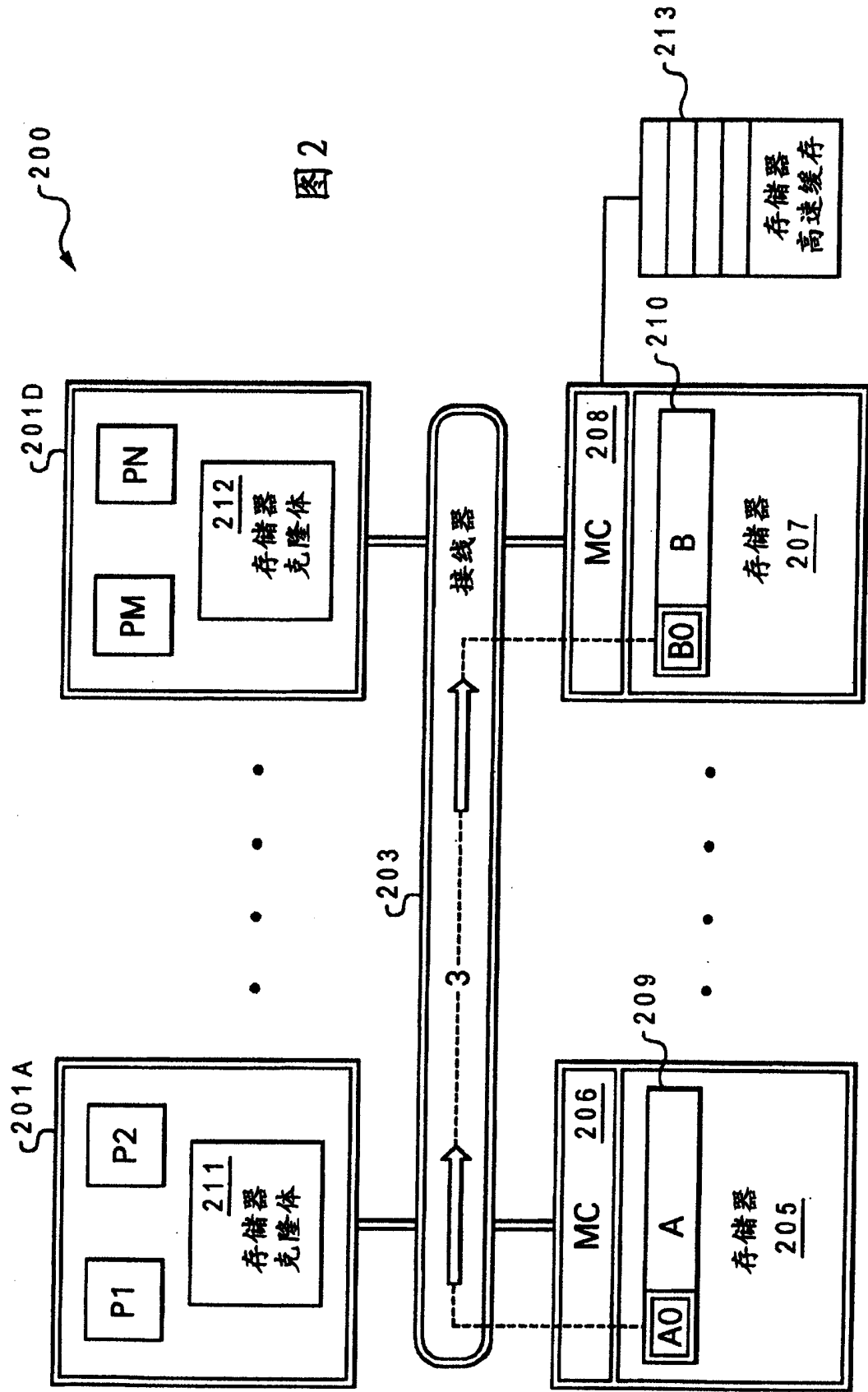
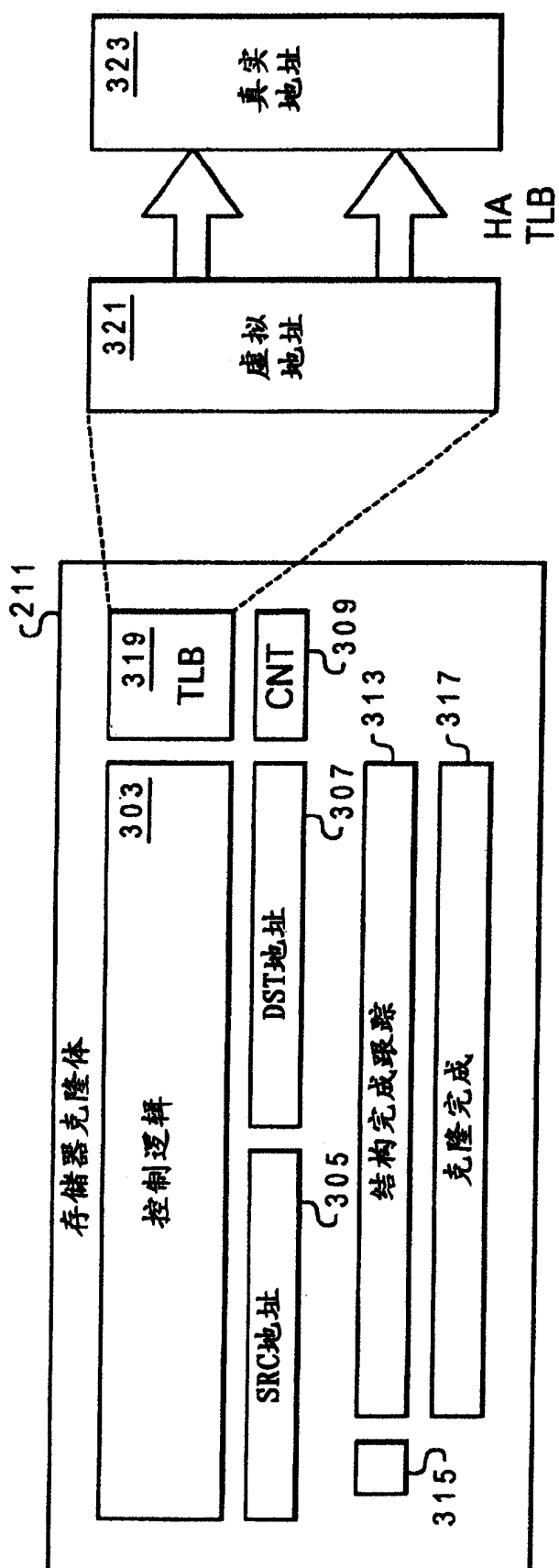


图2



3
图

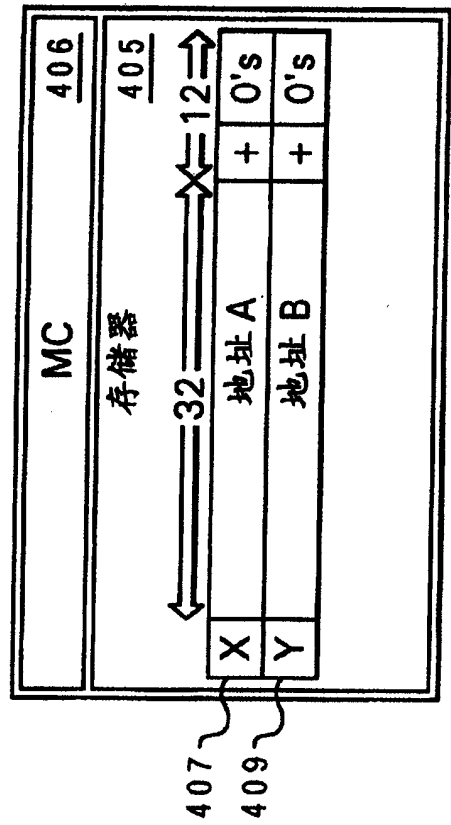


图 4A

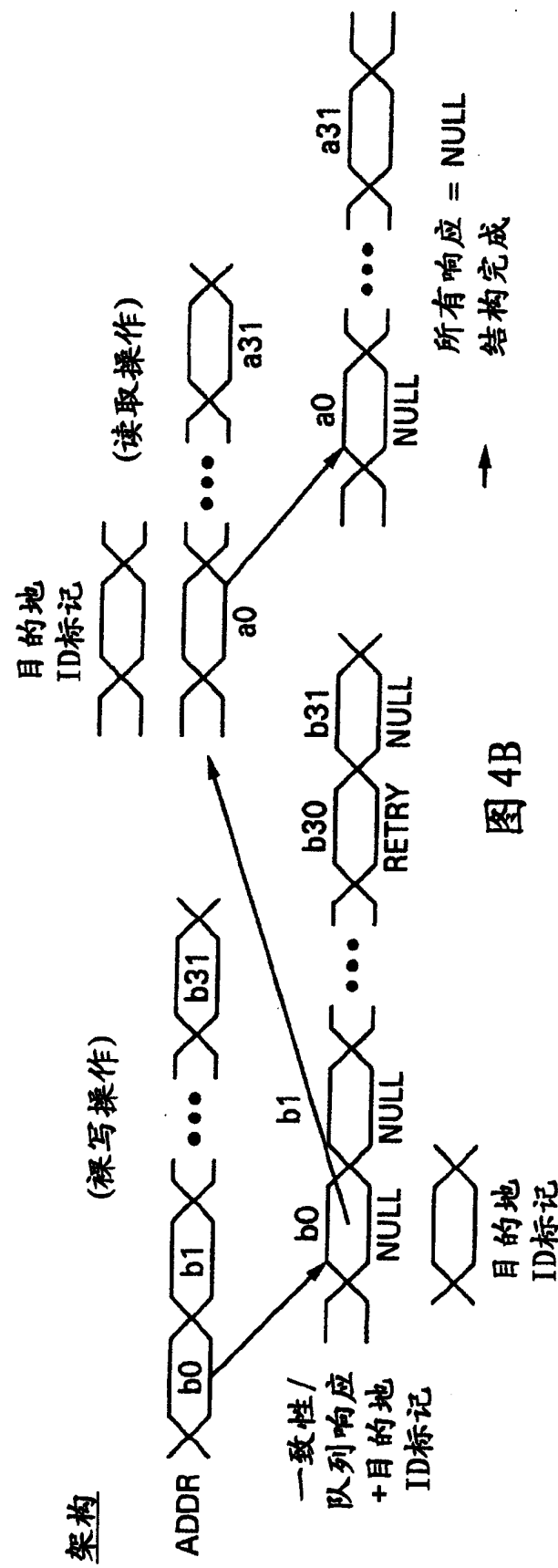


图 4B

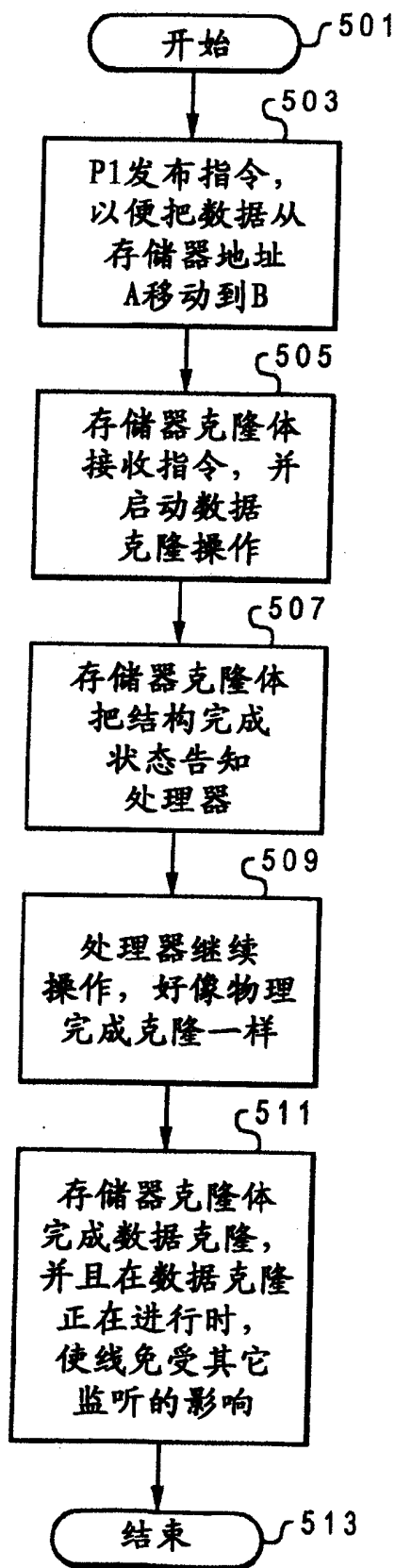


图 5A

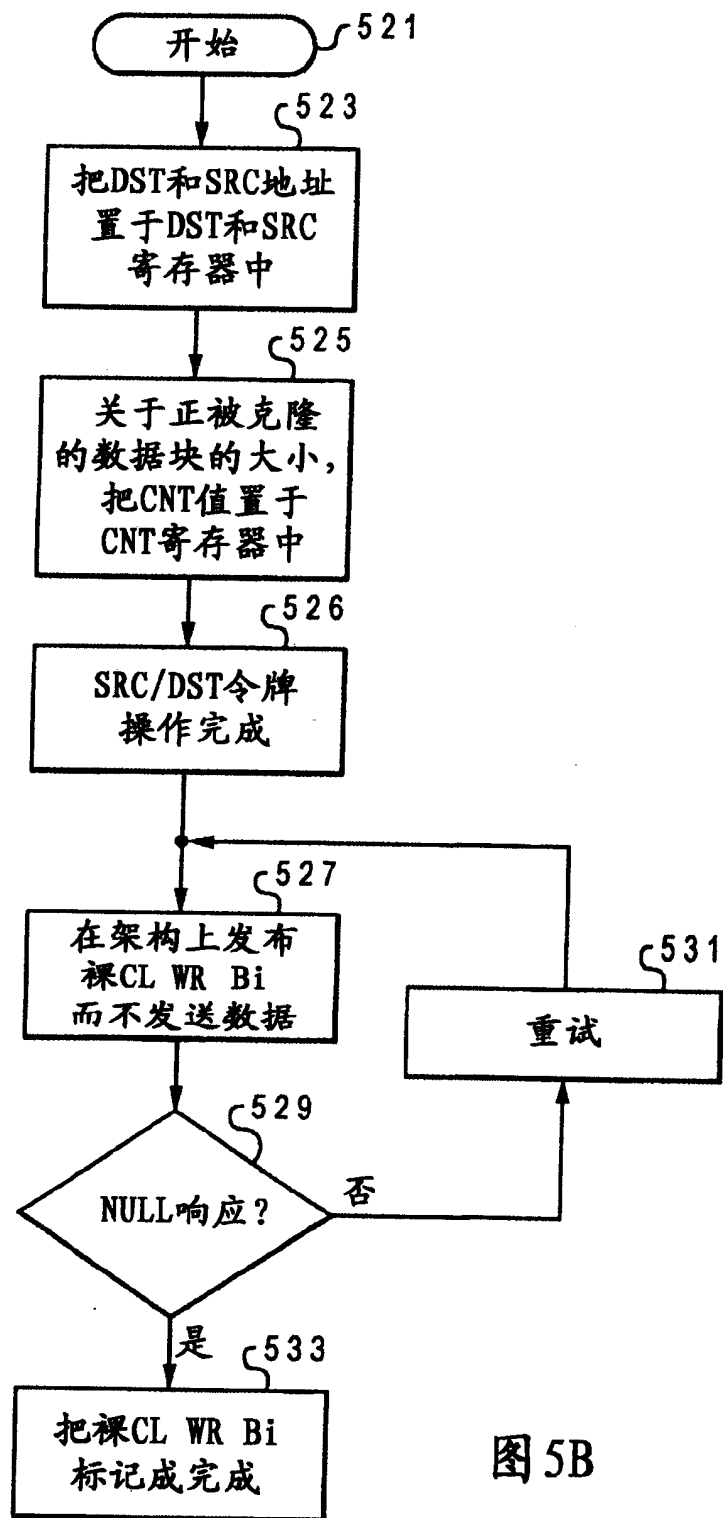


图 5B

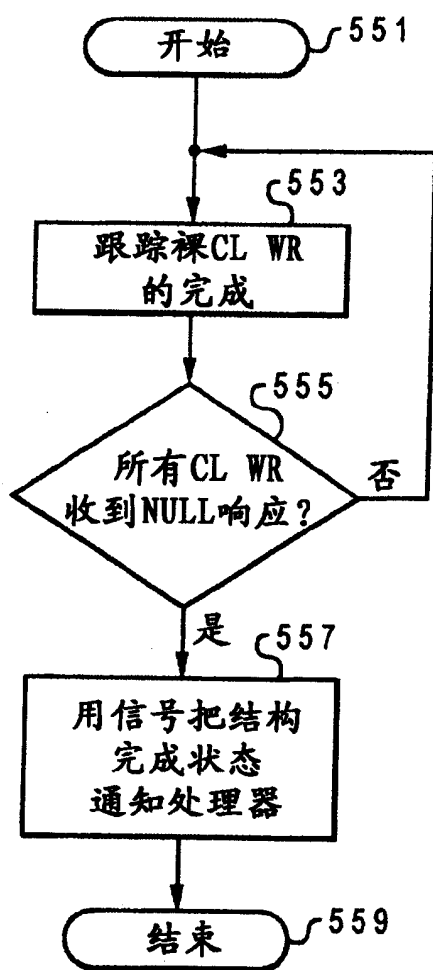


图 5C

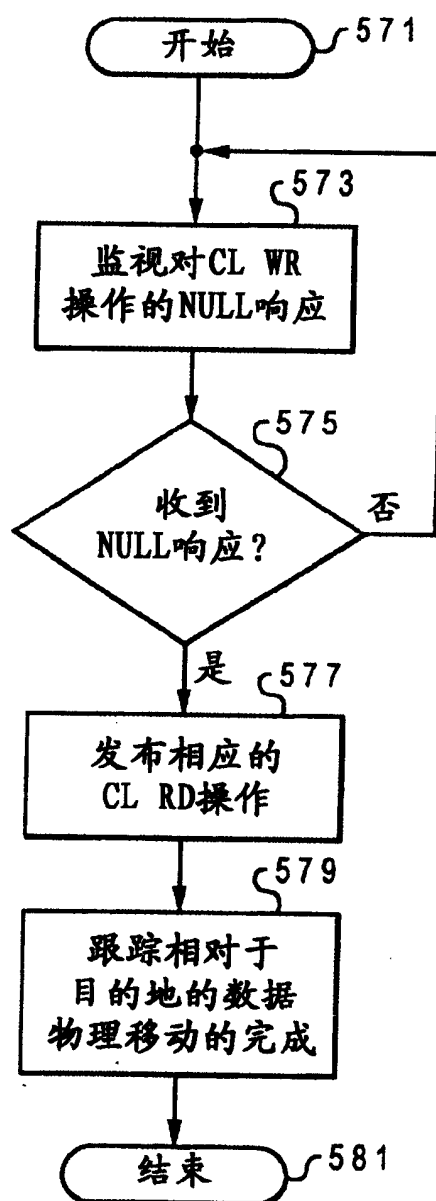
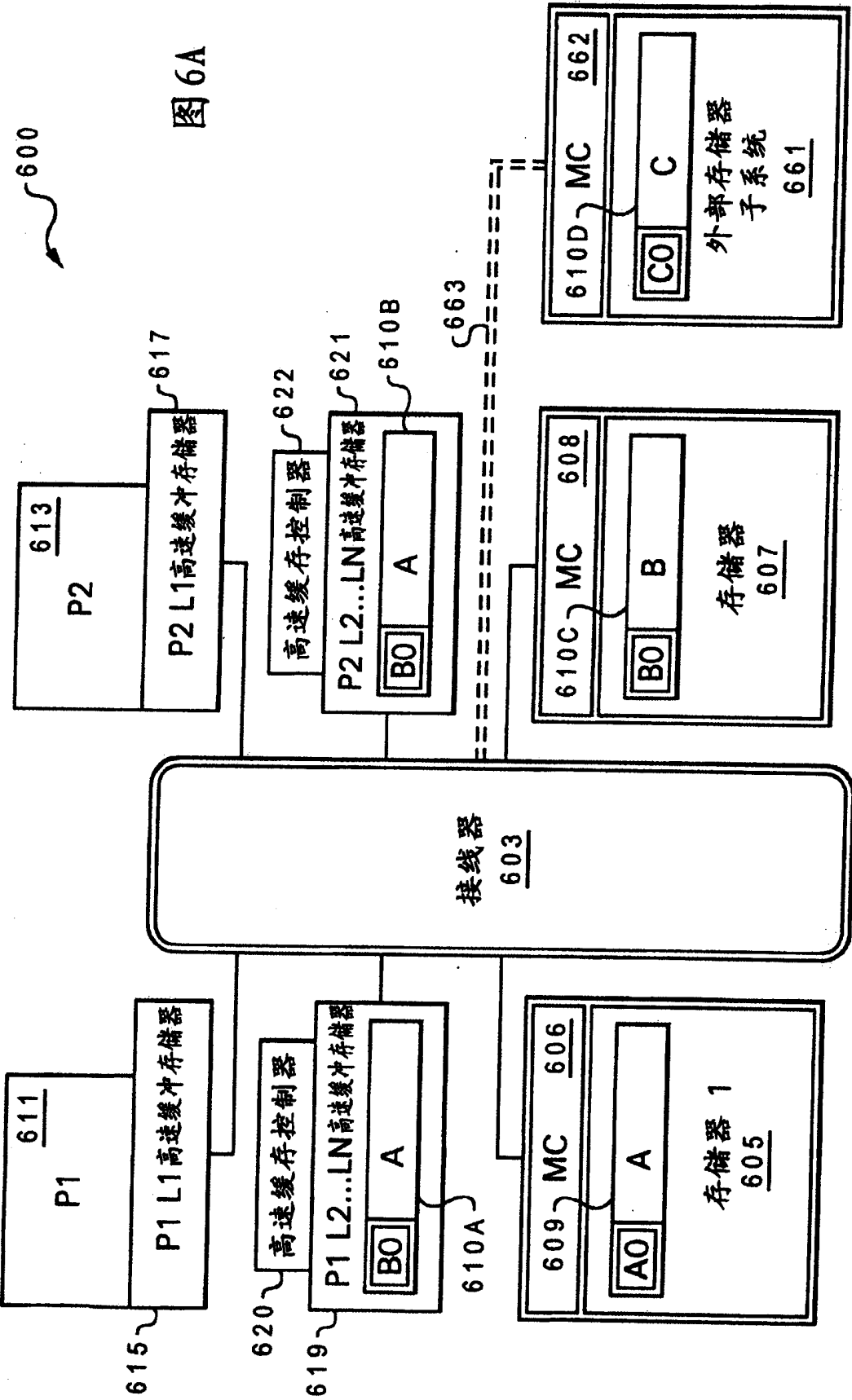


图 5D



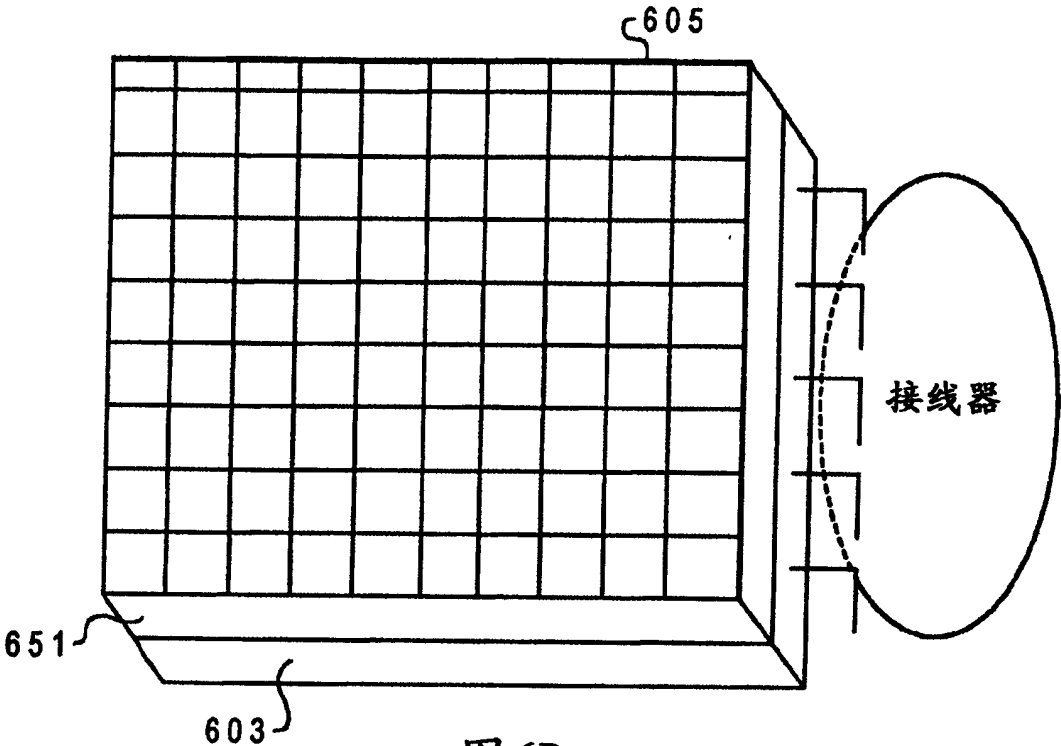


图 6B

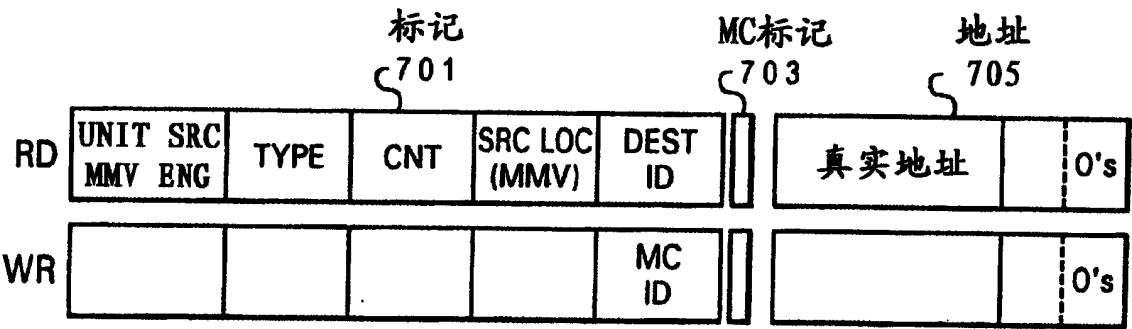


图 7A

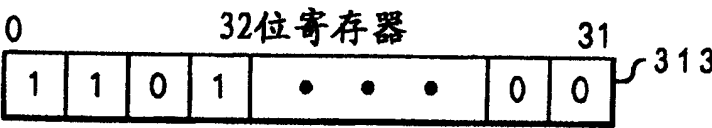


图 7B

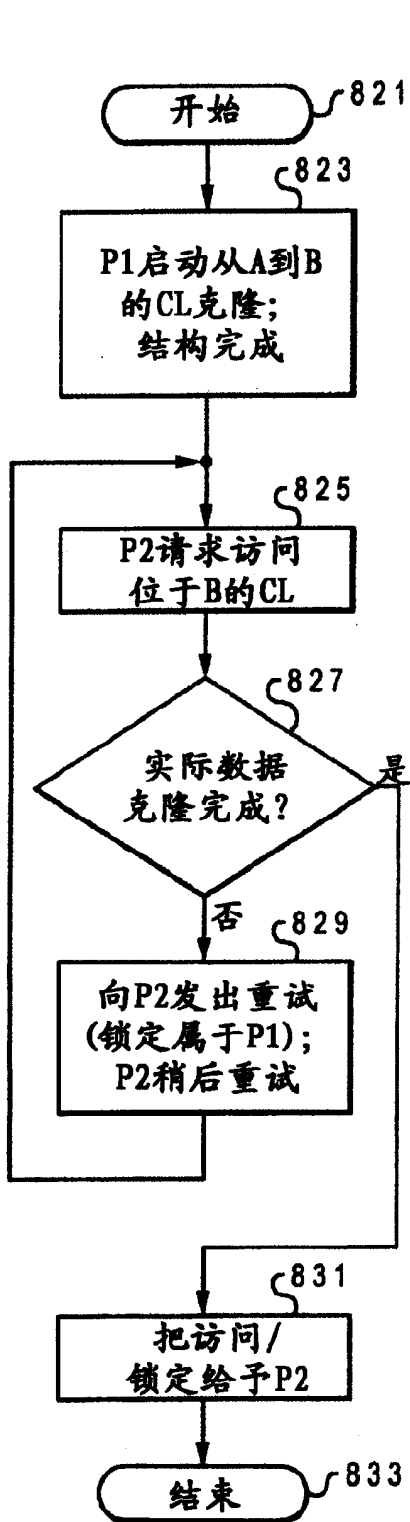


图8A

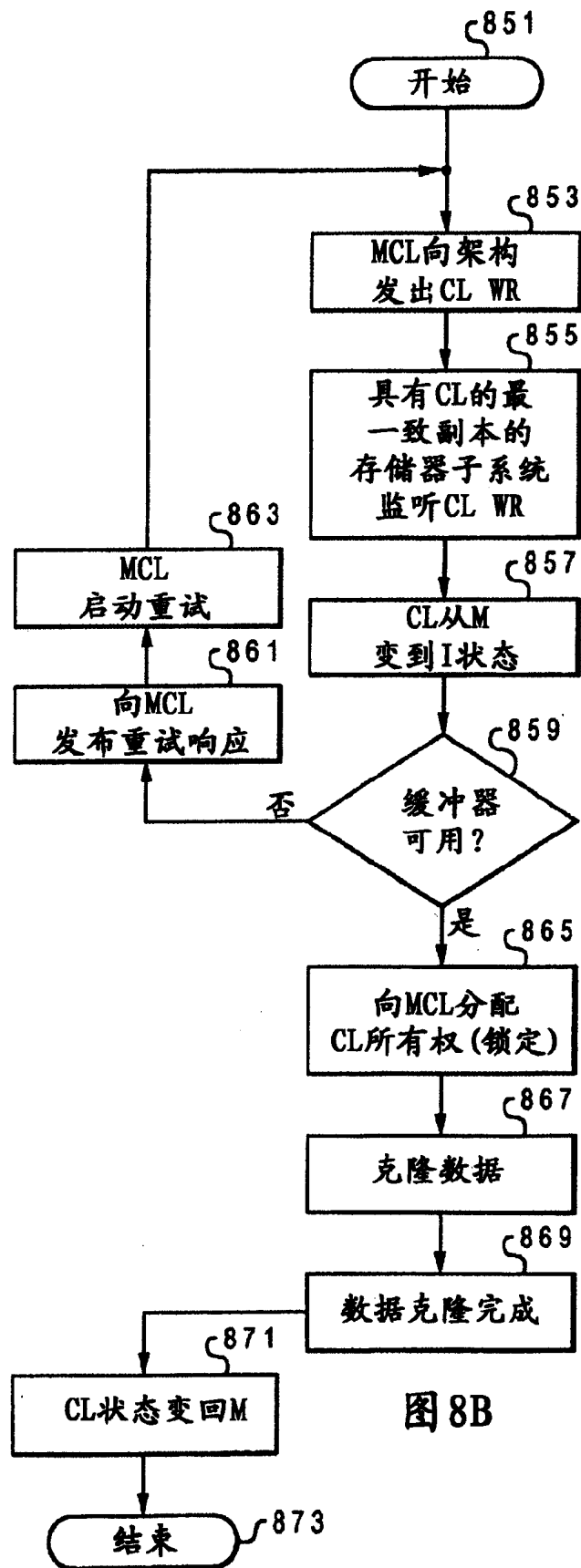


图8B

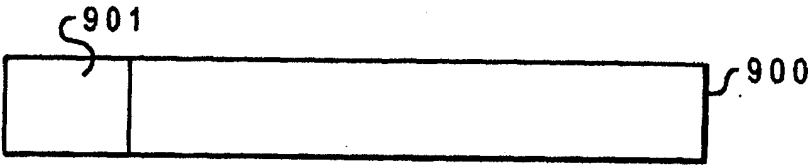


图 9A

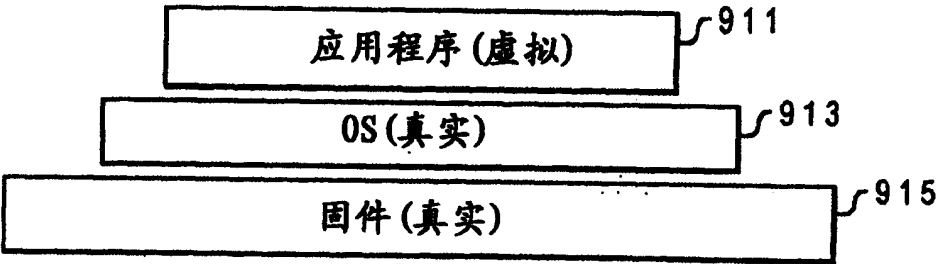


图 9B