(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 107111488 B
(45)授权公告日 2021.06.15

(21)申请号 201580069226.X
(22)申请日 2015.11.23
(65)同一申请的已公布的文献号
申请公布号 CN 107111488 A
(43)申请公布日 2017.08.29
(30)优先权数据
1423041.1 2014.12.23 GB (续)
(85)PCT国际申请进入国家阶段日
2017.06.16
(86)PCT国际申请的申请数据
PCT/GB2015/053559 2015.11.23
(87)PCT国际申请的公布数据
WO2016/102919 EN 2016.06.30
(73)专利权人 ARM 有限公司
地址 英国剑桥
(72)发明人 斯蒂芬·迪斯特尔霍斯特
迈克尔·约翰·威廉姆斯 (续)
(74)专利代理机构 北京东方亿思知识产权代理有限责任公司 11258
代理人 桑敏
(51)Int.Cl.
G06F 9/30(2006.01) (续)
(56)对比文件
US 2012179877 A1,2012.07.12
US 2012179877 A1,2012.07.12
Torvald Riegel 等.Optimizing Hybrid Transactional Memory:The Importance of Nonspeculative Operations.《23rd ACM Symposium on Parallelism in Algorithms and Architectures, SPAA″11》.2011, (续)
审查员 宁雪莹

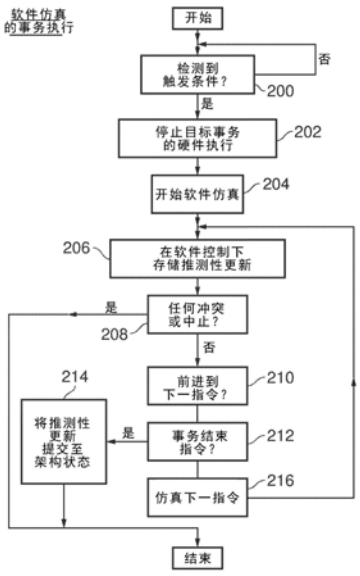
权利要求书3页 说明书7页 附图8页

(54)发明名称

调试数据处理事务

(57)摘要

提供了一种数据处理系统,该系统支持执行包括执行以产生推测更新的一个或多个程序指令的事务。如果事务完成且没有冲突,则在常规操作中推测性更新被提交。可以检测事务执行的开始,并且执行将被转变为由软件仿真执行,而不是由处理硬件直接执行。软件仿真可以存储表示在仿真期间生成的推测性更新的一个或多个版本的数据。软件仿真还可以检测与被仿真的事务的冲突。为了便于修改与被调查的事务相互作用的系统的其他部分的行为,可以响应于检测到对被仿真的事务的存储器访问请求而返回非标准响应信号。可以使用非标准响应信号来发动请求源遵循不同于其本要遵循的处理路径的后续处理路径。还提供了支持以检测导致部分完成的事务停止(冻结)的触发条件,并且将与部分完成的事务相关联的推测性更新保存到系统的架构状态。



CN 107111488 B

[转续页]

[接上页]

(30)优先权数据

1423042.9 2014.12.23 GB

1500183.7 2015.01.07 GB

(72)发明人 理查德·罗伊·格里森怀特

马修·詹姆斯·霍斯内尔

(51)Int.Cl.

G06F 9/38(2006.01)

G06F 9/455(2006.01)

G06F 11/36(2006.01)

(56)对比文件

Sean Lie等.Hardware Support for Unbounded Transactional Memory.《Massachusetts Institute of Technology 2004》.2004,

1.一种处理数据的方法,包括:

执行包括目标事务的程序指令,其中所述目标事务具有一个或多个事务处理程序指令,所述一个或多个事务处理程序指令执行以产生对状态数据的推测性更新,并且在所述目标事务完成且没有冲突时提交所述推测性更新;

检测与由处理硬件直接执行所述目标事务的预定事务处理程序指令相对应的触发条件,所述目标事务要进行诊断和/或调试操作;

在检测到所述触发条件时,发起对所述目标事务的执行的软件仿真,所述软件仿真操作来:

存储表示在仿真执行所述目标事务期间产生的所述推测性更新的一个或多个版本的数据以使得所述数据可用于所述诊断和/或调试操作;以及

检测与所述目标事务的冲突。

2.如权利要求1所述的方法,其中,所述目标事务具有多个事务处理程序指令,其中所述事务处理程序指令执行以产生对状态数据的推测性更新,并且在所述目标事务完成且没有冲突时提交所述推测性更新。

3.根据权利要求1和2中任一项所述的方法,其中,所述冲突包括访问所述目标事务所使用的存储器地址。

4.如权利要求3所述的方法,其中所述访问包括以下项中的一项:

向已经被所述目标事务访问的存储器地址写入;以及

从所述目标事务已经访问的存储器地址读取。

5.根据权利要求1和2中任一项所述的方法,其中,所述目标事务的所述软件仿真的单步执行允许所述软件仿真在仿真执行所述目标事务的各个指令之后停止。

6.根据权利要求1和2中任一项所述的方法,其中,所述程序指令包括多个目标事务,并且所述软件仿真操作来仿真多个并发执行的目标事务并检测所述多个并发执行的目标事务之间的冲突。

7.如权利要求6所述的方法,其中所述软件仿真操作来控制仿真执行所述多个并发执行的目标事务的相对速率。

8.根据权利要求1和2中任一项所述的方法,其中,所述目标事务的开始用事务开始指令标记,并且所述触发条件是执行所述事务开始指令。

9.根据权利要求1和2中任一项所述的方法,包括:

在检测到来自请求源的请求执行与经历软件仿真的所述目标事务冲突的存储器访问的请求时,向所述请求源发出非标准响应。

10.如权利要求9所述的方法,包括:

当在所述请求源处接收到所述非标准响应时,执行以下项中的一项:

暂缓所述请求源处的执行;

重试所述请求;以及

发起对所述请求源处的执行的软件仿真。

11.根据权利要求10所述的方法,包括:控制所述请求源处的软件仿真的执行和所述目标事务的软件仿真的相对速率。

12.根据权利要求1和2中任一项所述的方法,其中,所述软件仿真建模用于存储对状态

数据的所述推测性更新的资源可用性。

13.根据权利要求1和2中任一项所述的方法，其中，冲突检测硬件电路操作来在由所述处理硬件直接执行所述目标事务期间检测所述冲突，并且所述软件仿真重新使用所述冲突检测硬件电路来在仿真执行所述目标事务期间检测所述冲突。

14.根据权利要求13所述的方法，其中，所述软件仿真将所述冲突检测硬件电路配置为在仿真执行所述目标事务期间检测所述冲突。

15.一种用于处理数据的装置，包括：

处理硬件，用于直接执行包括目标事务的程序指令，其中所述目标事务包括一个或多个事务处理程序指令，所述一个或多个事务处理程序指令执行以产生对状态数据的推测性更新，并且在所述事务完成且没有冲突时提交所述推测性更新；

检测电路，用于检测与由所述处理硬件直接执行所述目标事务的预定事务处理程序指令相对应的触发条件，所述目标事务要进行诊断和/或调试操作；

仿真电路，用于在检测到所述触发条件时，执行对所述目标事务的执行的软件仿真，所述软件仿真操作来：

存储表示在仿真执行所述目标事务期间产生的所述推测性更新的一个或多个版本的数据以使得所述数据可用于所述诊断和/或调试操作；以及

检测与所述目标事务的冲突。

16.一种处理数据的方法，包括：

从请求源向请求目的地发出存储器访问请求；

在所述请求源处检测非标准响应信号的接收，所述非标准响应信号指示当所述请求目的地正在执行对事务的软件仿真时在所述请求目的地处检测到的冲突的存储器访问；

根据检测到所述非标准响应信号的接收，发起更新所述请求源的架构状态数据的第一路径的后续处理或者更新所述请求源的架构状态数据的第二路径的后续处理，所述第一路径与所述第二路径不同。

17.如权利要求16所述的方法，包括：在请求目的地处接收所述存储器访问请求，并且如果所述请求目的地正在执行事务的软件仿真，其中所述事务具有执行以产生对状态数据的推测性更新并且在所述事务完成且没有冲突时提交所述推测性更新的一个或多个事务处理程序指令，则将所述非标准响应信号返回到所述请求源。

18.根据权利要求16和17中任一项所述的方法，其中，所述第一路径是对程序指令执行的软件仿真。

19.根据权利要求16和17中任一项所述的方法，其中所述非标准响应信号具有分别触发不同的第一路径的后续处理的多种不同的形式。

20.一种用于处理数据的装置，包括：

发出电路，用于从请求源向请求目的地发出存储器访问请求；

检测电路，用于检测所述请求源处的非标准响应信号的接收，所述非标准响应信号指示当所述请求目的地正在执行对事务的软件仿真时在所述请求目的地处检测到的冲突的存储器访问；

处理路径控制电路，其根据检测到所述非标准响应信号的接收而操作，以发起更新所述请求源的架构状态数据的第一路径的后续处理或更新所述请求源的架构状态数据的第

二路径的后续处理，所述第一路径与所述第二路径不同。

21.一种处理数据的方法，包括：

执行包括事务的程序指令，其中所述事务具有一个或多个事务处理程序指令，所述一个或多个事务处理程序指令执行以产生对状态数据的推测性更新，并且在所述事务完成且没有冲突时提交所述推测性更新；

检测触发条件；

在检测到所述触发条件时，在完成所述事务之前停止所述事务的执行，并且将所述推测性更新中的至少一些推测性更新提交至状态数据。

22.根据权利要求21所述的方法，其中，所述事务具有多个事务处理程序指令，所述多个事务处理程序指令执行以产生对状态数据的推测性更新，并且在所述事务完成且没有冲突时提交所述推测性更新。

23.根据权利要求21和22中任一项所述的方法，包括：在检测到所述触发条件时，停止执行与所述事务交互的一个或多个并发执行的程序线程。

24.如权利要求23所述的方法，其中，所述事务具有由所述事务使用的工作数据集，并且所述方法包括将非标准响应信号返回到请求访问所述工作数据集中的任何数据的任何并发执行的程序线程。

25.如权利要求24所述的方法，其中，所述非标准响应信号由给定的并发执行的程序线程接收触发停止所述给定的并发执行的程序线程的执行。

26.根据权利要求23所述的方法，其中所述一个或多个并发执行的程序线程中的至少一个程序线程还执行另外的事务，并且停止执行所述另外的事务包括：在完成所述另外的事务之前停止执行所述另外的事务并且将所述另外的事务的至少一些推测性更新提交到状态数据。

27.根据权利要求21-22和24-26中任一项所述的方法，其中，检测所述触发条件包括以下项中的至少一项：

检测触发程序指令的执行；

检测具有与预定的触发程序计数器值匹配的相关联的程序计数器值的程序指令的执行；以及

检测对预定存储器地址的访问。

28.一种用于处理数据的装置，包括：

处理电路，用于执行包括事务的程序指令，其中所述事务具有一个或多个事务处理程序指令，所述一个或多个事务处理程序指令执行以产生对状态数据的推测性更新，并且在所述事务完成且没有冲突时提交所述推测性更新；

检测电路，用于检测触发条件；

控制电路，用于在检测到所述触发条件时操作来在完成所述事务之前停止执行所述事务并且将所述推测性更新中的至少一些推测性更新提交到状态数据。

调试数据处理事务

技术领域

[0001] 本公开涉及数据处理系统。

背景技术

[0002] 可以提供支持如下事务的数据处理系统，该事务中的多个程序指令执行以产生对状态数据的推测性更新，并且在事务完成且没有冲突时提交推测性更新。这样的事务可以被用于例如促进使用共享存储器的多线程处理，其中可以允许使用共享存储器内的数据值的事务根据在存储器访问之间出现冲突时避免进行推测性更新提交的能力并行地进行。在大多数情况下，不会出现这种冲突，因此并行处理可以有效地进行，而不需要支持更严格的机制(例如，使用存储器锁)的开销，但是当冲突确实出现时有可能恢复，因为对状态数据的推测性更新将不会被提交。

发明内容

[0003] 在本公开的至少一些示例性实施例中，提供了一种处理数据的方法，包括：

[0004] 执行包括目标事务的程序指令，其中该目标事务具有一个或多个程序指令，该一个或多个程序指令执行以产生对状态数据的推测性更新，并且在目标事务完成且没有冲突时提交推测性更新；

[0005] 检测与由处理硬件直接执行目标事务的程序指令相对应的触发条件；

[0006] 在检测到触发条件时，发起对目标事务的执行的软件仿真，软件仿真操作来：

[0007] 存储表示在仿真执行目标事务期间产生的推测性更新的一个或多个版本的数据；以及

[0008] 检测与目标事务的冲突。

[0009] 在本公开的至少一些示例性实施例中，提供了一种用于处理数据的装置，包括：

[0010] 处理硬件，用于直接执行包括目标事务的程序指令，其中该目标事务包括一个或多个程序指令，该一个或多个程序指令执行以产生对状态数据的推测性更新，并且在事务完成且没有冲突时提交推测性更新；

[0011] 检测电路，用于检测与由处理硬件直接执行目标事务的程序指令相对应的触发条件；

[0012] 仿真电路，用于在检测到触发条件时，执行对目标事务的执行的软件仿真，软件仿真操作来：

[0013] 存储表示在仿真执行目标事务期间产生的推测性更新的一个或多个版本的数据；以及

[0014] 检测与目标事务的冲突。

[0015] 在本公开的至少一些示例性实施例中，提供了一种处理数据的方法，包括：

[0016] 从请求源发出存储器访问请求；

[0017] 在请求源处检测非标准响应信号的接收；

[0018] 根据检测到非标准响应信号的接收，发起更新请求源的架构状态数据的第一路径的后续处理或者更新请求源的架构状态数据的第二路径的后续处理，所述第一路径与所述第二路径不同。

[0019] 在本公开的至少一些示例性实施例中，提供了一种用于处理数据的装置，包括：

[0020] 发出电路，用于从请求源发出存储器访问请求；

[0021] 检测电路，用于检测请求源处的非标准响应信号的接收；

[0022] 处理路径控制电路，其根据检测到非标准响应信号的接收而操作，以发起更新请求源的架构状态数据的第一路径的后续处理或更新请求源的架构状态数据的第二路径的后续处理，所述第一路径与所述第二路径不同。

[0023] 在本公开的至少一些示例性实施例中，提供了一种处理数据的方法，包括：

[0024] 执行包括事务的程序指令，其中该事务具有一个或多个程序指令，该一个或多个程序指令执行以产生对状态数据的推测性更新，并且在事务完成且没有冲突时提交推测性更新；

[0025] 检测触发条件；

[0026] 在检测到触发条件时，在完成事务之前停止事务的执行，并且将推测性更新中的至少一些推测性更新提交至状态数据。

[0027] 在本公开的至少一些示例性实施例中，提供了一种用于处理数据的装置，包括：

[0028] 处理电路，用于执行包括事务的程序指令，其中该事务具有一个或多个程序指令，该一个或多个程序指令执行以产生对状态数据的推测性更新，并且在事务完成且没有冲突时提交推测性更新；

[0029] 检测电路，用于检测触发条件；

[0030] 控制电路，用于在检测到触发条件时操作来在完成事务之前停止执行事务并且将推测性更新中的至少一些推测性更新提交到状态数据。

附图说明

[0031] 现在将仅通过示例的方式来参照附图描述示例性实施例：

[0032] 图1示意性地示出了包括事务的程序指令流；

[0033] 图2示意性地示出了用于执行程序指令的数据处理系统；

[0034] 图3是示意性地示出事务执行的流程图；

[0035] 图4是示意性地示出了软件仿真的事务执行的流程图；

[0036] 图5是示意性地示出事务之间可能产生的冲突的类型的图示；

[0037] 图6是示意性地示出向请求源发出非标准响应的流程图；

[0038] 图7是示意性地示出响应于接收到非标准响应控制后续处理的流程图；

[0039] 图8是示意性地示出包括多个处理器核心的数据处理装置的图示；以及

[0040] 图9是示意性地示出当检测到触发条件时停止执行保存推测性更新的事务的流程图。

具体实施方式

[0041] 当执行作为事务的一部分的程序指令时，其中该程序指令产生推测性更新，该推

测性更新在事务未完成且没有冲突时被丢弃，在获取可用于了解此类事务中出现的故障的诊断数据(调试数据)方面存在困难。特别地，在处理中的特定点处发生执行的停止的传统诊断/调试机制(例如断点和观察点)通常具有丢失与部分完成的事务相关联的推测性更新的副作用，因为这些推测性更新通常是由于调用诊断/调试机制而被刷新的。在本公开的一些实施例中，可以通过如下操作来解决这个问题：检测与由处理硬件直接执行目标事务相对应的触发条件，并且使用此来启动对该目标事务的软件仿真以代替直接执行。软件仿真可以操作来存储/保存在仿真期间产生的推测性更新的一个或多个版本，以使得这些版本可用于调试/诊断操作。软件仿真可以持续检测与被仿真的事务的任何冲突，以便它以与由处理硬件直接执行基本对应的方式与剩余的系统部分交互。

[0042] 一些事务可以包括执行来产生对状态数据的推测性更新的一个程序指令。例如，在某些情况下，事务开始和事务结束之间可能只有一个事务。而且，在某些情况下，单个执行的指令可能会触发若干对状态数据的推测性更新，这些推测性更新可能在它们全部完成且没有冲突时被提交。因此，在某些情况下，事务可以包括触发推测性更新的单个指令。

[0043] 然而，事务通常可能包括多个程序指令，每个程序指令产生对状态的推测性更新，推测性更新在事务完成且没有冲突时被提交。

[0044] 虽然事务之间的冲突可能采取多种形式(例如竞争共享资源)，但是一种常见的冲突来源是访问被仿真的目标事务所使用的存储器地址。软件仿真可以捕获这种访问并将其表示为冲突。该访问可以是对所追踪的与使用中的那些相同的地址或者相同的区域内的地址的任何访问(例如，读取或写入)。

[0045] 在一些示例性实施例中可以支持的有用特征是软件仿真提供了对目标事务的单步(single-stepping)执行，以使得可以在仿真执行目标事务的单个指令后停止软件仿真。这可以允许对要获取的目标事务的行为具有更精细的理解

[0046] 虽然仿真可能被限制于单个事务，但是本公开还提供了多个目标事务可以被并发地进行软件仿真。软件仿真可以用于检测在由软件仿真所支持的并发执行的这多个目标事务之间产生的冲突。

[0047] 在一些示例实施例中，目标事务的开始可以由事务开始指令标记。在这种情况下，触发启动软件仿真的触发条件可以是执行目标事务的这样的事务开始指令。

[0048] 当给定的目标事务受到软件仿真时，可能期望识别与被软件仿真的目标事务相冲突的存储器访问的请求源。可以通过向发出这种冲突的存储器访问请求的请求源发出非标准响应来促进这种行为。然后，这种非标准响应可以用于触发请求源中的非标准行为，以便于促进对其行为的理解。这样的非标准响应可以包括例如暂缓请求源的执行、重试请求或者发起对请求源处的执行的软件仿真。

[0049] 在本公开的一些示例性实施例中，可以促进对系统有潜在冲突的不同部分的交互的理解，其中可以控制对冲突的请求源的执行的软件仿真和对目标事务的软件仿真的相对速率。

[0050] 在本公开的一些示例性实施例中，软件仿真还可以用于针对用于存储对状态数据的推测性更新的资源的可用性进行建模。这种资源的可用性可能是造成不正确操作的一个问题，因此仿真这种行为是有用的。

[0051] 支持事务的数据处理系统通常包括检测硬件电路，其用于检测在基于硬件直接执

行事务期间所产生的冲突。这种检测硬件电路可以例如包括用于支持数据一致性的存储器窥探硬件。在这种系统中，对事务的软件仿真可以重新利用冲突检测硬件电路来检测在仿真执行目标事务期间产生的冲突。软件仿真可以利用适当的数据/参数来对冲突检测硬件电路进行编程，使得其用于检测与被软件仿真的事务的冲突。

[0052] 根据是否响应于存储器访问请求而接收到非标准响应信号，本公开的一些示例性实施例支持不同形式的行为。在这样的实施例中，根据是否已经接收到非标准响应信号，遵循进行来更新架构状态数据的第一路径的后续处理或遵循以不同方式进行来更新架构状态数据的第二路径的后续处理。这些不同的路径是用于进一步处理和推进进程的正当路径。

[0053] 在一些示例实施例中，当在请求目的地执行对事务的软件仿真时在请求目的地处检测到冲突的存储器访问时，可以返回非标准响应信号。以这种方式，通过返回非标准响应信号，与软件仿真的事务交互的请求源可以被触发以遵循与其他方式相比不同的处理路径。例如，非标准响应信号可以触发请求源本身开始对其处理的软件仿真。

[0054] 虽然应当理解，对于基于事务的处理的常规假设是如果出现冲突，则不提交推测性更新，但在本公开的一些示例实施例中，该模型可以被破坏，以使得在检测到触发条件时，停止执行事务并且提交对状态数据的推测性更新，即使事务尚未正常完成。这种行为偏离与事务相关联的结构性假设，因此正在进行的处理可能不再可能继续，但在某些情况下，对获得的状态数据的推测性更新的可见性可以允许有益地理解对于其他方式而言可能困难的系统的行为。

[0055] 除了冻结事务(针对该事物检测到触发条件)的执行，在本公开的一些实施例中，还可以停止执行与该事务交互的一个或多个并发执行的程序线程。这可以通过以下方式促进：使用如先前所述的非标准响应信号来触发停止执行任何并发执行的程序线程，该程序线程请求访问由本身已经被停止的事务所使用的工作数据集合内的任何数据。这些另外的线程本身可能是另外的事务，这些事务然后可以保存其推测性更新，即使它们尚未完成。

[0056] 应当理解，与停止部分完成的事务并保存该部分完成的事务的推测性更新相关联的触发条件可以采取各种不同的形式。这些形式可以包括例如检测触发的程序指令的执行；检测具有与预定触发的程序计数器匹配的相关联的程序计数器值的程序指令的执行；和/或检测对预定存储器地址或地址范围的访问。

[0057] 图1示意性地示出了可以由处理电路执行的程序指令流2。应当理解，该程序指令流可以是在多线程系统内执行的一个线程。或者，该程序指令流可以是运行的单线程程序指令。在图1所示的程序指令流中，包括事务开始指令TStart和事务结束指令TEnd。这些指令分别指示包括指令IA、IB、IC和ID的事务的边界。由TStart和TEnd限制的这些事务由处理电路执行以产生对状态数据的推测性更新。这些推测性更新被存储在系统的存储器或其他存储资源(例如，影子寄存器、允许回滚的专用存储器等)中，直到确定事务已经完成且没有冲突，此时，推测性更新被提交至系统(例如，利用所存储的回滚数据来更新系统的架构状态(例如形成系统的程序员模型的状态)，该回滚数据然后被丢弃以释放相关联的存储器资源以支持另外的事务)。

[0058] 图2示意性地示出了包括处理器核心6和存储器系统8的数据处理系统4。存储器系统8可以是包括主存储器以及一个或多个级别的缓存存储器的分级系统。分级的存储器系

统8可以由其他处理器或与同一处理器上运行的其他线程共享。处理器核心6提供用于执行从存储器8取出的程序指令的处理电路。程序指令执行的结果可以写回存储器8。加载和存储程序指令可以被用于从存储器8读取数据以及向存储器8写入数据。处理器核心6包括寄存器组10、乘法器12、移位器14和加法器16，它们一起执行处理操作以执行程序指令，直到控制由指令解码器18产生的作为程序指令的控制信号到达指令流水线20内的解码阶段。还与处理器核心6相关联的是调试和诊断电路22，其用于执行例如指令采样以捕获指令诊断数据并且跟踪包括被采样的指令的事务以便产生事务诊断数据。该指令诊断数据和事务诊断数据可以被写出到存储器8或者输出在跟踪数据流中(这依赖于实施例和/或操作支持这两个潜在输出机制的实施例的要求)。调试和诊断电路22还可以检测开始对事务的软件仿真的触发条件。

[0059] 图3是示意性地示出执行包括多个程序指令的事务的流程图。在步骤24处，处理等待，直到事务开始指令被执行。步骤26分配用于例如在事务在没有冲突的情况下完成并且推测性更新被提交之前存储对状态数据的推测性更新的事务资源。步骤28选择事务内的第一指令。步骤30确定在当时是否检测到任何冲突或中止。如果检测到任何此类冲突或中止，则步骤32用于丢弃任何推测性更新并返回在步骤26处分配的事务资源以用于其他事务。

[0060] 如果在步骤30处没有检测到冲突或中止，则步骤34用于执行所选择的指令。步骤36将步骤34的执行的结果作为对状态数据的推测性更新存储在所分配的事务资源内。步骤38选择下一指令。步骤40确定所选择的指令是否是事务结束指令。如果指令不是事务结束指令，则处理返回到步骤30。如果步骤40确定所选择的指令是事务结束指令，则步骤42用于提交存储在所分配的事务资源内的推测性更新，以便更新系统的架构状态，因为事务已经执行且没有冲突或中止产生。

[0061] 图4是示意性地示出软件仿真的事务执行的流程图。在步骤200处，处理等待，直到调试和诊断电路22检测到触发条件。这种触发条件可以采取各种不同的形式。一个示例形式是执行与特定事务相关联的事务开始指令。可以实现此的一种方式是检测具有与预定程序计数器值匹配的程序计数器值并且对应于要进行诊断/调试的目标事务的开始的事务开始指令的执行。

[0062] 当检测到触发条件时，步骤202用于停止目标事务的硬件执行。步骤204开始对目标事务的软件仿真。该软件仿真可以是促进对目标事务的行为进行单步分析的逐指令软件仿真(即单步执行)。

[0063] 步骤206用于在软件的控制下针对每个被仿真的指令存储与目标事务相关联的对状态数据的推测性更新。这提供了存储的多版本的推测性更新，以促进对目标事务行为的理解。步骤208确定对于正在进行软件仿真的目标事务是否出现任何冲突或中止。如果出现这种冲突或中止，则软件仿真的事务的执行结束。如果没有检测到冲突或中止，则步骤210用于将执行点前进到要被仿真的下一指令。步骤212确定下一指令是否是事务结束指令。如果下一指令是事务结束指令，则步骤214用于将推测性更新提交到系统的架构状态。如果步骤212确定下一指令不是事务结束指令，则步骤216用于对下一指令进行仿真，并且处理返回到步骤206。

[0064] 图5示意性地示出了可能在并发执行的事务之间出现的冲突的各种形式。线程0受软件仿真控制。系统中作为线程0所使用的工作数据集的一部分的存储器地址空间内的区

域在图5中用“#”标记。

[0065] 在线程1的情况下，执行对用于线程0的数据的一部分的写入操作(作为动作a0)。冲突检测硬件电路将此标识为冲突。这种冲突检测硬件电路可以包括例如提供来支持共享存储器操作(并且被编程/配置为软件仿真的设置的一部分)的窥探或数据一致性机制。检测到的冲突用于触发将通知信号发送到线程0(作为动作a1)，以使得线程0的仿真能感知冲突。此外，在步骤a2处，触发线程1以开始对线程1的软件仿真并触发暂停的信号被发送。

[0066] 可能出现的冲突的另外的示例关于线程0和线程2被示出。在这种情况下，线程0执行对其工作数据集中的存储器地址的写入(作为步骤b0)。随后，线程2执行对该存储器地址的读取(或者至少针对存储器中与所访问的存储器正被跟踪的粒度相对应的相同区域)。该读取操作在图5中被表示为b1。冲突读取触发通过动作b2中止被软件模拟的线程0，并返回非标准响应信号NACK(作为动作b3)。非标准响应信号被发送至线程2，其中线程2是b1所发送的读取请求的请求源。

[0067] 图6是示意性地示出了检测具有工作数据集的事务与系统内的其他处理之间的冲突的流程图。在步骤218处，检测对访问被仿真的事务的工作数据集内的存储器地址的请求。步骤220确定该请求是否是向被仿真的事务已经访问的地址的写入。如果请求是写入，则步骤222以例如NACK信号的形式向源发出非标准响应请求，如图5所示。步骤222确定请求是否是读取被仿真的事务已经访问的地址。如果请求是读取，则可以再次发出非标准响应信号。如果步骤220或步骤224的确定都不是肯定的，则步骤226用于允许访问执行。

[0068] 图7示意性地示出请求源响应于接收到非标准响应的行为。在步骤228处，处理等待，直到存在要执行的存储器访问。步骤230将存储器访问请求发送到请求目的地。步骤232等待，直到从请求目的地接收到响应。步骤234确定接收到的响应是否是非标准响应。如果接收到的响应是非标准响应，则步骤236用于触发请求源遵循第一路径的后续处理，例如暂缓、重试该请求或对请求源处的处理进行软件仿真。如果返回的响应不是非标准响应，则步骤238用于控制请求源遵循第二路径的后续处理，诸如执行在步骤228处所请求的访问。应当理解，第一路径的后续处理和第二路径的后续处理都用于更新系统的架构状态，并且都是用于在所执行的处理中促进进程的合法路径。

[0069] 图8示意性地示出了包括多个处理器核心(Core)242、244、246和248的数据处理设备，每个处理器核心具有本地缓存存储器。共享的缓存存储器250以及主存储器252也在数据处理装置240的分级存储器系统内被提供。窥探控制单元(SCU)和事务(TM)支持电路254被提供并且在各点处被耦合到存储器系统。窥探控制单元和事务支持电路254用于确保使用共享存储器系统的不同处理线程的数据一致性，以及用于检测在部分完成的事务(具有与其相关联的对状态数据的推测性更新)和系统内的其他存储器访问之间出现的冲突。如果检测到这样的冲突，则可能会中止所涉事务，并丢弃推测性更新。

[0070] 图8中还示出了调试电路256，其用于支持数据处理设备240的调试和诊断操作。调试电路256包括检测电路258，其检测用于触发诊断或调试操作(诸如对事务的软件仿真、事务的暂缓或重试等)的启动的触发条件。检测电路可以检测各种不同形式的触发条件，包括例如触发执行程序指令，执行具有与预定程序计数器匹配的相关程序计数器值的程序指令和/或检测对预定存储器地址或存储器地址区域的访问。

[0071] 图9是示意性地示出了在理解的事务的一些实施例中可以提供的另外的示例行为

的流程图。在步骤260处，处理等待，直到检测到事务的执行。然后，步骤262确定是否发生了与“冻结”执行该事务相关联的触发条件。如果没有出现这样的触发条件，则步骤264确定事务是否仍在执行，如果是，则返回到步骤262的处理。如果事务在没有检测到触发条件的情况下终止，则处理返回到步骤260。

[0072] 如果在步骤262处检测到触发条件，则步骤266用于将为正被执行的事务所保持的推测性更新提交至系统的架构状态。这违反了事务的正常行为，其中系统的架构状态通常仅在整个事务完成且没有检测到冲突的情况下才更新。然而，在执行事务中途保存推测性更新可以提供用于理解事务的行为的有用信息。

[0073] 然后，处理进行到步骤268，其中针对是否对已经停止的事务作出了任何冲突的请求做出判定。如果接收到任何此类冲突的请求，则步骤270用于向冲突的请求的请求源返回非响应信号。该非标准响应信号可以用于触发请求源自身停止其执行并保存其状态，例如，如果请求源是另一个事务，则该另一个事务可能会停止其操作，并在其执行中途保存其状态。

[0074] 尽管在此参照附图详细描述了说明性实施例，但是应当理解，权利要求书不限于这些精确的实施例，并且本领域技术人员可以对其进行各种改变、添加和修改而不脱离所附权利要求的范围和精神。例如，可以将独立权利要求的特征与从属权利要求的特征进行各种组合。

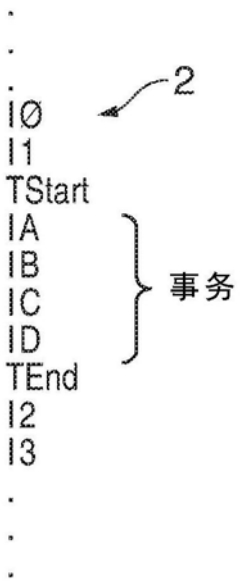

2
I0
I1
TStart
IA
IB
IC
ID
事务
TEnd
I2
I3

图1

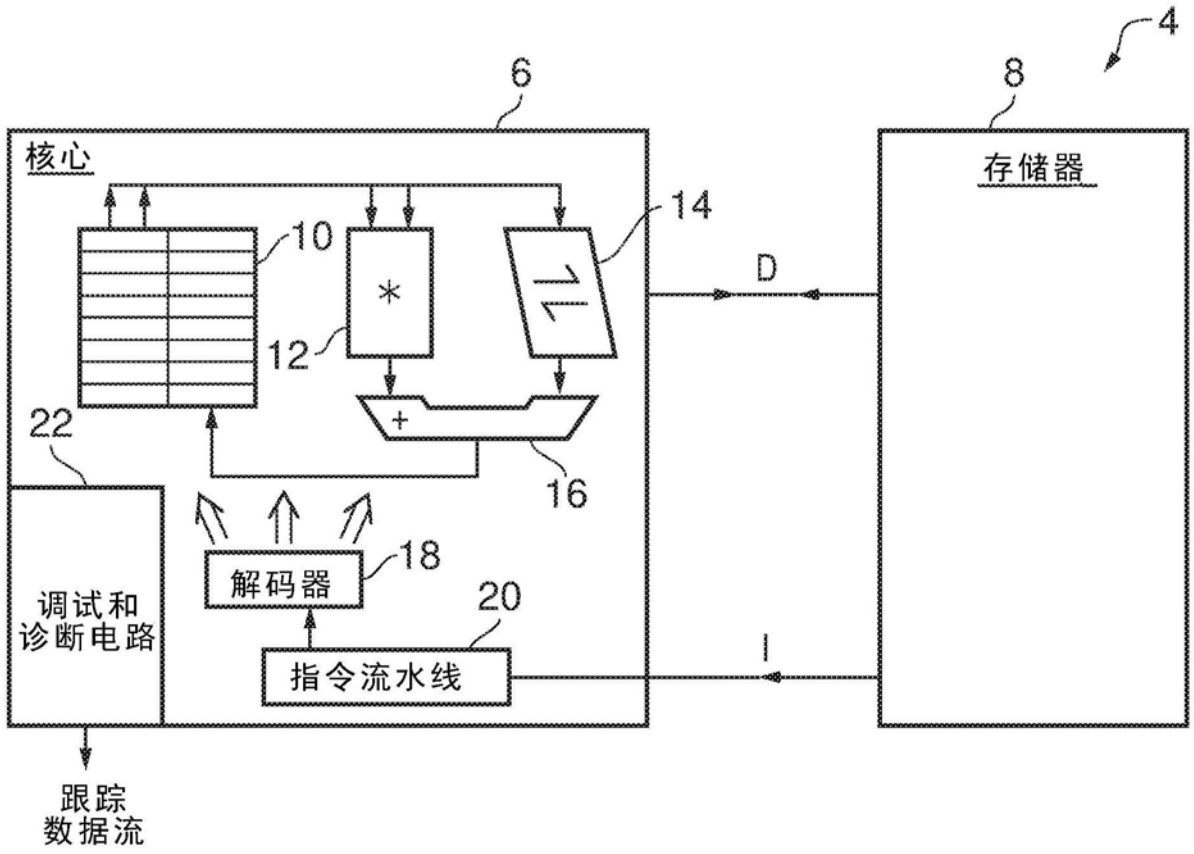

4
6
8
核心
存储器
10
12
14
16
18
20
22
D
I
解码器
指令流水线
调试和
诊断电路
跟踪
数据流

图2

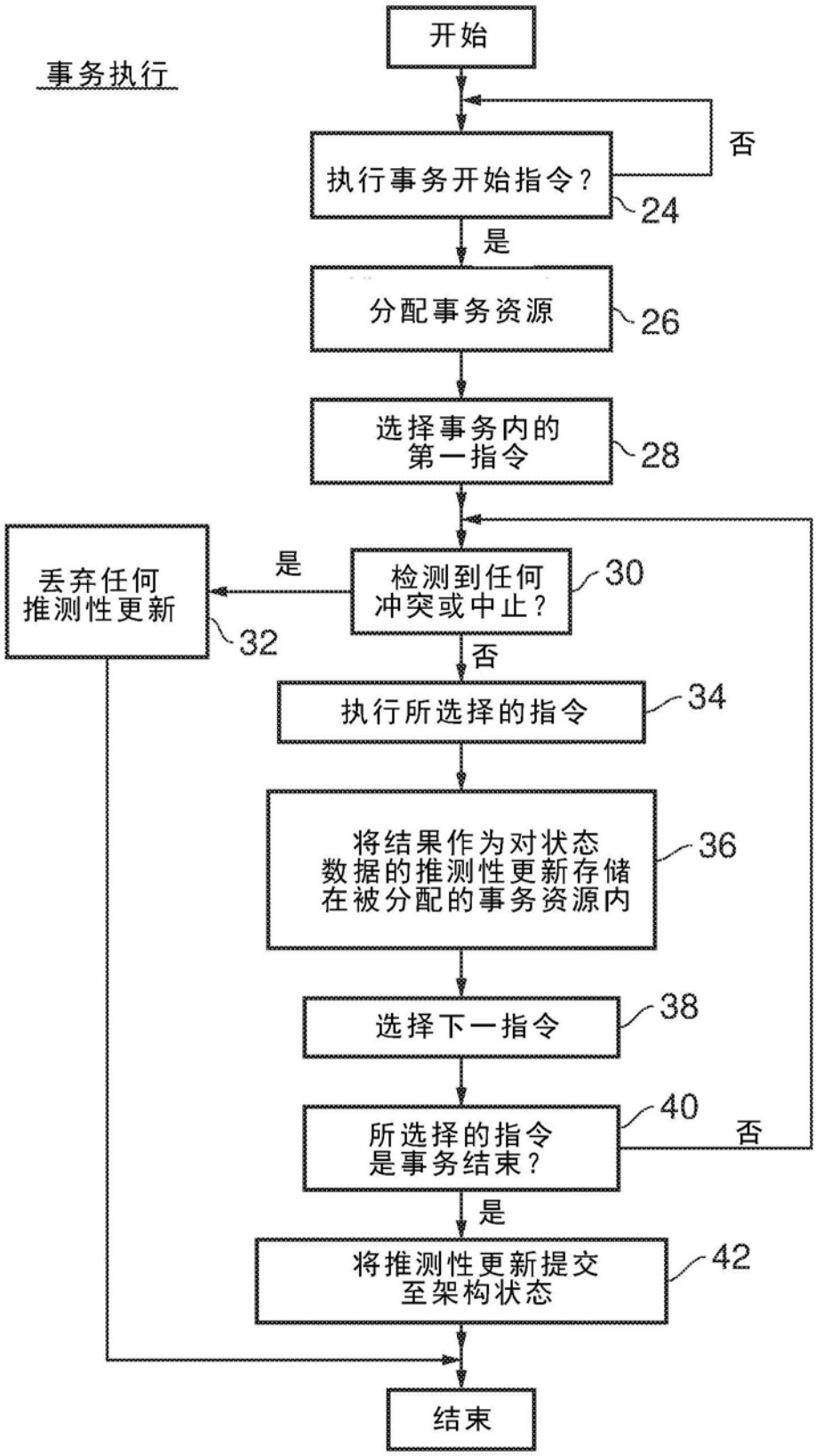

事务执行
开始
执行事务开始指令？
否
24
是
分配事务资源
26
选择事务内的
第一指令
28
丢弃任何
推测性更新
32
是
检测到任何
冲突或中止？
30
否
执行所选择的指令
34
将结果作为对状态
数据的推测性更新存储
在被分配的事务资源内
36
选择下一指令
38
所选择的指令
是事务结束？
40
否
是
将推测性更新提交
至架构状态
42
结束

图3

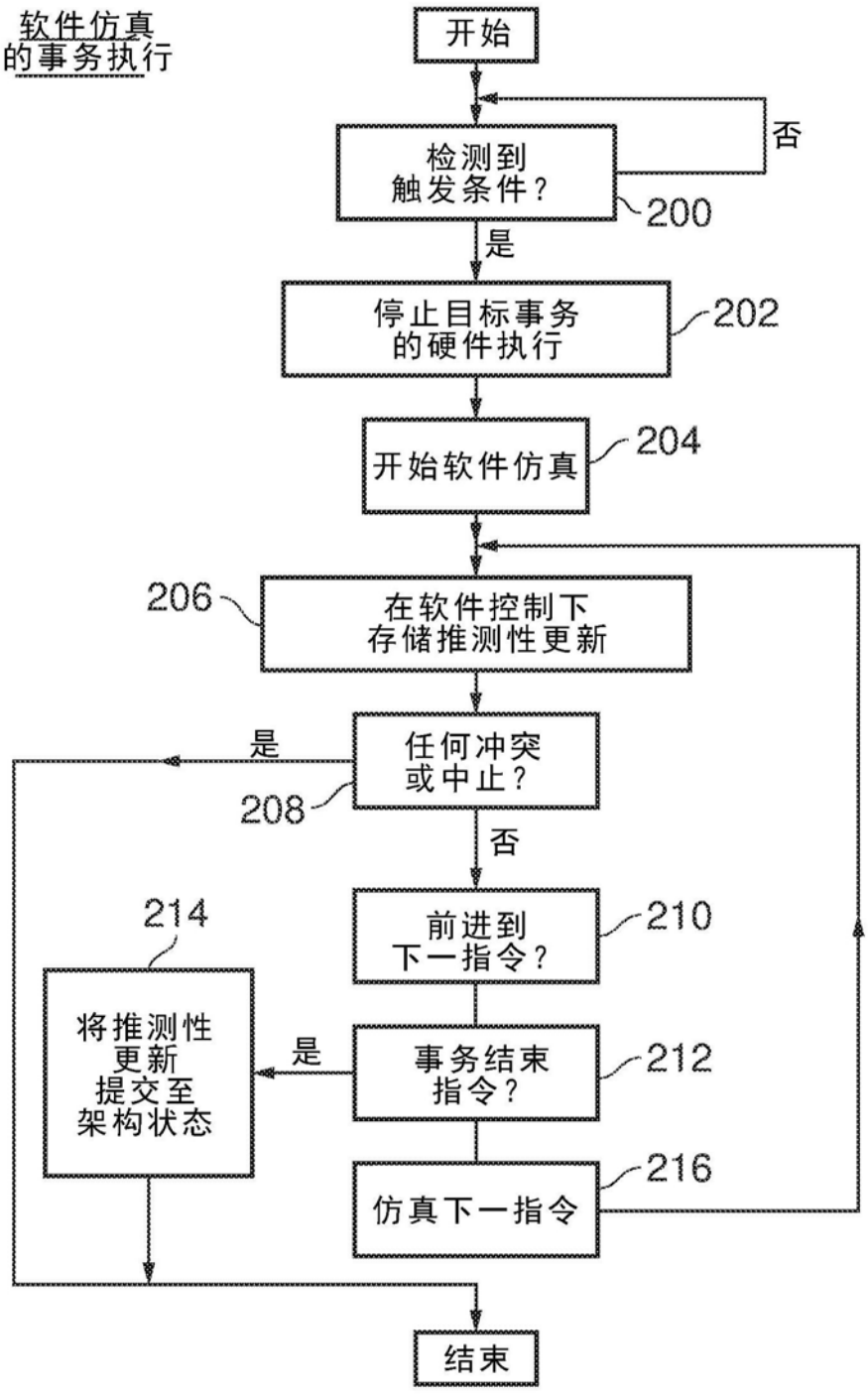

软件仿真
的事务执行
开始
检测到
触发条件？
否
200
是
停止目标事务
的硬件执行
202
开始软件仿真
204
206
在软件控制下
存储推测性更新
是
任何冲突
或中止？
208
否
214
前进到
下一指令？
210
将推测性
更新
提交至
架构状态
是
事务结束
指令？
212
仿真下一指令
216
结束

图4

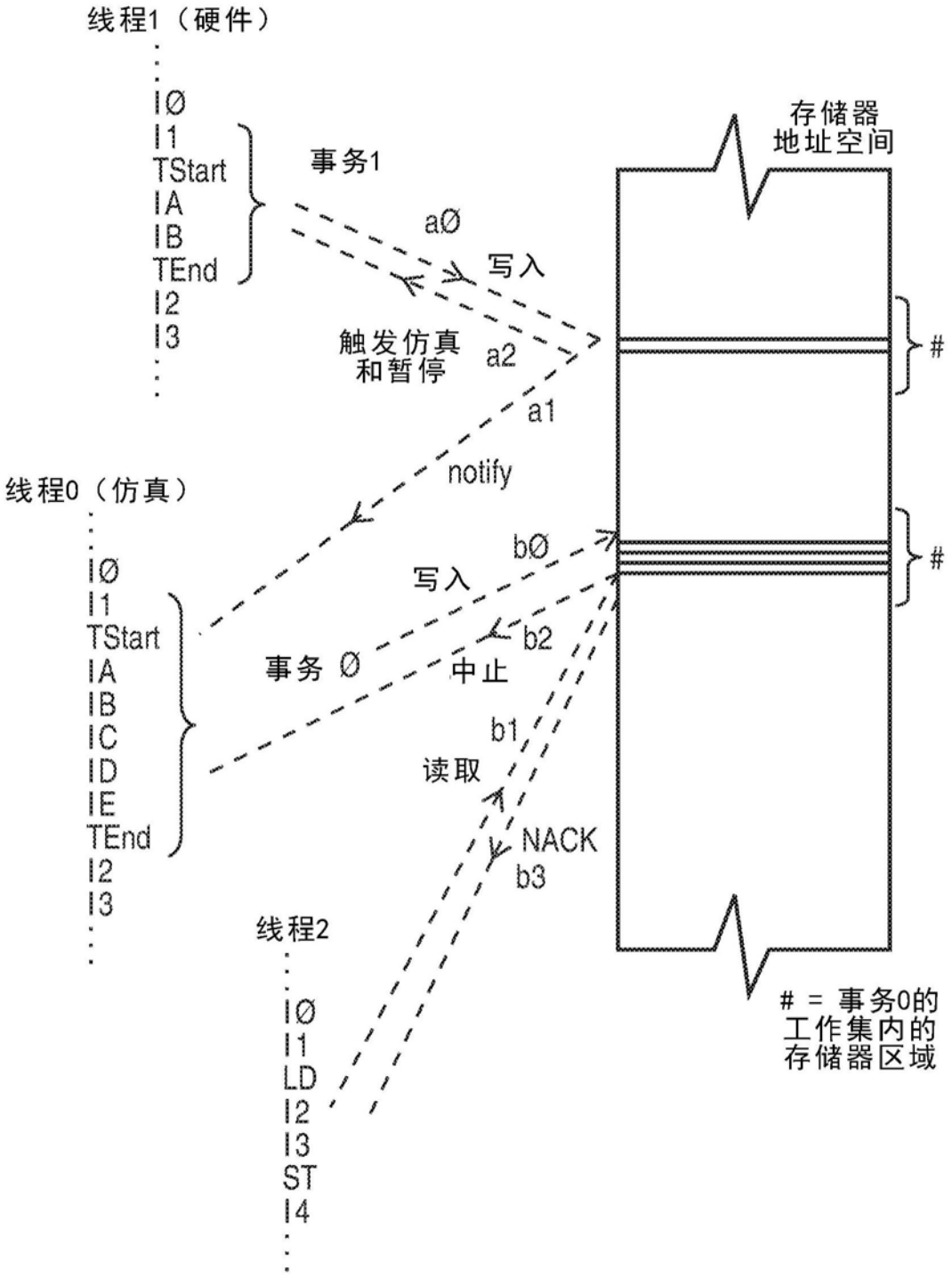

线程1（硬件）
I0
I1
TStart
IA
IB
TEnd
I2
I3
事务1
a0
写入
触发仿真
和暂停
a2
a1
notify
存储器
地址空间
#
#
线程0（仿真）
I0
I1
TStart
IA
IB
IC
ID
IE
TEnd
I2
I3
b0
写入
b2
事务 0
中止
b1
读取
NACK
b3
线程2
I0
I1
LD
I2
I3
ST
I4
= 事务0的
工作集内的
存储器区域

图5

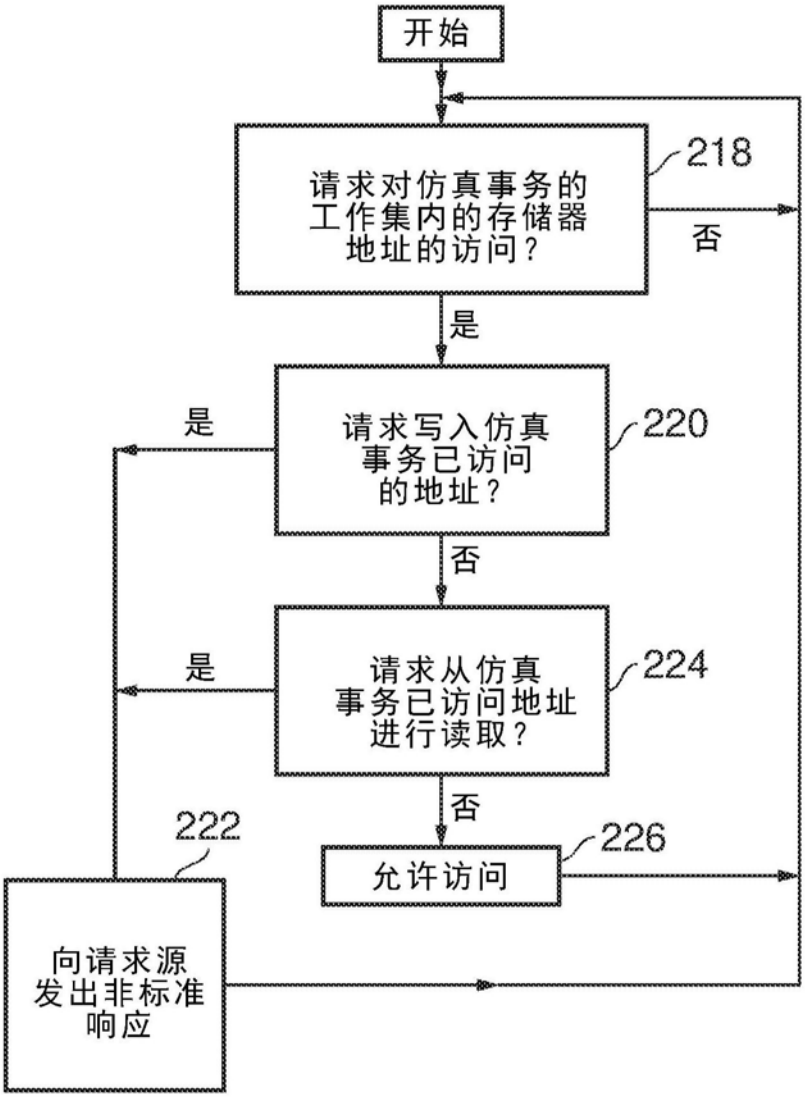

开始
请求对仿真事务的工作集内的存储器地址的访问？
218
否
是
请求写入仿真事务已访问的地址？
220
是
否
请求从仿真事务已访问地址进行读取？
224
是
否
222
允许访问
226
向请求源发出非标准响应

图6

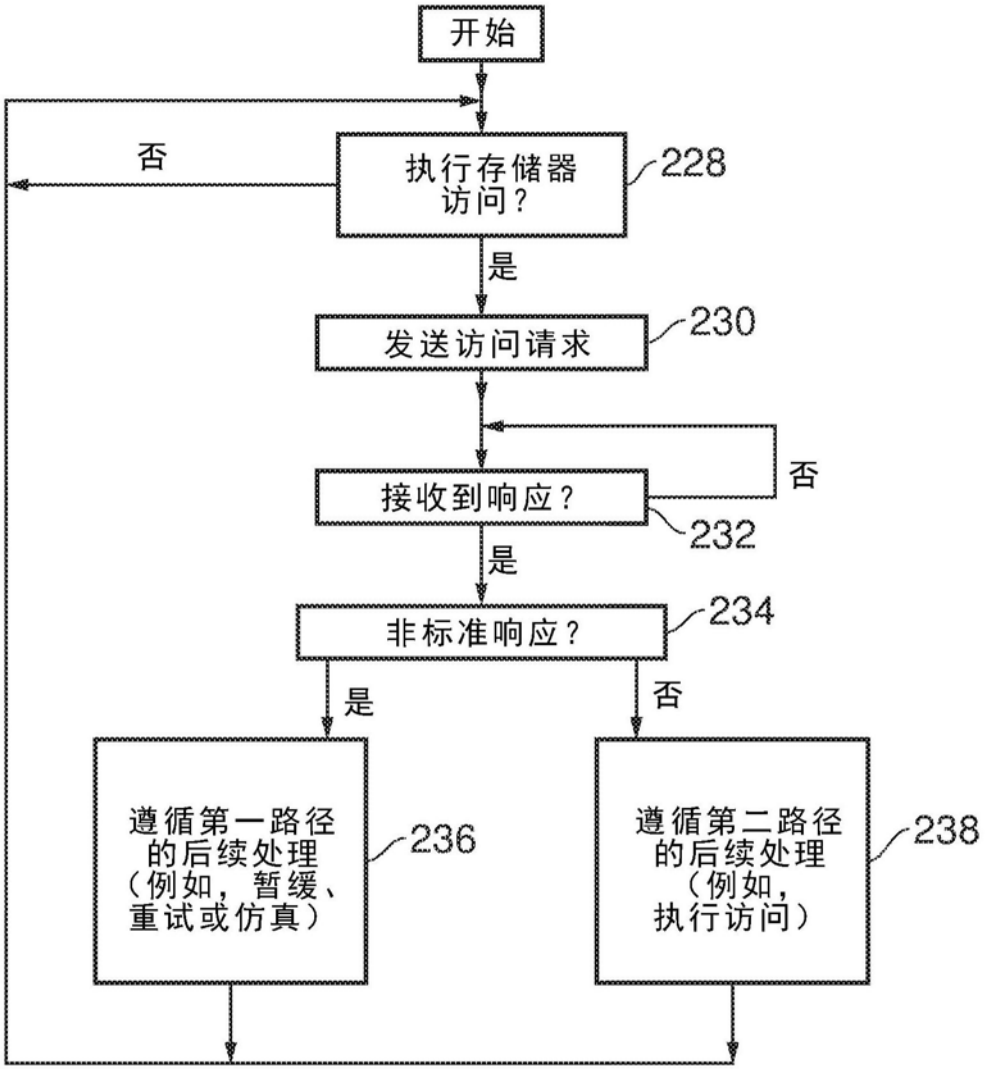

开始
执行存储器访问？
228
否
是
发送访问请求
230
接收到响应？
232
否
是
非标准响应？
234
是
否
遵循第一路径的后续处理（例如，暂缓、重试或仿真）
236
遵循第二路径的后续处理（例如，执行访问）
238

图7

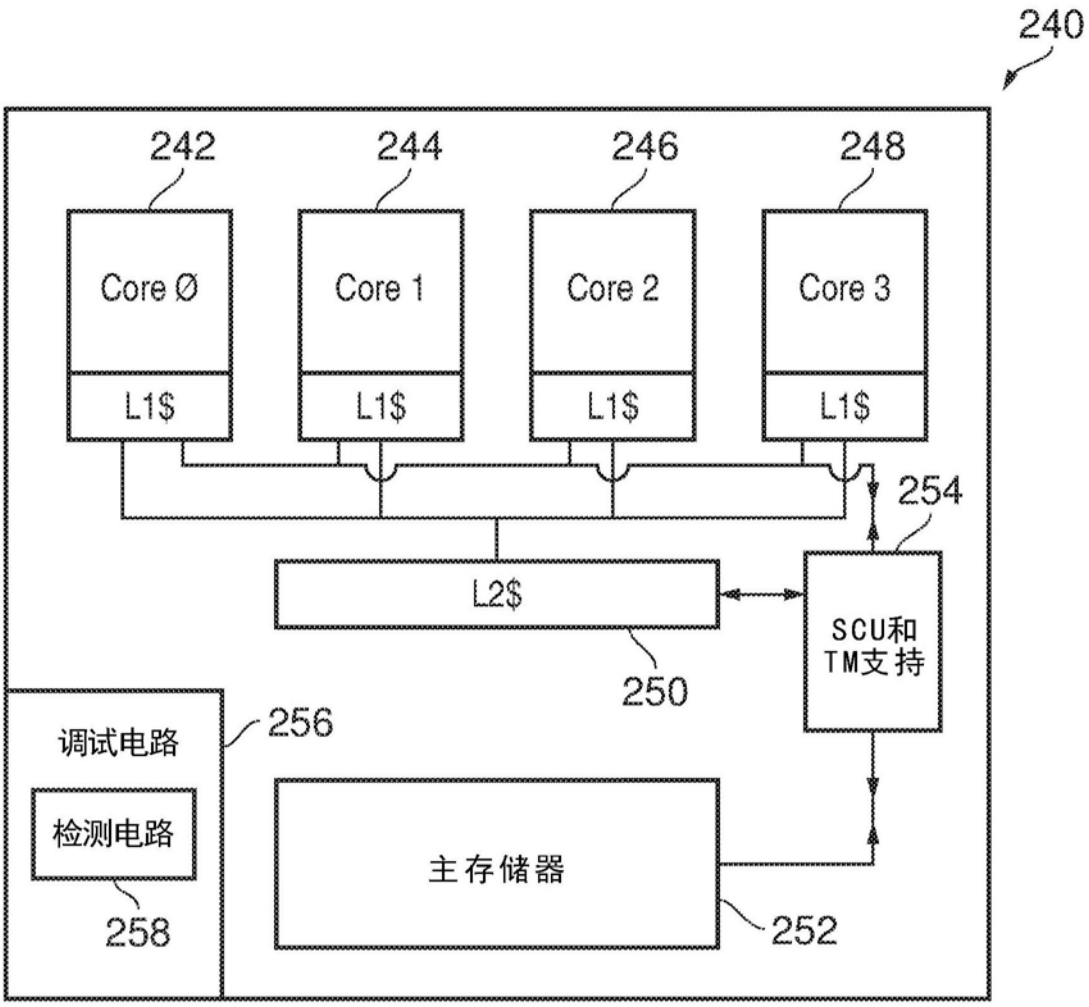

240
242
244
246
248
Core Ø
Core 1
Core 2
Core 3
L1$
L1$
L1$
L1$
254
L2$
250
SCU和
TM支持
调试电路
256
检测电路
258
主存储器
252

图8

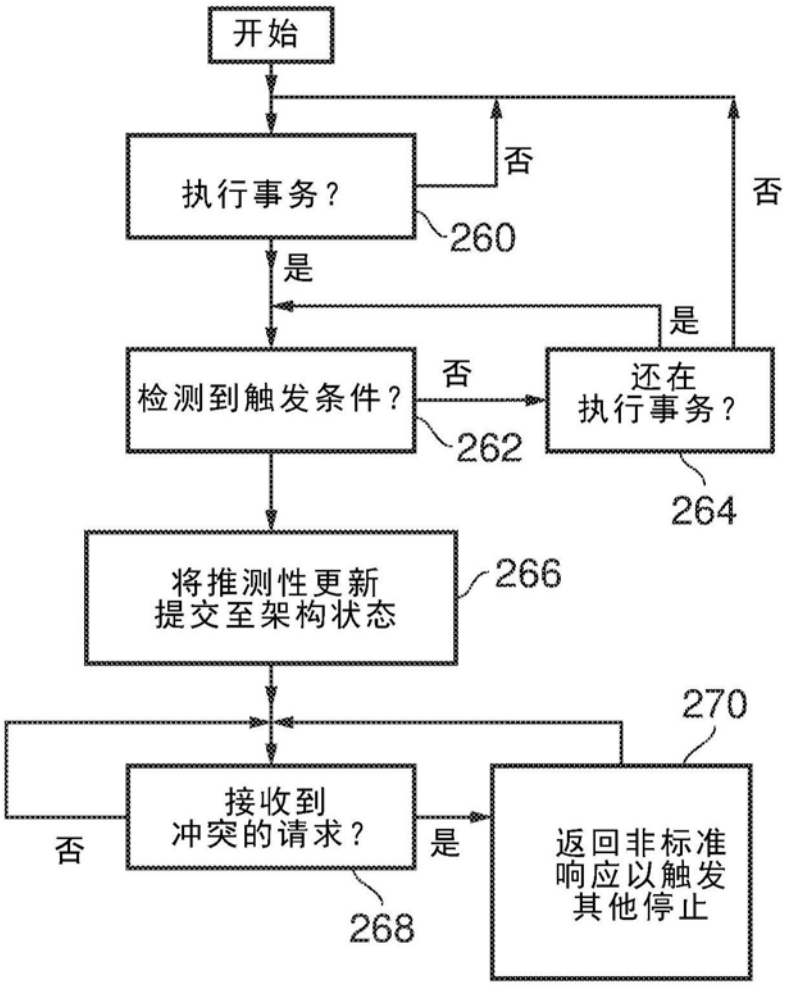

开始
执行事务？
否
260
是
否
是
检测到触发条件？
否
262
还在
执行事务？
264
将推测性更新
提交至架构状态
266
270
接收到
冲突的请求？
否
是
返回非标准
响应以触发
其他停止
268

图9