[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 15/78 (2006.01)

[12] 发明专利说明书

专利号 ZL 200580013263.5

[45] 授权公告日 2009 年 9 月 9 日

[11] 授权公告号 CN 100538691C

[22] 申请日 2005.4.12
[21] 申请号 200580013263.5
[30] 优先权
[32] 2004. 4.26 [33] EP [31] 04101732.8
[86] 国际申请 PCT/IB2005/051196 2005.4.12
[87] 国际公布 WO2005/103934 英 2005.11.3
[85] 进入国家阶段日期 2006.10.26
[73] 专利权人 皇家飞利浦电子股份有限公司
地址 荷兰艾恩德霍芬
[72] 发明人 A·拉杜勒斯库
K·G·W·古森斯
[56] 参考文献
CN1180870A 1998.5.6
WO2004034176A2 2004.4.22
COMMUNICATION SERVICES FOR NETWORKS ON CHIP. RADULESCU A ET AL. PROCEEDINGS OF THE INTERNATIONAL WORKSHOP ON SYSTEMS, ARCHITECTURES, MODELING AND SIMULATION, Vol. 2 . 2002
审查员 张 坦
[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 顾 珊 梁 永

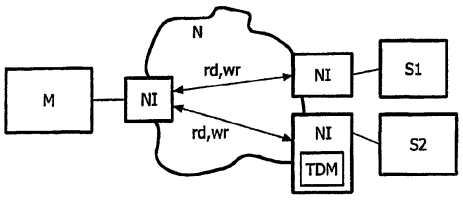
权利要求书 2 页 说明书 13 页 附图 4 页

[54] 发明名称

用于发出事务的集成电路、数据处理系统和方法

[57] 摘要

提供了一种集成电路其包括多个处理模块(M、S)和配置用于耦合所述模块(M、S)的网络(N)。所述集成电路包括：第一处理模块(M)，其用于将原子操作编码为第一事务，并且用于将所述第一事务发出到至少一个第二处理模块(S)。 此外，提供了一种事务解码装置(TDM)，其用于将所发出的第一事务解码为至少一个第二事务。



1. 一种用于处理事务的集成电路，所述集成电路包括:

- 第一处理模块（M），用于将原子操作编码为第一事务，
- 第二处理模块（S），和
- 用于耦合所述模块（M，S；IP）的网络（N），每个模块经由各自的网络接口（NI）耦合到所述网络，
- 其中，与该第二处理模块（S）相关联的网络接口（NI）包括事务解码装置（TDM），用于将所发出的第一事务解码为至少一个第二事务，并且所述第二处理模块（S）执行所述第二事务，并向所述事务解码装置（TDM）发出响应。

2. 根据权利要求 1 的集成电路，其中

所述第一处理模块（M）编码所述事务解码装置（TDM）所需的所有信息，用于管理将所述原子操作执行为所述第一事务。

3. 根据权利要求 1 的集成电路，其中

所述事务解码装置（TDM）包括请求缓冲器（REQB），用于对向第二处理模块（S）发出的请求进行排队；响应缓冲器（RESPB），用于对来自所述第二处理模块（S）的响应进行排队；和消息处理器（MP），用于检查输入的请求，并且用于驱动信号到所述第二处理模块（S）。

4. 根据权利要求 3 的集成电路，其中

所述第一事务包括报头，其具有命令，并且任选地包括标记和地址，而且所述第一事务包括零个、一个或多个值的有效负载，

其中由消息处理器（MP）开始所述命令的执行。

5. 一种用于在集成电路中发出事务的方法，所述集成电路包括多个处理模块（M；S），以及用于耦合所述模块（M；S）的网络（N），每个模块经由各自的网络接口（NI）耦合到所述网络，

所述方法包括步骤:

- 通过第一个所述处理模块（M）将原子操作编码为第一事务，
- 通过处于与第二个所述处理模块（S）相关联的网络接口（NI）内的事务解码装置（TDM）将所发出的第一事务解码为至少一个第二事务，并且
- 通过第二个所述处理模块（S）执行所述第二事务，并向所述事务解码装置（TDM）发出响应。

6. 一种数据处理系统，包括:

- 第一处理模块（M），其用于将原子操作编码为第一事务，
- 第二处理模块（S），和
- 用于耦合所述模块（M，S; IP）的网络（N），每个模块经由各自的网络接口（NI）耦合到所述网络，

其中，与该第二处理模块（S）相关联的网络接口（NI）包括事务解码装置（TDM），其用于将所发出的第一事务解码为至少一个第二事务，并且所述第二处理模块（S）执行所述第二事务，并向所述事务解码装置（TDM）发出响应。

用于发出事务的集成电路、数据处理系统和方法

技术领域

本发明涉及一种具有多个处理模块的集成电路和一种配置用于提供处理模块之间的连接的网络，一种用于在这种集成电路中发出事务的方法，和一种数据处理系统。

背景技术

由于对实现新的特征和改善现有功能的不断增长的需要，硅上系统呈现出复杂性的持续增加。这是通过增加集成电路中集成的元件的密度实现的。同时，电路操作的时钟速度也趋向于增加。较高的时钟速度以及增加的元件密度减小了可以在相同的时钟域中同步操作的面积。这产生了对模块化方法的需要。根据该方法，处理系统包括多个相对独立的复杂模块。在传统的处理系统中，系统模块通常经由总线相互通信。然而，随着模块数目的增加，出于下列原因，该通信方式不再是实用的。一方面，大量的模块形成了过高的总线负载。另一方面，由于仅能够使一个设备向总线发送数据，因此总线形成了通信瓶颈。通信网络形成了克服这些缺点的有效方法。

近来，片上网络（NoC）作为一种对高度复杂的芯片中的互连问题的解决方案，受到了极大的关注。原因是双重的。首先，由于 NoC 构造和管理全局连线，因此有助于解决新的深亚微米技术中的电气问题。同时它们共享连线，降低了它们的数目并且提高了它们的利用率。NoC 还可以是能量有效的和可靠的，并且相比于总线是可缩放的。其次，NoC 还使计算同通信分离，其在管理百万级晶体管芯片的设计中是必要的。NoC 实现了该分离，这是因为它们传统上是使用协议栈设计的，其提供了良好定义的接口，将通信服务的使用和服务的实现分开。

然而，在设计片上系统时使用用于片上通信的网络，产生了许多必须考虑的新的问题。这是因为，与其中通信模块是直接连接的现有的片上互连相反（例如，总线、开关、或点对点连线），在 NoC 中，模块经由网络节点远程通信。结果，互连仲裁从集中式变为分布式，并且必须由知识产权（IP）块或网络处理如无序块事务、较高的延时

和端到端流量控制的问题。

大部分该题目已成为局域和广域网络（计算机网络）的领域中的研究课题，并且作为用于并行机互连网络的互连。此两者与片上网络有很大关系，并且该领域中的许多结果同样适用于片上。然而，NoC的前提不同于片外网络，并且因此，必须重新评估大部分网络设计选择。片上网络具有不同的属性（例如，更紧密的链路同步）和限制（例如，更高的存储器成本），导致了不同的设计选择，其最终影响网络服务。

NoC同片外网络的主要差别在于它们的限制和同步。典型地，片上相比于片外，资源限制是更紧张的。存储（即、存储器）和计算资源相对更加昂贵，同时片上相比于片外，点对点链路的数目是更大的。由于通用的片上存储器，诸如RAM，占用了大的面积，因此存储是昂贵的。由于存储器中的开销面积变得是首要的，因此使存储器以相对小的尺寸分布在网络元件中，是更差的。

对于片上网络，相比于片外网络，计算也是相对高成本的。片外网络接口通常包含专用处理器，用于实现上至网络层或甚至更高层的协议栈，以减轻主机处理器的通信处理负担。在网络接口中包括专用处理器在片上是不可行的，这是因为网络接口的尺寸将变得与待连接到网络的IP相当或者比其更大。而且，由于这些IP仅具有一个专用功能，并且不具有运行网络协议栈的能力，因此在IP自身上运行协议栈也是不可行的。

计算机网络拓扑通常具有不规则的（可能是动态的）结构，其可能引入缓冲循环。例如，通过引入拓扑或路由中的限制，还可以避免死锁。胖树型拓扑已被考虑用于NoC，其中通过在缓冲器溢出的情况中在网络中弹回分组，避免了死锁。针对系统设计的基于区块的方法使用网状或圆环网络拓扑，其中使用例如，转弯模型路由算法，可以避免死锁。死锁主要是由缓冲器中的循环引起的。为了避免死锁，路由必须是无循环的，这是因为在实现可靠的通信方面其成本较低。死锁的第二个原因是事务的原子链。原因在于，当模块锁定时，存储事务的序列可能由原子事务链外部的事务填满，阻挡对链中事务的访问到达锁定模块。如果必须实现原子事务链（用于同允许该原子事务链的处理器兼容，诸如MIPS），则网络节点应能够过滤原子链中的事务。

相比于直接互连，诸如总线或开关，引入网络作为片上互连在根本上改变了通信。这是因为网络的多跳本质，其中通信模块不是直接连接的，而是由一个或多个网络节点隔开。这与其中模块是直接连接的现有的普遍互连（即，总线）相反。该变化的含意在于仲裁（其必须从集中式变为分布式），并且在于通信属性（例如，排序或流量控制）。

现代的片上通信协议（例如，设备事务级 DTL、开放内核协议 OCP 和 AXI-协议）基于分段和管线操作，其中事务由请求和响应组成，并且在由主设备发出的请求由从设备接受之后，由其他设备将总线释放以备使用。分段管线通信协议用于多跳互连中（例如，片上网络或者具有桥的总线），允许有效率地利用互连。

关于多跳互连的一个困难是，如何执行原子操作（例如，测试和设置、比较-交换等）。事务的原子链是由单一的主设备发起的事务序列，其排他性地在单一的从设备上执行。即，一旦链中的第一个事务要求从设备，则拒绝其他的主设备访问该从设备。典型地在多处理系统中使用原子操作，以实现较高级别的操作，诸如互斥或信号量（semaphore），因此其广泛地用于实现主模块之间的同步机制（例如，信号量）。

目前存在两种方法用于实现原子操作（为了简化这里仅描述了测试和设置操作，但是可以相似地对待其他的原子操作），即 a)锁定或 b)标记。通过锁定互连，用于由请求原子链的主设备排他性地使用，可以实现原子操作。使用锁定，即，主设备锁定资源直至原子事务结束，事务总是成功，然而这在开始时可能耗费时间，并且将影响其他方面。换言之，互连、从设备或者部分地址空间由主设备锁定，其意味着在锁定时没有其他的主设备能够访问锁定的实体。因此容易地实现了原子态，但是带来了性能损失，特别是在多跳互连的情况中。时间资源的锁定是较短的，这是因为，一旦批准主设备访问总线，则其可以快速地执行链中的所有事务，并且不需要用于链中的后继事务的仲裁延迟。因此，锁定的从设备和互连可以在短的时间内再次开放。

此外，可以通过下列方法实现原子操作，通过设置标记限制对锁定从设备的访问的批准，即，主设备将资源标记为正在使用，并且如果当原子事务完成时，该标记仍这样设置，则原子事务成功，否则失

败。在该情况中，较快速地执行原子事务，不会影响其他方面，但是存在失败的可能。这里，对于排他性访问的情况，原子操作限制于一对两个事务：ReadLinked 和 WriteConditional。在 ReadLinked 之后，将标记（初始重置）设置为从设备或地址范围（还被称为从区域）。后面，尝试 WriteConditional，当该标记仍这样设置时其成功。当在由该标记标出的从设备或从范围中执行其他的写操作时，重置该标记。互连未被锁定，并且仍可由其他的模块使用，然而，这是以从设备的较长的锁定时间为代价的。

其次，所锁定/标记的是：整个互连、从设备（或一组从设备）、或者存储器区域（在从设备中，或者跨越数个从设备）。

通常，这些原子操作由两个事务组成，必须在没有来自其他事务的任何干扰的情况下顺序执行这两个事务。例如，在测试和设置操作中，首先执行读事务，将读取值同零（或者其他的预定的值）比较，并且在成功之后，通过写事务将另一值写回。为了获得原子操作，在读和写事务之间，不应在相同的位置上允许写事务。

在这些情况中，主设备（例如，CPU）必须在关于该原子操作的互连上执行两个或多个事务（即，Locked Read 和 Locked Write、以及 ReadLinked 和 WriteConditional）。对于多跳互连，其中事务的延时是相对高的，原子操作引入了不需要的长的等待时间。

由多跳互连中的高的延时引起的其他的问题是这两种实现方案所特有的。对于锁定，由于具有分布式仲裁，因此锁定完整的多跳互连是不可行的，并且锁定将耗费过多的时间，并且涉及仲裁器之间的过多的通信。因此，在 AXI 和 OCP 协议中，锁定从设备或从区域而非互连。然而，即使在该情况中，锁定的从设备或从区域将禁止来自锁定主设备以外的所有主设备的访问。因此，来自其他的主设备的针对该从设备的所有业务在互连中积累，并且将引起网络拥塞，由于还影响了未以锁定从设备或从区域为目标的业务，因此其是不理想的。

对于排他性访问，随着延时的增加（典型地在多跳互连中），并且随着试图访问相同的从设备或从区域的主设备的数目的增加，WriteConditional 成功的可能性降低。

对于这两种方案，限制对其他业务的影响的一种解决方案是，使从区域的尺寸尽可能小。在该情况中，使被影响的（对于锁定）或者

影响（对于排他性访问）原子操作的关联业务减少。然而，具有大量的锁定/标记的实现成本或者用于实现该方案的动态可编程表格的实现复杂度是过高的。

发明内容

因此，本发明的目的在于，提供一种集成电路，其具有改善的处理原子事务链的能力。

因此，集成电路包括第一处理模块，其用于将原子操作编码为第一事务，以及至少一个第二处理模块。一个网络耦合上述模块，每个模块经由各自的网络接口耦合到网络。与该第二处理模块相关联的网络接口包括一种事务解码装置，其用于将所发出的第一事务解码为至少一个第二事务。

在该集成电路中，减少了互连上的负载，即，在互连上存在较少的消息。因此，将减少用于支持原子操作的成本。

根据本发明的一个方面，所述处理模块编码所述事务解码装置所需的所有信息，用于管理针对所述第一事务的所述原子操作执行。因此，将所需的所有信息传递到事务解码装置，其可以在其自身上执行进一步的处理步骤，同时不会同第一处理模块交互。

根据本发明的另一方面，所述第一事务在所述网络上从所述第一处理模块传输到所述事务解码装置。因此，执行时间是较短的，并且因此实现了较短的主设备和连接的锁定，这是因为，在第二处理模块侧，即在从设备侧，而非在第一处理模块侧，即在主设备侧，执行原子事务。

根据本发明的优选方面，所述事务解码装置包括：请求缓冲器，其用于对关于第二处理模块的请求进行排队；响应缓冲器，其用于对来自所述第二处理模块的响应进行排队；和消息处理器，其用于检查进入的请求，并且用于向所述第二处理模块发出信号。

根据本发明的另一方面，所述第一事务包括报头，其具有命令，并且任选地包括命令标记和地址；以及有效负载，其包括零个、一个或多个值，其中由消息处理器开始所述命令的执行。在简单的 P 和 V 的情况中，存在 0 个值。扩展 P 和 V 操作具有 1 个值，TestAndSet 具有 2 个值。

本发明还涉及一种用于在集成电路中发出事务的方法，该集成电路包括多个处理模块，以及用于连接模块的网络。每个模块经由各自的网络接口耦合到网络。第一处理模块将原子操作编码为第一事务。所发出的第一事务由处于与第二处理模块相关联的网络接口内的事务解码装置解码为至少一个第二事务。

本发明还涉及一种数据处理系统，其包括多个处理模块和配置用于耦合所述模块的网络。每个模块经由各自的网络接口耦合到网络。第一处理模块将原子操作编码为第一事务。此外，作为与第二处理模块相关联的网络接口的一部分的事务解码装置将所发出的第一事务解码为至少一个第二事务。

本发明基于这样的思想，即通过将原子操作完整地编码为单一的事务并且通过将该事务的执行转移到从设备，即接收侧，将锁定资源的时间或者将使用排他性访问标记资源的时间减少到最小。

在附属权利要求中描述了本发明的其他方面。

附图说明

图 1 示出了根据第一实施例的片上系统的示意性表述；

图 2A 和 2B 示出了根据第一实施例的用于实现原子操作的方案；

图 3A 和 3B 示出了根据第二实施例的用于实现原子操作的方案；

图 4 示出了根据优选实施例的消息结构；

图 5 示出了目标模块的接收侧及其相关联的网络接口的示意性表述；

图 6 示出了目标模块的可替换的接收侧及其相关联的网络接口的示意性表述。

具体实施方式

下面的实施例涉及片上系统，即同一芯片上的多个模块经由某种类型的互连相互通信。该互连体现为片上网络 NoC，其可以在单一的芯片或在多个芯片上延伸。该片上网络可以包括连线、总线、时分多工、开关、和/或网络中的路由器。在所述网络的运输层处，通过连接执行模块之间的通信。连接被视为第一模块和至少一个第二模块之间的一组信道，每个信道具有一组连接属性。对于第一模块和单一的第二模

块之间的连接，该连接包括两个信道，即从第一模块到第二模块的信道，也就是请求信道，和从第二模块到第一模块的第二信道，也就是响应信道。请求信道被保留用于从第一模块到第二模块的数据和消息，而响应信道被保留用于从第二模块到第一模块的数据和消息。然而，如果连接涉及一个第一模块和 N 个第二模块，则提供 2*N 个信道。连接属性可以包括排序（有序的数据运输）、流量控制（远程缓冲器被保留用于连接，并且允许数据产生者仅在确保空间对于产生的数据可用时发送数据）、吞吐量（确保吞吐量的下限）、延时（确保延时的上限）、损耗（数据丢失）、传输终止、事务完成、数据正确性、优先级、或者数据递送。

图 1 示出了根据本发明的片上系统。该系统包括主模块 M、两个从模块 S1、S2。每个模块分别经由网络接口 NI 连接到网络 N。网络接口 NI 用作主和从模块 M、S1、S2 同网络 N 之间的接口。网络接口 NI 被提供用于管理各个模块和网络 N 之间的通信，由此模块可以执行它们的专用操作，而不必处理同网络或其他模块的通信。网络接口 NI 可以通过网络 N 相互之间发送诸如读 rd 和写 wr 的请求。

上文所述的模块可以是所谓的知识产权块 IP（计算元件、存储器或者子系统，其可以内部地包含互连模块），其在所述网络接口 NI 处同网络交互。

特别地，将事务解码装置 TDM 配置在与从设备 S1、S2 其中一个相关联的至少一个网络接口 NI 中。原子操作被实现为通信协议中将要包括的特殊事务。这种思路是，将资源锁定或者使用排他性访问进行标记的时间减少到最小。为了实现这一点，通过主侧将原子操作完整地编码为单一的事务，并且将其执行转移到从侧。

图 2A 和 2B 中说明了其实现方案。图 2A 中示出了使用锁定的传统的原子操作，而图 2B 中示出了根据第一实施例的原子操作。

因此，图 2A 示出了片上网络环境中的第一和第二主设备 M1、M2 与从设备 S 之间的通信方案的基本表述。第一主设备 M1 请求“读和锁定”操作，即读取从设备 S 中的值并且锁定从设备 S，并且从设备 S 返回响应“读和锁定”，可能返回读取值。然后，将从设备 S 锁定（L1）到主设备 M1，由此阻挡来自第二主设备 M2 的请求“写 2”，即，其执行被延迟。在主设备 M1 自从设备 S 接收到响应“读和锁定”之后，其

向从设备S发出请求“写1”，以便于将值写入到从设备S中。来自主设备M1的该第二请求由从设备S接收，并且在操作结束时，将响应“写1”传递到主设备M1并释放从设备S的锁定（L2）。因此，从设备S从L1到L2是锁定的，并且阻挡请求“写2”直至L2，即从设备S的释放。现在从设备S可以继续处理来自第二主设备M2的请求“写2”。

在图2B中示出了根据第一实施例的片上网络环境中的第一和第二主设备M1、M2与从设备S之间的通信方案的基本表述。主设备M1请求“测试和设置”操作。通过主设备1将用于在从设备侧处理该请求的所有信息包括在单一的原子事务中。该单一的原子事务“测试和设置”由与从设备相关联的事务解码装置TDM接收。由原子事务解码装置TDM发出事务的执行，从设备执行请求的操作，并且在该事务被执行之后，从设备发出响应“测试和设置”。在L10处接收到第一请求时将从设备锁定到主设备M1，并且在L20处终止事务执行且发出响应“测试和设置”时，释放从设备。因此，来自第二主设备M2的请求“写”被阻挡直至在L20处释放从设备。

换言之，仅在从设备处的原子操作的执行过程中阻挡从设备，该过程比如图2A所示的执行短得多。而且，由于不需要在主设备自身中实现原子操作，因此主设备更简单。在主设备上存在较少的负担（其不需要执行部分原子操作）。然而，复杂性被转移到可重复使用的互连，特别是网络接口。

在比较如图2A和图2B所示的通信方案时，可以观察到，根据图2A的传统的实现方案中的锁定时间（L1-L2）较长，这是因为主设备M1参与原子操作的执行，即请求“读、锁定”和请求“写1”。因此，对于两倍网络延时加上主设备M1执行其部分原子操作的时间，从设备S被锁定。在全部该时间中，阻挡了以从设备S为目标的业务量（例如，来自主设备M2）。

图3A和3B示出了根据作为优选实施例的第二实施例用于实现原子操作的方案。图3A中示出了使用锁定的传统的原子操作，而在图3B中示出了根据第二实施例的原子操作。

在图3A中，特别地，如图1所示的主设备M和从设备S之间的通信连同主设备M的中间网络接口MNI和从设备S的中间网络接口SNI。特别地，针对两种示例执行描述了基础原理，即作为第一执行示例ex1

的 LockedRead 和作为第二执行示例 ex2 的 ReadLinked。

主设备 M 发出第一事务 t1，其可以是作为执行 ex1 的 LockedRead 和作为执行 ex2 的 ReadLinked。事务 t1 被转发到主设备 M 的网络接口 MNI，经由网络 N 传递到从设备的网络接口 SNI 并且最终到达从设备 S。从设备 S 执行事务 t1，并且可能经由网络接口 SNI 和与主设备相关联的网络接口 MNI 向主设备返回某些数据。同时，阻挡从设备 S 执行 LockedRead 或 ReadLinked，并且将其分别标记为执行 Write 或 WriteConditional。当主设备 M 接收到从设备 S 的响应时，其执行第二事务 t2，这在上文提及的执行 ex1 和 ex2 的两种情况中是比较。随后，主设备 M 向从设备发出第三事务 t3，分别地，在执行 ex1 的情况中其是 Write 命令，而在执行 ex2 的情况中其是 WriteConditional 命令。从设备 S 接收该命令，并且返回对应的响应。随后，释放从设备 S。

在图 3B 中，示出了根据第二实施例的片上网络环境中的主设备 M 和从设备 S 之间的通信方案的基本表述。基础片上网络环境的基本结构对应于如图 3A 中描述的环境，然而，在片上网络环境中额外地包括事务解码装置 TDM。主设备 M 发出原子事务 ta，如 TestAndSet，其经由主设备 M 的网络接口 MNI 被转发到事务解码装置 TDM。

如根据图 3A 所描述的，描述了关于 TestAndSet 命令的原子事务 ta 的实现或解码的两个不同的执行示例，即作为第一执行示例 ex1 的 LockedRead 和 Write 以及作为第二执行示例 ex2 的 ReadLinked 和 WriteConditional。

这里，主设备 M 发出原子事务 ta。现在由事务解码装置 TDM 执行由主设备 M 执行的如根据图 3A 所描述的原子事务 ta 的解码以及第一、第二和第三事务 t1、t2、t3 的处理。因此，事务解码装置 TDM 将原子事务 ta 解码为事务 t1，即解码为第一或第二执行示例 ex1 或 ex2。因此，一旦从设备 S 经由与该从设备相关联的网络接口 SNI 从事务解码装置 TDM 中接收到第一事务 t1，即 ex1 或 ex2，则执行第一事务 t1，并且该从设备向事务解码装置 TDM 发出响应，其可能包含某些数据。事务解码装置 TDM 根据第二事务 t2，即根据第一或第二执行示例 ex1 或 ex2 执行比较，其中其是关于这两种情况的比较。随后，事务解码装置 TDM 向从设备 S 发出作为 ex1 的 Write 或作为 ex2 的 WriteConditional，其执行第三事务，并且在 LockedRead 和 Write，

即第一执行示例 ex1，以及 ReadLinked 和 WriteConditional，即第二执行示例 ex2 的情况中，将从设备解锁，如果标记仍被设置则其成功。向主设备 M 发出对应的响应。

如图 3B 所示，存在较少的需经网络转发的事务。此外，主设备 M 仅需处理一个原子事务而具有较低的处理负担，同时该原子事务在事务解码装置 TDM 处被扩展为多个较简单的事务。根据第二实施例的主设备 M 需了解该原子事务，某些处理步骤现在由事务解码装置 TDM 执行而非由主设备 M 执行。例如，由事务解码装置 TDM 执行第一和第二事务 t1 和 t3 之间的比较 t2。

可替换地，从设备还可以了解原子事务，但是在该情况中，事务解码装置 TDM 可以是从设备 S 的一部分。这将导致简化的网络，这是因为事务解码装置 TDM 从网络中转移并且配置在从设备 S 中。因此，此外，在与从设备相关联的网络接口 SNI 和从设备自身之间将传递较少的事务。特别地，这可以仅为原子事务。

原子事务的示例可以是测试和设置，以及比较和交换。在这两种情况中，必须由事务请求承载两个数据值：待比较的值（CMPVAL）和待写入的值（WRVAL）。在这两个示例中，CMPVAL 同事务地址处的值比较。如果它们是相同的，则写入 WRVAL。来自从设备的响应对于测试和设置在该位置处是新的值，以及对于比较和交换是旧的值。应当注意，任何布尔函数可以替换该简单比较（例如，小于或等于，如下文所述的信号量扩展中使用的）。

信号量事务是更先进的，并且从事务的观点来看是更简单的，其将调用不使用任何参数的 P 和 V。P 等待直至其访问事务中指明的地址，然后尝试使由事务地址指明的位置处的值递减。如果该值是正的，则使其递减，并且返回成功。如果该值是零或正的，则其不变并返回失败。V 总是成功并且递增所所指明的地址处的位置。

P 和 V 事务的扩展是可行的，其中待递增/递减的值（VAL）被规定为 P/V 事务的数据参数。如果事务地址处的值大于或等于 VAL，则 P 使事务地址处的位置递减 VAL，并且返回成功。否则使该位置不变并且返回失败。V 总是成功的并且使寻址位置递增 VAL。

本发明涉及将操作编码为事务，其在从设备侧的互连中实现和执行。

测试和设置事务在IC设计中与高延时互连(例如,具有桥的总线、片上网络)尤其相关,随着芯片复杂度的增加其将变成固有的。

上文提及的测试和设置事务的优点包括不需要锁定互连。在互连上存在较少的负载(即,较少的消息)。主设备处的测试和设置操作的执行时间较短。CPU/主设备仅需要执行单一的指令,而代替了执行关于测试和设置操作的三个指令(读、比较、写)。而且,减少了用于支持原子操作的成本。然而,缺点在于目前的CPU仍不提供该指令。

图 4 示出了根据第一实施例的消息结构。这里,请求消息由报头 hd 和有效负载 pl 组成。报头 hd 由命令 cmd(例如,读、写、测试和设置)、标记(例如,有效负载尺寸、比特掩码、缓冲)和地址组成。有效负载 pl 可以是空的(例如,对于读命令),可以包含一个值 v1(例如,写命令)或者两个值 V1、V2(例如,测试和设置命令)。

图 5 示出了接收侧,即从设备 S 及其相关联的网络接口 NI。从设备的网络接口以及特别是事务解码装置 TDM,实现了测试和设置操作。仅示出了同测试和设置操作的实现相关的这些部分的网络接口,即事务解码装置 TDM。

从设备网络接口中的事务解码装置 TDM 包含两个消息队列,即请求缓冲器 REQB 和响应缓冲器 RESB、消息处理器 MP、比较器 CMP、比较器缓冲器 CMPB 和选择器 SEL。事务解码装置 TDM 包括连接到请求缓冲器 REQB 的请求输入、连接到响应缓冲器 RESB 的输出的响应输出、关于待写入到从设备中的数据 wr_data 的输出、关于自从设备输出的数据 rd_data 的输入、关于从设备 S 中的地址“地址”的控制输出、用于选择读/写 wr/rd 的输出、以及关于有效写 wr_valid 的输出、关于读接受 rd_accept 的输出、关于写接受 wr_accept 和关于有效读 rd_valid 的输入。消息处理器 MP 包括下列输入:请求缓冲器 REQB 的输出、写接受输入 wr_accept、读有效输入 rd_valid 和比较器 CMP 的结果输出 res。消息处理器包括下列输出:地址输出、写/读选择输出 wr/rd、写有效输出 wr_valid、读接受输出 rd_accept、关于选择器的选择信号 SEL、写使能信号 wr_en、读使能信号 rd_en、关于比较器的读使能信号 cren、和关于比较器的写使能信号 cwen。

请求缓冲器或队列 REQB 容纳经由网络接收自主设备的并且将在从设备处递送的请求(例如,读、写、测试和设置命令及其标记、地址

和（有可能）数据）。响应缓冲器或队列 RESB 容纳由从设备 S 针对主设备 M 产生的消息，作为对命令的响应（例如，读数据、应答）。

而且，消息处理器 MP 检查每个消息报头 hd，其是针对请求缓冲器 REQB 的输入。依赖于报头 hd 中的命令 cmd 和标记，驱动信号到从设备。在写命令的情况中，将 wr/rd 信号设置为写，并且通过设置 wr_valid 在 wr_data 输出上提供数据。对于读命令，将 wr/rd 设置为读，并且将选择器 SEL 设置为使读数据 rd-data 通过。当读数据出现在输入 rd-data 上时（即 rd_valid 是高的），设置 rd_en（即准备好接受），并且当响应队列接受数据时（为了简化未示出信号），生成 rd_accept。选择器 SEL 响应消息处理器 MP 的选择器信号 SEL，将请求缓冲器 REQB 的输出或 rd_data 输出传递到响应缓冲器 RESB 或比较器缓冲器 CMPB。

对于测试和设置命令，消息处理器 MP 首先向从设备发出读命令，并且将接收的数据存储在比较器缓冲器或队列 CMPB 中。然后，消息处理器 MP 激活请求缓冲器 REQB 和比较器缓冲器 CMPB，以产生尺寸=N 个字的通过比较器 CMP 的数据。如果每对字具有相同的字，则比较测试成功，并且请求缓冲器或队列 REQB（尺寸也为 N 个字）中的下一个值被写入到从设备 S。在该情况中，还经由响应队列 REQB 将所写的值直接返回到主设备 M。如果测试失败，则放弃请求队列中的第二个值（即不写入到从设备），并且将第二个读命令发出到经由响应队列 REQB 返回到主设备的相同的地址。

图 6 示出了如图 5 中所示的接收侧的可替换的配置方案的示意性表述。图 6 的配置方案的操作基本上对应于图 5 的配置方案的操作。图 6 的配置方案对应于图 5 的配置方案，但是图 5 的消息处理器 MP 分为两部分，即分为消息处理器 MP 和消息处理器 MP 同从设备 S 之间的协议外壳 PS。这里，对应于事务解码装置 TDM 的部分，即消息处理器 MP、比较器 CMP、比较器队列 CNPB 和选择器 sel，由虚线围绕。请求队列 REQB 和响应队列 RESPQ 可以是网络 N 的一部分。

协议外壳 PS 用于将消息处理器 MP 的消息翻译为从设备 S 可以通信的协议，即总线协议。特别地，消息或信号事务请求 t_req、事务请求有效 t_req_valid 和事务请求接受 t_req_accept 以及信号事务响应 t_resp、事务响应有效 t_resp_valid 和事务响应接受 t_resp_accept，被翻译为从设备 S 的各个输出和输入信号，如根据图 5 所描述的。

可替换地，事务解码装置TDM和协议外壳PS可以在与从设备S相关联的网络接口NI中实现，或者被实现为网络N的一部分。

上文所述的片上网络可以在单一的芯片或多个芯片的环境中实现。

应当注意，上文提及的实施例说明而非限制本发明，并且在不偏离附属权利要求的范围的前提下，本领域的技术人员将能够设计许多可替换的实施例。在权利要求中，任何置于括号之间的参考符号不应被解释为限制权利要求。词“包括”不排除权利要求中列出的元素或步骤以外的元素或步骤的存在。元素之前的词“一个”不排除多个该元素的存在。在列举数个装置的设备权利要求中，数个该装置可由一个相同的硬件项物化。在互不相同的独立权利要求中陈述了特定的措施这一事实，并非表明这些措施的组合不是有利的。

而且，权利要求中的任何参考符号不应被解释为限制该权利要求的范围。

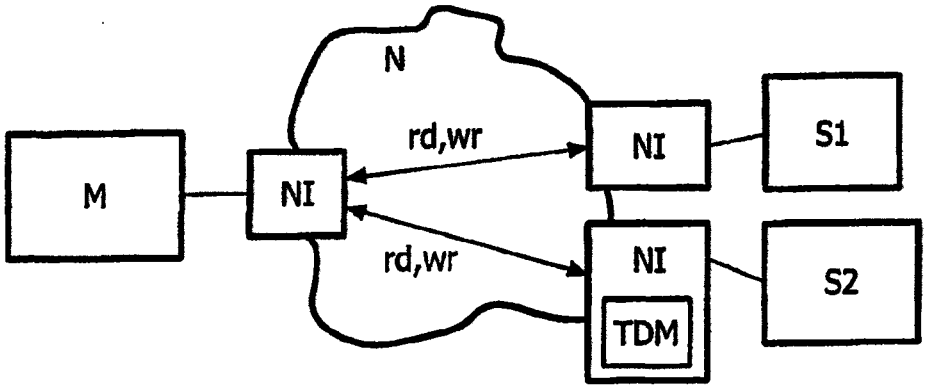

N
rd,wr
NI
S1
M
NI
rd,wr
NI
S2
TDM

图 1

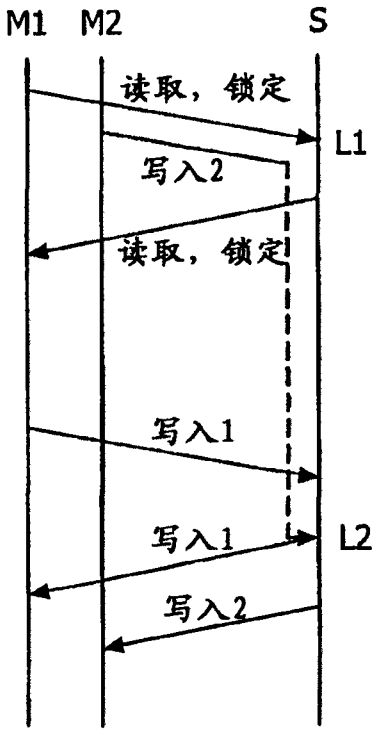

M1
M2
S
读取，锁定
L1
写入2
读取，锁定
写入1
写入1
L2
写入2

图 2A

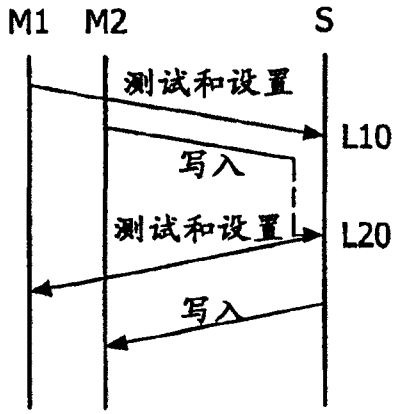

M1
M2
S
测试和设置
L10
写入
测试和设置
L20
写入

图 2B

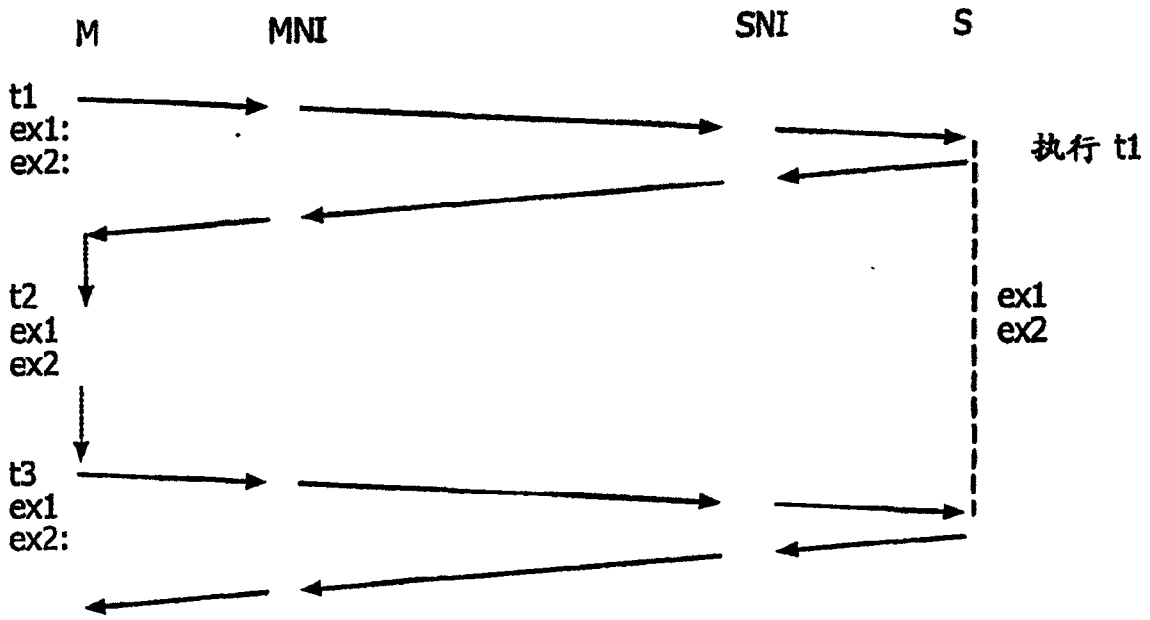

M
MNI
SNI
S
t1
ex1:
ex2:
执行 t1
t2
ex1
ex2
ex1
ex2
t3
ex1
ex2:

图 3A

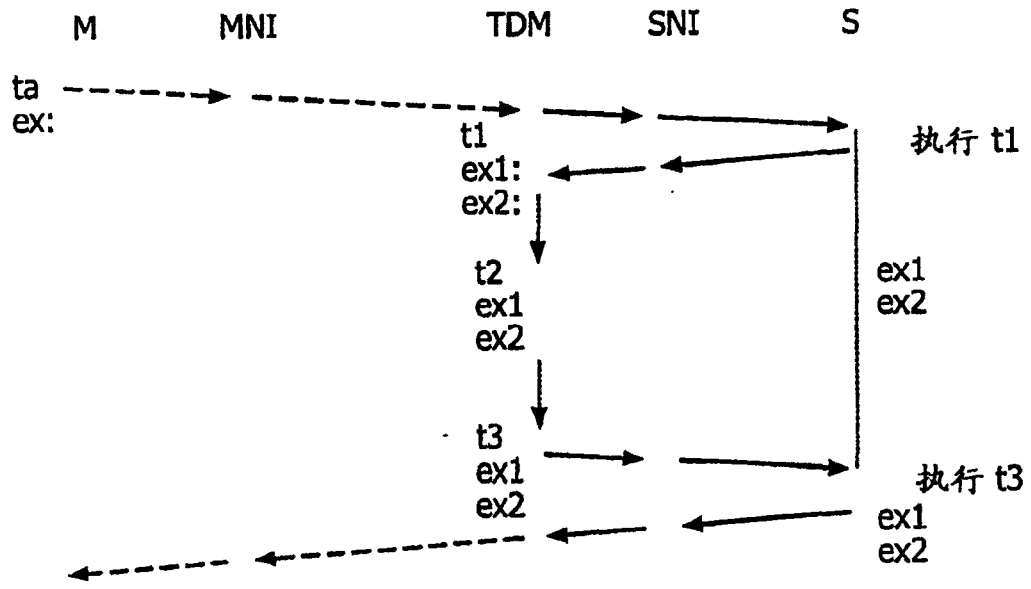

M
MNI
TDM
SNI
S
ta
ex:
t1
ex1:
ex2:
执行 t1
t2
ex1
ex2
ex1
ex2
t3
ex1
ex2
执行 t3
ex1
ex2

图 3B

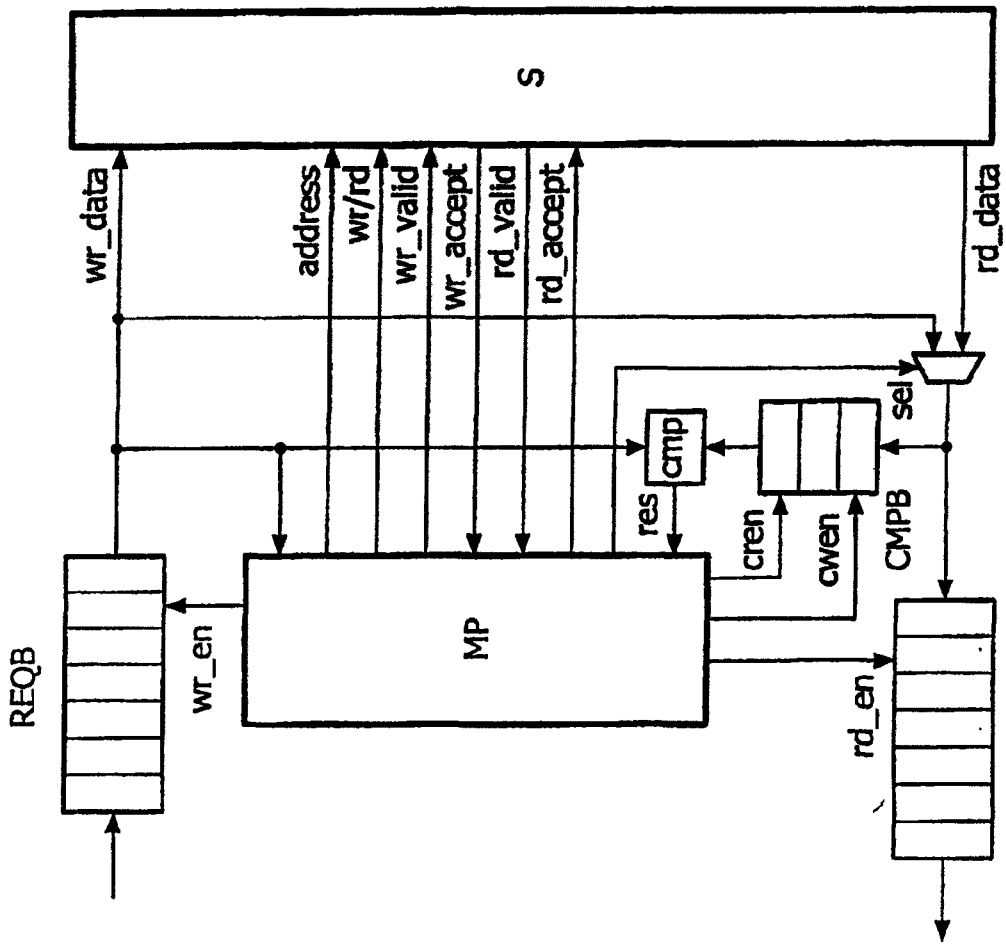

S
wr_data
address
wr/rd
wr_valid
wr_accept
rd_valid
rd_accept
rd_data
sel
cmp
res
cren
cwen
CMPB
REQB
wr_en
MP
rd_en
RESPB

图 5

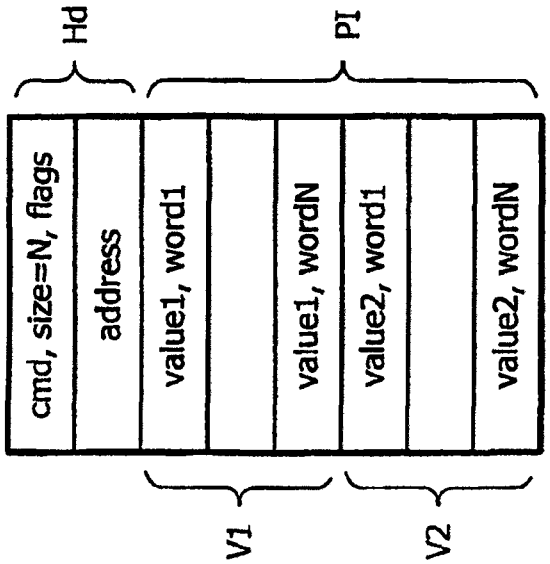

Hd
PI
cmd, size=N, flags
address
value1, word1
value1, wordN
value2, word1
value2, wordN
V1
V2

图 4

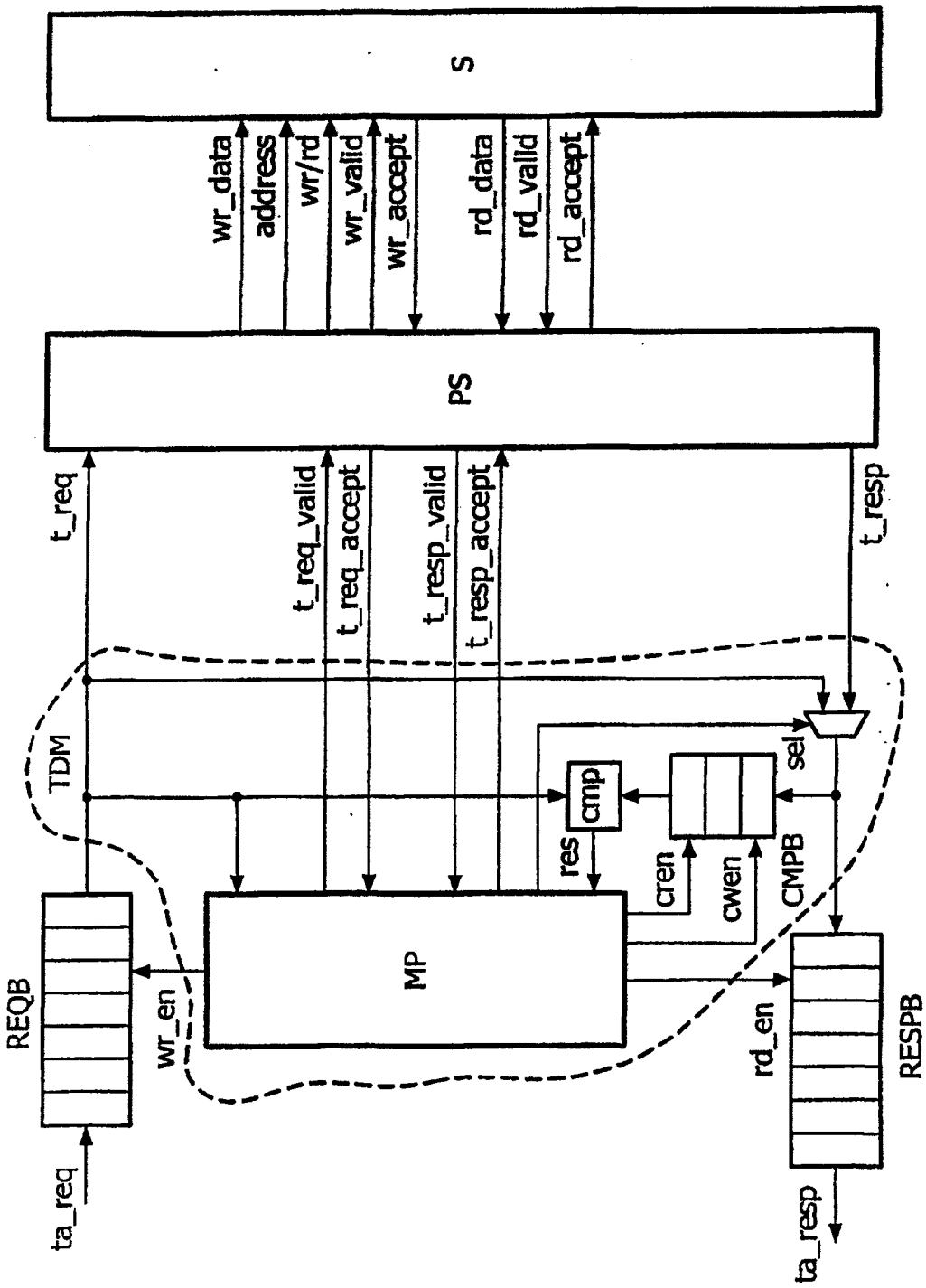

S
wr_data
address
wr/rd
wr_valid
wr_accept
rd_data
rd_valid
rd_accept
PS
t_req
t_req_valid
t_req_accept
t_resp_valid
t_resp_accept
t_resp
TDM
sel
cmp
res
cren
cwen
CMPB
MP
REQB
wr_en
rd_en
RESPB
ta_req
ta_resp

图 6