(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 106233254 B
(45)授权公告日 2019.04.26

(21)申请号 201580015831.9

(22)申请日 2015.03.16

(65)同一申请的已公布的文献号
申请公布号 CN 106233254 A

(43)申请公布日 2016.12.14

(30)优先权数据
14/226,947 2014.03.27 US

(85)PCT国际申请进入国家阶段日
2016.09.23

(86)PCT国际申请的申请数据
PCT/EP2015/055447 2015.03.16

(87)PCT国际申请的公布数据
WO2015/144479 EN 2015.10.01

(73)专利权人 国际商业机器公司
地址 美国纽约

(72)发明人 D·格雷纳 M·法雷尔
D·L·奥西塞克 D·W·施密特
F·Y·布萨巴 J·P·库巴拉
J·D·布拉德伯里 L·C·海勒
T·斯莱格尔 C·小盖尼

(74)专利代理机构 北京市中咨律师事务所 11247
代理人 于静 张亚非

(51)Int.Cl.
G06F 9/50(2006.01)
G06F 9/38(2006.01)
G06F 9/30(2006.01)

(56)对比文件
US 2008114973 A1,2008.05.15,
US 6341347 B1,2002.01.22,
US 2008256339 A1,2008.10.16,
CN 102880552 A,2013.01.16,
CN 101763285 A,2010.06.30,
邓宇 等.Matrix DSP中多线程机制的研究与设计.《计算机科学》.2013,第40卷(第4期),第51-54页.

审查员 王婉君

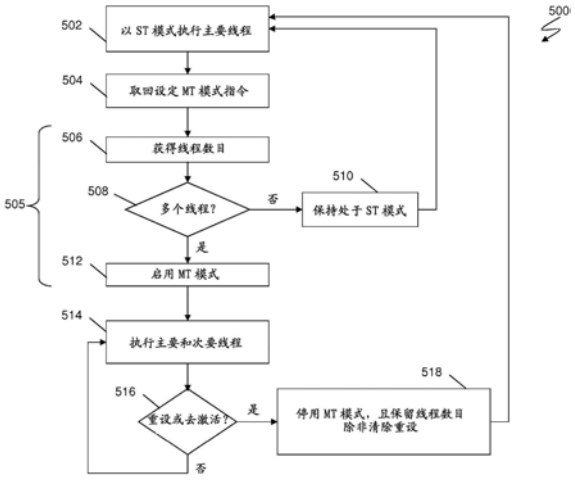
权利要求书2页 说明书17页 附图17页

(54)发明名称

多线程计算机系统中的地址扩展及缩短

(57)摘要

一种计算机系统包括配置,所述配置具有能在单线程(ST)模式与多线程(MT)模式之间配置的核心。所述ST模式处理主要线程,且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程。多线程工具被配置为控制对所述配置的利用以执行一种方法,所述方法包括在所述ST模式中使用核心地址值存取所述主要线程及自所述ST模式切换至所述MT模式。在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者,其中所述扩展后的地址值包括与线程地址值串接的所述核心地址值。



CN 106233254 B

1.一种计算机系统,包括:

配置,其包括能在单线程(ST)模式与多线程(MT)模式之间配置的核心,所述ST模式处理主要线程,且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程;及

多线程工具,其被配置为控制对所述配置的利用以执行一种方法,所述方法包括:

在所述ST模式中使用核心地址值存取所述主要线程;

自所述ST模式切换至所述MT模式;及

在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者,所述扩展后的地址值包括与线程地址值串接的所述核心地址值,其中所述扩展后的地址值包括移位后的核心地址值,所述移位后的核心地址值被移位基于所请求最大线程识别符的量。

2.如权利要求1所述的计算机系统,其中所述线程地址值被串接至所述核心地址值的低阶位以形成所述扩展后的地址值。

3.如权利要求1所述的计算机系统,其中所述方法进一步包括在所述MT模式与所述ST模式之间切换,且进一步其中基于所述核心处于相应ST模式或MT模式中而选择所述核心地址值或所述扩展后的地址值中的一者。

4.如权利要求3所述的计算机系统,其中在所述ST模式中使用标准格式地址,且所述核心基于停用所述MT模式而自所述MT模式恢复至所述ST模式。

5.如权利要求4所述的计算机系统,其中基于停用所述MT模式,仅所述主要线程为可存取的且所述一个或多个次要线程并非可存取的。

6.如权利要求4所述的计算机系统,其中自所述MT模式至所述ST模式的恢复进一步包括:使所述扩展后的地址值移位及去除所述线程地址值。

7.如权利要求1所述的计算机系统,其中基于程序指定最大线程识别符确定所述线程地址值中的线程识别符位的数目。

8.一种用于配置中的地址调整的计算机实施的方法,所述配置包括能在单线程(ST)模式与多线程(MT)模式之间配置的核心,所述ST模式处理主要线程,且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程,所述方法包括:

在所述ST模式中使用核心地址值存取所述主要线程;

自所述ST模式切换至所述MT模式;及

在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者,所述扩展后的地址值包括与线程地址值串接的所述核心地址值,其中所述扩展后的地址值包括移位后的核心地址值,所述移位后的核心地址值被移位基于所请求最大线程识别符的量。

9.如权利要求8所述的方法,其中所述线程地址值被串接至所述核心地址值的低阶位以形成所述扩展后的地址值。

10.如权利要求8所述的方法,进一步包括:

在所述MT模式与所述ST模式之间切换,其中基于所述核心处于相应ST模式或MT模式中而选择所述核心地址值或所述扩展后的地址值中的一者。

11.如权利要求10所述的方法,其中在所述ST模式中使用标准格式地址,且所述核心基

于停用所述MT模式而自所述MT模式恢复至所述ST模式。

12.如权利要求11所述的方法,其中基于停用所述MT模式,仅所述主要线程为可存取的且所述一个或多个次要线程并非可存取的。

13.如权利要求8所述的方法,其中基于程序指定最大线程识别符确定所述线程地址值中的线程识别符位的数目。

14.一种用于配置中的地址调整的计算机程序产品,所述配置包括能在单线程(ST)模式与多线程(MT)模式之间配置的核心,所述ST模式处理主要线程,且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程,所述计算机程序产品包括:

具有程序指令的计算机可读存储介质,其中所述计算机可读存储介质并非信号,所述程序指令能由处理电路读取以使所述处理电路执行一种方法,所述方法包括:

在所述ST模式中使用核心地址值存取所述主要线程;

自所述ST模式切换至所述MT模式;及

在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者,所述扩展后的地址值包括与线程地址值串接的所述核心地址值,其中所述扩展后的地址值包括移位后的核心地址值,所述移位后的核心地址值被移位基于所请求最大线程识别符的量。

15.如权利要求14所述的计算机程序产品,其中所述线程地址值被串接至所述核心地址值的低阶位以形成所述扩展后的地址值。

16.如权利要求14所述的计算机程序产品,其中所述方法进一步包括在所述MT模式与所述ST模式之间切换,且进一步其中基于所述核心处于相应ST模式或MT模式中而选择所述核心地址值或所述扩展后的地址值中的一者。

17.如权利要求16所述的计算机程序产品,其中在所述ST模式中使用标准格式地址,且所述核心基于停用所述MT模式而自所述MT模式恢复至所述ST模式,且基于停用所述MT模式,仅所述主要线程为可存取的且所述一个或多个次要线程并非可存取的。

多线程计算机系统中的地址扩展及缩短

技术领域

[0001] 本发明一般地涉及一种支持多个线程的计算机系统，且更具体言之，涉及多线程计算机系统中的地址扩展及缩短。

背景技术

[0002] 当计算机系统的处理器速度在过去数十年内增加时，可存取此类计算机系统的存储器所藉以的速度未成比例增加。因此，处理器的周期时间愈快，等待数据自存储器中取回的延迟愈明显。藉由各种级别的缓存且在最新处理器中藉由多线程(MT)减轻此类延迟的影响。

[0003] MT允许处理器的各种核心资源由多个指令流(称为线程)共享。核心资源可包括执行单元、高速缓存、转换后备缓冲器(TLB)及其类似者，其可一般统称为核心。在由高速缓存未命中所产生的延时或一个线程中的其他延迟期间，一个或多个其他线程可利用核心资源，因此增加核心资源的利用率。在超标量处理器同时多线程(SMT)实施中，多个线程可同时由一个或多个核心的核心资源来服务。

[0004] 在现代硬件平台中，MT通常以对在MT硬件上运行的操作系统(OS)透明的方式实施。此特性的一个方面为OS不需要修改以利用MT硬件。然而，相对于OS的透明MT操作可导致响应时间、容量供应、容量规划及计费的高可变性。此可变性可出现是因为OS不感知其任务是否具有核心的独占性控制，或其任务是否作为共享核心的线程执行。借助设计，当在核心在使用中时存在高平均线程密度时，可获得具MT能力的硬件上的存储器密集工作负荷的最高容量。额外容量可归因于由MT提供的增加的高速缓存利用。如果OS不一致地维持所利用核心的高平均线程密度，则由MT提供的额外总吞吐量容量将不可用。举例而言，如果当存在低计算利用率时硬件每核心执行单一MT线程且当存在高计算利用率时以高线程密度执行，则其可非常难以判定多少总MT计算容量可用于工作负荷。MT线程利用的此硬件可变性可以以类似于先前关于容量所描述的方式导致事务响应时间及计费两者的可变性。

发明内容

[0005] 各实施例包括一种用于多线程计算机系统中的地址扩展及缩短的系统、方法及计算机程序产品。根据一个方面，一种计算机系统包括配置，所述配置具有能在单线程(ST)模式与多线程(MT)模式之间配置的核心(core)。所述ST模式处理主要线程，且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程。多线程工具被配置为控制对所述配置的利用以执行一种方法，所述方法包括在所述ST模式中使用核心地址值存取所述主要线程及自所述ST模式切换至所述MT模式。在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者，其中所述扩展后的地址值包括与线程地址值串接的所述核心地址值。

[0006] 根据另一方面，提供一种用于配置中的地址调整的计算机实施的方法。所述配置包括能在ST模式与MT模式之间配置的核心，其中所述ST模式处理主要线程，且所述MT模式

处理所述主要线程及所述核心的共享资源上的一个或多个次要线程。在所述ST模式中使用核心地址值存取所述主要线程。执行自所述ST模式至所述MT模式的切换。在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者。所述扩展后的地址值包括与线程地址值串接的所述核心地址值。

[0007] 另一方面包括一种用于配置中的地址调整的计算机程序产品。所述配置包括能在ST模式与MT模式之间配置的核心，其中所述ST模式处理主要线程，且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程。所述计算机程序产品包括具有程序指令的计算机可读存储介质，其中所述计算机可读存储介质并非信号。所述程序指令能由处理电路读取以使所述处理电路执行一种方法，所述方法包括在所述ST模式中使用核心地址值存取所述主要线程，及自所述ST模式切换至所述MT模式。在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者，其中所述扩展后的地址值包括与线程地址值串接的所述核心地址值。

附图说明

[0008] 当本说明书完结时在权利要求中特定地指出且清楚地要求保护被视为实施例的主题。实施例的前述及其他特征及优点自结合附图而进行的以下详细描述显而易见，在附图中：

[0009] 图1A描绘可根据一个实施例实施的计算环境；

[0010] 图1B描绘可根据一个实施例实施的计算环境；

[0011] 图2描绘可根据一个实施例实施的核心的处理电路；

[0012] 图3描绘可根据一个实施例实施的计算环境；

[0013] 图4描绘可根据一个实施例实施的计算环境中的系统管理程序(hypervisor)上下文保持的一个实例；

[0014] 图5描绘根据一个实施例的用于多线程的动态启用的过程流；

[0015] 图6A描绘根据一个实施例的CPU地址扩展过程的一个实例；

[0016] 图6B描绘根据一个实施例的CPU地址缩短过程的一个实例；

[0017] 图7描绘根据一个实施例的用于设定多线程命令(order)的过程流；

[0018] 图8描绘根据一个实施例的存储多线程能力信息的一个实例；

[0019] 图9描绘根据一个实施例的用于确定多线程能力的过程流；

[0020] 图10描绘根据一个实施例的各种线程上下文位置的一个实例；

[0021] 图11描绘根据一个实施例的多线程寄存器保留的一个实例；

[0022] 图12描绘根据一个实施例的用于多线程寄存器保留的过程流；

[0023] 图13描绘根据一个实施例的多线程寄存器恢复的一个实例；

[0024] 图14描绘根据一个实施例的用于多线程寄存器恢复的过程流；及

[0025] 图15描绘根据一个实施例的计算机可读介质。

具体实施方式

[0026] 例示性实施例提供支持单线程及多线程操作模式的计算机系统中的多线程操作。如本文所使用，逻辑线程是指单一指令流及其相关联状态。亦即，在架构级别处，每一逻辑

线程表示独立中央处理单元(CPU)或处理器。在硬件级别处，当分派线程时，线程为与客体状态的维持组合的与逻辑线程相关联的指令流的执行。因此，本文中可互换地使用术语“线程”与“CPU”。

[0027] 在一个例示性实施例中，CPU含有用于指令执行、中断动作、时序功能、初始程序加载及其他机器相关功能的定序及处理工具。CPU定义可映射至多种基础物理实施的逻辑功能。CPU在执行指令时可处理固定长度的二进制整数及浮点数(例如，二进制、十进制及十六进制)、可变长度的十进制整数，及固定或可变长度的逻辑信息。处理可为并行或串行的。处理元素的宽度、移位路径的多重性及执行不同类型算术中的同时性程度可在不影响逻辑结果的情况下在CPU的一个模型与另一模型之间不同。

[0028] CPU执行的指令可包括多个指令类别，诸如通用、十进制、浮点支持(FPS)、二进制浮点(BFP)、十进制浮点(DFP)、十六进制浮点(HFP)、控制及I/O指令。通用指令可用于执行二进制整数算术运算及逻辑、分支及其他非算术运算。十进制指令对十进制格式的数据进行运算。BFP、DFP及HFP指令分别对BFP、DFP及HFP格式的数据进行运算，同时FPS指令独立于格式对浮点数据进行运算或自一种格式转换至另一种格式。特权控制指令及I/O指令可在CPU在监督状态下时被执行，且半特权控制指令可在经历适当授权机制的情况下在问题状态下执行。

[0029] CPU提供可用于程序但在主存储装置中不具有可寻址表示的寄存器。寄存器可包括(例如)当前程序状态字组(PSW)、通用寄存器、浮点寄存器及浮点控制寄存器、向量寄存器、控制寄存器、存取寄存器、前缀寄存器、当天时间(TOD)可编程寄存器及用于时钟比较器及CPU定时器的寄存器。此寄存器集合可被称为CPU的架构式寄存器上下文。在配置中的每一CPU可提供对TOD时钟的存取，TOD时钟可由配置中的所有CPU共享。指令操作码可判定哪一类型的寄存器将用于操作。

[0030] 每一CPU可具有指示其是否提供全套功能及工具(例如，通用CPU)，或是否预期处理特定类型的工作负荷(例如，特殊CPU)的类型属性。主要CPU为通用CPU或具有与在最后初始程序加载(IPL)操作之后开始的CPU(IPL CPU)相同类型的CPU。次要CPU为具有不同于IPL CPU的CPU类型的除通用CPU以外的任何CPU。

[0031] 多线程工具可用于实施支持架构的计算机系统上。多线程工具提供对于多线程的支持以启用共享核心的一组线程(其亦可被称为CPU)。当多线程工具被启用时，核心内的CPU可共享诸如执行单元或高速缓存的某些硬件资源。当核心中的一个CPU等待硬件资源时(通常，在等待存储器存取时)，核心中的其他CPU可利用核心中的共享资源而非使其保持闲置。当已安装及启用多线程工具时，线程与为核心的成员的CPU同义。当多线程工具未被安装或工具被安装但未被启用时，核心包括单一CPU或线程。

[0032] 当多线程工具被安装时，其可通过执行设定多线程信号处理器(SIGP)命令而被启用。在一个例示性实施例中，当多线程工具被启用时，在配置中的CPU的数目增加一倍数，该倍数的值由程序指定最大线程标识(PSMTID)确定。核心中的CPU的数目可比PSMTID多1。对应于此倍数的数个CPU被分组至核心中。配置中的相同CPU类型的每一核心可具有相同数目个CPU。核心内的每一CPU为相同CPU类型；然而，基于模型及CPU类型，核心内的一些CPU可能不可操作。

[0033] 在一个例示性实施例中，控制程序(诸如，操作系统(OS))显式地启用多线程以便

其可由OS管理的配置使用。备选地，系统管理程序可启用多线程，且系统管理程序的客体及其应用可明显受益。应用程序通常不感知多线程是否已被启用。当多线程被启用时，配置中的所有CPU的CPU地址被调整以包括在地址的最左侧位中的核心标识(或核心ID)及在地址的最右侧位中的线程标识(线程ID或TID)。核心ID亦可被称为核心地址值，且TID可被称为线程地址值。核心内的CPU可共享诸如执行单元或较低级别高速缓存的某些硬件工具，因此核心的一个CPU内的执行可影响核心中的其他CPU的性能。

[0034] 为了管理与在单线程与多线程模式之间动态切换配置的一个或多个核心相关联的改变，包括数个支持特征。为维持与不支持多线程的程序的兼容性，单线程模式可为在重设或去激活后的缺省模式。例示性实施例包括用于在自多线程模式转变至单线程模式之后保留来自多线程模式的线程上下文、传送所述线程上下文及恢复所述线程上下文以支持线程上下文的分析及/或恢复的特性。

[0035] 可由例示性实施例实施的计算环境可(例如)基于国际商业机器公司(Armonk,New York)提供的z/Architecture。z/Architecture描述于2012年8月的题为“z/Architecture Principles of Operation”的IBM®公开(IBM公开第SA22-7832-09号)中，该公开以全文引用的方式并入本文中。在一个实例中，基于z/Architecture的计算环境包括由国际商业机器公司(Armonk,New York)提供的eServer zSeries。计算环境可包括(例如)具有具一个或多个核心(例如，处理器核心)的一个或多个分区(例如，逻辑分区)的处理器复合体，及如本文中进一步描述的一个或多个级别的系统管理程序。

[0036] 图1A展示作为支持多线程(MT)的计算环境的一个实例的计算机系统100。在图1A的实例中，计算机系统100包括多个处理器核心102、输入/输出(I/O)子系统104及系统存储器160。I/O子系统104可提供对本领域公知的I/O设备的存取。处理器核心102(本文中亦简称为“核心”)可包括具有支持元件的处理电路。在图1A的实例中，五个核心102被描绘为核心1 110、核心2 120、核心3 130、核心4 140及核心5 150;然而，亦涵盖较大或较少数目个核心102。MT工具103可为核心102中的每一者的硬件组件。在此实例中，核心102中的每一者能够支持至多四个线程。举例而言，核心1 110可支持线程111、112、113及114。核心2 120可支持线程121、122、123及124。核心3 130可支持线程131、132、133及134。核心4 140可支持线程141、142、143及144。核心5 150可支持线程151、152、153及154。应注意，并非每一核心102的所有四个线程均可在任何瞬时可操作。举例而言，在核心3 130中，线程131及132可在线程133及134被允许可操作(以阴影描绘)时可操作。

[0037] 图1A亦描绘计算机系统100的系统存储器160，其中系统存储器160的各部分被分成逻辑分区1(LPAR1)170、LPAR2 180及LPAR3 190。LPAR 170、180、190表示可执行操作系统(诸如，Linux™或IBM®z/OS™、z/VM或zTPF操作系统)的虚拟化计算系统(亦称为配置)。图1A亦展示核心102至LPAR 170、180、190的分派。在此说明中，核心1 110及核心2120专用于由LPAR1 170使用。核心3 130专用于由LPAR2 180使用，且核心5 150专用于由LPAR3 190使用。核心4 140可在LPAR2 180与LPAR3 190之间共享，但在图1A中展示为被指派给LPAR2 180。LPAR3 190展示由分区采用的两种不同类型的核心102的一个实例，其中在此实例中核心4 140允许多个线程可操作，但核心5 150不允许多个线程可操作。在图1A的实例中，LPAR1 170提供用于OS 171以及程序172、173、174及175的处理资源。LPAR2 180提供用于OS 181以及程序182、183及184的处理资源。LPAR4 190提供用于OS 191以及程序192及193的处

理资源。

[0038] 在LPAR中执行的操作系统的控制下,程序在核心的线程上执行。在一个例示性实施例中,个体线程有时仅执行一个程序;然而,被设计为可重入的程序可在多个线程或核心上同时执行。举例而言,LPAR1 170的OS 171的程序172可在核心1 110中的线程111及113上及核心2 120的线程121及124中执行。取决于OS的控制,不同程序可根据分派规则及服务质量协议被分派在相同或不同线程上。

[0039] 各种级别固件亦驻留在系统存储器160中,包括(例如)毫码162及LPAR系统管理程序163。毫码162可体现为固件以支持较低级别系统功能。LPAR系统管理程序163可为(例如)许可内部代码,诸如IBM Processor-Resource/System Manager™(PR/SM™)。LPAR系统管理程序163可建立LPAR 170、180、190且可管理在核心102上的分派。当MT工具103安装在计算机系统100中时,毫码162及LPAR系统管理程序163亦分别含有MT工具支持代码164及165。MT工具支持代码164及165可视为MT工具103的一部分,因为支持MT的逻辑可散布在毫码162、LPAR系统管理程序163及核心102之间。尽管未描绘,但OS 171、181、191中的每一者亦可包括MT工具支持代码以在其相应LPAR 170、180、190中启用及利用MT。

[0040] 图1B展示与图1A相同的计算系统100,除在图1B的计算环境中核心4 140现被指派给LPAR3 190而非LPAR2 180以外。亦注意不同于线程143及144不可操作的图1A,在图1B中,当在核心4 140上分派LPAR3190时所有四个线程141至144当前可操作。LPAR在核心102上的分派及解除分派为动态的,且在其他时间其他LPAR(未图示)可在同一核心102上操作。

[0041] 现转向图2,大体展示根据一个实施例的用于实施处理核心(诸如,图1A及图1B中的核心102中的一者)的处理电路200的方块图。处理电路200为可同时在MT环境中支持一个或多个线程的处理电路的一个实例。图2中所展示的处理电路200包括可将处理电路200耦合至其他处理器及外围设备的系统控制器接口单元202。系统控制器接口单元202亦可将Dcache 204(其读取及存储数据值)、Icache 208(其读取程序指令)及高速缓存接口单元206连接至外部存储器、处理器及其他外围设备。

[0042] Icache 208可结合指令取回单元(IFU)210提供指令流的加载,所述指令取回单元预先取回指令且可包括推测性加载及分支预测能力。可将所取回指令提供至指令解码单元(IDU)212以便解码成指令处理数据。

[0043] IDU 212可将指令提供至发出单元214,所述发出单元可控制指令至各种执行单元(诸如,用于执行一般运算的一个或多个定点单元(FXU)216及用于执行浮点运算的一个或多个浮点单元(FPU)218)的发出。FPU 218可包括二进制浮点单元(BFU)220、十进制浮点单元(DFU)222或任何其他浮点单元。发出单元214亦可经由一个或多个LSU管线耦合至一个或多个加载/存储单元(LSU)228。多个LSU管线被视为用于执行加载及存储以及用于分支的地址产生的执行单元。LSU 228及IFU 210两者可利用转换后备缓冲器(TLB)230以提供用于操作数及指令地址的缓冲转换。

[0044] FXU 216及FPU 218耦合至各种资源,诸如通用寄存器(GPR)224及浮点寄存器(FPR)226。GPR 224及FPR 226通过LSU 228提供用于自Dcache 204加载及存储的数据值的数据值存储。

[0045] 处理电路200亦可包括计数器及/或定时器250以支持基于系统时间的产生及诊断动作。举例而言,计数器及/或定时器250以及各种诊断及量测工具可用于支持当天时间。

[0046] 现转向图3，描绘类似于图1A的计算环境，除在图3中第二级别系统管理程序300是在计算机系统100的LPAR2 180中执行以外。第二级别系统管理程序300(例如，IBM z/VM操作系统)包括MT支持代码301，其类似于由LPAR(第一级别)系统管理程序163提供的MT支持代码165。第二级别系统管理程序300提供对客体操作系统311、321及331分别操作所在的多个虚拟机310、320及330(亦称为配置)的支持。客体操作系统311、321及331可包括(例如)Linux™或IBM®z/OS™、z/VM或z/TPF OS，或可包括诸如IBM交谈式监视器系统(CMS)的客体开发环境。每一客体OS 311、321及331可以或可以不启用多线程，在此状况下第二级别系统管理程序300可负责使用可用于第二级别系统管理程序300操作所在的LPAR2 180的物理处理资源(核心130、140及线程131至134、141至144)分派客体OS 311、321、331及相关联程序312、313、322、323、332及333。各种虚拟机310、320、330的程序312、313、322、323、332、333可在可用于相应客体OS 311、321及331的线程131至134、141至144上执行。客体OS 311、321及331无需包括MT支持代码，这是因为如果第二级别系统管理程序300利用多线程，则客体OS可明显受益于MT。

[0047] 现转向图4，描绘可根据一个实施例实施的计算环境中的系统管理程序上下文保持的一个实例。在图4的实例中，数个支持结构被描绘于图1A及图1B的LPAR系统管理程序163内。举例而言，结构410可支持图1A的LPAR1 170，包括存储用于当前在如图1A中所展示的物理线程111、112、113、114、121、122、123、124上执行的逻辑线程411、412、413、414、421、422、423、424的架构式寄存器上下文(亦即，线程上下文)的状态描述及卫星(satellite)块。在分派这些逻辑线程时，物理线程保持线程的当前架构式寄存器上下文。架构式寄存器上下文在逻辑线程不再被分派时将被保持在状态描述及卫星块中。结构430可支持图1A的LPAR2 180，包括存储用于当前在如图1A中所展示的物理线程131、132、141、142上执行的逻辑线程431、432、441、442的架构式寄存器上下文的状态描述及卫星块。结构450可支持图1A的LPAR3 190，包括存储用于当前在如图1A中所展示的物理线程151上执行的逻辑线程的架构式寄存器上下文的状态描述及卫星块451。结构450亦包括存储用于当前未在物理处理器(如以阴影所展示)上分派的逻辑线程的架构式寄存器上下文的状态描述及卫星块461、462、463及464。亦可藉由LPAR系统管理程序163保留支持未在物理核心上分派的LPAR的其他结构，诸如包括用于逻辑线程的状态描述及卫星结构471、472、473及474的LPAR A(图1A中未描绘)的结构470。其他结构实例包括支持包括用于逻辑线程的状态描述及卫星结构481及482的未经分派LPAR B(图1A中未描绘)的结构480以及用于逻辑线程的状态描述及卫星结构485的未经分派LPAR C(图1A中未描绘)的结构484。

[0048] 尽管数个结构被描绘在图4的实例中，但应理解，额外结构可由LPAR系统管理程序163及计算机系统100中的其他位置支持以管理多线程。举例而言，支持图3的虚拟机310、320、330的多线程的结构可由图3的第二级别系统管理程序300保留。

[0049] 现转向图5，描绘根据一个实施例的用于多线程的动态启用的过程流500。在块502处，主要线程在单线程(ST)模式中执行。在块504处，在ST模式中取回多线程(MT)模式设定指令。在执行如在505处共同地描绘的此指令时，在块506处获得从由MT模式设定指令指定的位置请求的线程的数目。当发出设定MT模式指令时可由参数寄存器指定位置。MT模式设定指令可为包括设定MT命令及与所请求线程的数目相关联的程序指定最大线程id(PSMTID)的信号处理器(SIGP)指令。参看图7在本文中进一步描述与SIGP指令的设定MT命

令相关联的过程的一个实例。

[0050] 继续过程500，在块508处，执行关于所请求线程的数目是否指示多个线程的判定。举例而言，多个线程可藉由大于1的值指示。在0值指示单个线程的实施例中，1或大于1的值可指示多个线程。基于判定所请求线程的数目不指示多个线程，在块510处核心保持在ST模式中，设定MT模式指令的执行完成，且控制返回至块502。基于判定所请求线程的数目指示多个线程，在块512处启用MT模式，且设定MT模式指令的执行完成。在块514处，执行包括主要线程及一个或多个次要线程的多个线程。在块516处，如果不存在重设或去激活，则过程500循环回至块514；否则，在块518处，基于恢复至ST模式的配置的重设或去激活而停用MT模式。作为停用MT模式的一部分，线程(PSMTID)的数目被保留用于非清除重设或被置0用于清除重设。过程500返回至块502。

[0051] 当激活加载正常、加载与转储、加载清除或加载清除清单引导密钥时CPU可进入加载状态。如果通道命令(command)字(CCW)型初始程序加载操作成功地完成，则CPU自加载状态改变至操作状态。

[0052] CPU重设可用于在信息损毁量最少的情况下清除设备检查指示及任何由此产生的CPU状态的不可预测性。详言之，在保留CPU状态以用于分析或重新继续操作时，CPU重设可用于清除检查条件。如果CPU重设由加载正常或加载与转储密钥的激活引起，则(a)CPU重设可将架构模式设定成缺省模式，且(b)如果安装及启用多线程工具，则停用多线程。当CPU重设设定缺省模式时，其可保存当前PSW，使得可恢复PSW。

[0053] 初始CPU重设提供CPU重设连同当前PSW、CPU定时器、时钟比较器及其他寄存器(诸如：中断事件地址、所捕获的PSW、控制、浮点控制、前缀及TOD可编程寄存器)的初始化的功能。如果初始CPU重设由加载正常或加载与转储密钥的激活引起，则其可将架构模式设定为缺省模式。如果在初始CPU重设由加载正常或加载与转储密钥的激活所引起时多线程被启用，则初始CPU重设功能可针对核心的最低编号CPU而执行，且CPU重设针对核心中的所有其他CPU而执行。清除重设引起初始CPU重设及子系统重设被执行，且另外，除TOD时钟以外，清除或初始化配置中的所有CPU中的所有存储位置及寄存器。清除不影响外部存储装置，诸如由控制程序使用以保存不可寻址页面的内容的直接存取存储器件。

[0054] CPU通电重设引起初始CPU重设被执行且将通用寄存器、存取寄存器、控制寄存器及浮点寄存器的内容清除至0/具有有效检查块码的缺省值。应理解，状态的清除或初始化无需为0值但在已清除状态下可缺省为非0值。如果CPU通电重设建立配置，则其可将架构模式设定成缺省模式；否则，其可将架构模式设定成已在配置中的CPU的模式。可手动地发起CPU重设、初始CPU重设、子系统重设及清除重设。

[0055] 在例示性实施例中，每一CPU具有被指派的编号，称为其CPU地址。CPU地址独特地识别配置内的一个CPU。CPU通过在SIGP指令的CPU地址字段中指定此地址而被指定。用信号通知故障警报、紧急信号或外部调用的CPU可在中断情况下由在CPU地址字段中存储此地址而识别。CPU地址由配置定义程序指派且通常并不由于重配置改变而改变。程序可通过使用存储CPU地址指令而确定CPU的地址。存储CPU地址指令亦可用于识别CPU地址，借助该CPU地址在多处理配置中识别CPU。

[0056] 当启用多线程时，CPU地址可包括与核心内的CPU的标识串接的核心标识(核心ID)。核心内的CPU标识为线程标识(线程ID或TID)。在配置内，所有核心提供相同数目个

CPU;然而,取决于模型及CPU类型,核心中的一些CPU可能不可操作。

[0057] 基于由信号处理器设定多线程命令使用的参数寄存器的PSMTID,固定数目个位表示线程标识。此位数目被称为TID宽度。

[0058] 在启用多线程之前,核心ID可由CPU地址的最右侧位形成。核心ID左移TID宽度个位,从而在多线程可用之后产生CPU地址的最左侧位。线程ID具有相同TID宽度数目个位,且在多线程被启用之后占据CPU地址的最右侧位。可在连续数目范围中指派线程ID。表1例示PSMTID的实例关系,TID宽度及CPU地址位包括核心标识及线程标识。

[0059]

PSMTID	TID 宽度	CPU 地址位	
		核心 ID	线程 ID
0	0	0-15	-
1	1	0-14	15
2-3	2	0-13	14-15
4-7	3	0-12	13-15
8-15	4	0-11	12-15
16-31	5	0-10	11-15

[0060] 表1-实例地址位映射

[0061] 在图6A中将地址扩展描绘为根据一个实施例的CPU地址扩展过程600A的一个实例。在块602处,可使用核心地址值604作为数个CPU地址位在ST模式中存取主要线程。箭头606指示自ST模式切换至MT模式。在块608处,可使用扩展后的地址值610在MT模式中存取主要线程或一个或多个次要线程。扩展后的地址值610包括被移位为移位后的核心地址值612且与线程地址值614串接的核心地址值604。移位后的核心地址值612为核心识别符(核心ID),且线程地址值614为线程识别符(TID)。移位后的核心地址值612可基于所请求的最大线程识别符(例如,PSMTID)而移位一定量。线程地址值614中的TID位的数目可基于如上文表1中所展示的PSMTID而确定。线程地址值614可串接至移位后的核心地址值612的低阶位以形成扩展后的地址值610。全0的线程地址值614将指明主要线程,且大于0的值识别及处理次要线程。

[0062] 当在MT模式与ST模式之间切换时,核心地址值604(ST模式)或扩展后的地址值610(MT模式)被选择以在相应ST模式或MT模式中用作CPU地址。核心地址值604为用于ST模式中的标准格式地址的一个实例,且核心基于停用MT模式而自MT模式恢复至ST模式。在一个例示性实施例中,仅主要线程(亦即,非次要线程)可基于停用MT模式而存取。图6B描绘根据一个实施例的CPU地址缩短过程600B的一个实例。图6B的箭头616说明自块608的MT模式切换

回至块602的ST模式。自MT模式至ST模式的恢复可包括将扩展后的地址值610向右移位及去除线程地址值614以从移位后的核心地址值612形成包括核心地址值604(核心ID)的标准格式地址作为CPU地址。

[0063] 当重设功能停用多线程时，(a)使线程ID为0的CPU的CPU地址向右移位在启用期间使用的相同TID宽度数目个位，(b)0在地址左侧被插入TID宽度数目个位中，且(c)所述CPU地址恢复至其原始非多线程格式(亦即，标准格式地址)。当启用多线程时在具有非0线程ID的核心中的所有CPU在停用多线程时不再为可操作的。

[0064] 当不启用多线程时，CPU地址保持由配置定义过程指派的值而不变。在此状况下，线程标识不存在。

[0065] 数个信号处理器命令可将包括(例如)以下各者的命令提供至CPU：开始、停止、重新开始、停止及存储状态、初始CPU重设、CPU重设、在地址处的存储状态、设定架构、感测运行状态、设定多线程、在地址处的存储额外状态及其类似者。初始CPU重设或CPU重设可由信号处理器指令发起且不影响架构模式或其他CPU不停用多线程，且不引起I/O重设。

[0066] 设定架构命令指定配置中的所有CPU待设定为的架构模式。架构差异可包括不同寻址模式、寄存器定义及由CPU支持的指令。在架构模式改变后，可将寄存器的选择位字段设定成缺省状态(例如，置0)，清除配置中的所有CPU的存取寄存器转换后备缓冲器(ALB)及转换后备缓冲器(TLB)，且可在配置中的所有CPU上执行序列化及检查点同步功能。

[0067] 感测运行状态命令可指示被寻址的CPU是否正在运行。在ST模式中，可返回指示符作为运行/非运行状态。在MT模式中，指示符可用于识别被寻址CPU为成员的核心的任何CPU是正在运行，还是被寻址CPU为成员的核心的所有CPU未正在运行。

[0068] 设定MT命令启用多线程工具。参数寄存器的位位置可含有待提供于配置中的PSMTID。PSMTID可被定义为比可待在每一核心中寻址的CPU的数目小1。举例而言，所指明位位置中的值3指示待提供最多四个线程。当配置中的所有CPU被视为待寻址时，可忽略SIGP指令的CPU地址寄存器的内容。如果接受，则在SIGP指令的执行期间由所有CPU完成设定MT命令。参看图7，描绘用于SIGP设定MT命令702的过程700。可提供错误指示且基于判定由以下各者中的一者或多者发出SIGP设定MT命令702而防止MT模式的启用：无效命令、不正确状态及无效参数，如本文中关于图7的过程700进一步描述的。

[0069] 如果在块704处多线程工具未安装或CPU并未在有效架构模式中启用(708)，则不接受设定MT命令且可分别在块706或710处返回无效命令指示。如果在块712处在配置中的其他CPU不在停止或检查停止状态下，或如果在块716处所述配置已被启用以用于多线程，则不接受设定MT命令且可分别在块714或718处返回不正确状态指示。

[0070] 如果在块720处PSMTID为无效的，则不接受设定MT命令且在块722处可返回无效参数指示。当在块724处PSMTID为0时，配置未被启用以用于多线程，保持处于ST模式中，且在块728处提供任何状态作为条件代码。在一个例示性实施例中，当PSMTID有效且非0时，在块726处，配置被启用以用于多线程，导致CPU地址扩展，配置中的所有CPU的ALB及TLB已清除其内容，且在配置中的所有CPU上执行序列化及检查点同步功能。可在块728处在条件代码中提供状态。在成功完成后，除执行设定MT命令的CPU以外的所有CPU保持在停止或检查停止状态下。然而，如果在启用多线程之前CPU在检查停止状态下，则同一核心中具有非0线程ID的CPU置于停止状态还是检查停止状态下可能是不可预测的。

[0071] 线程上下文亦可被称为架构式寄存器上下文。在启用多线程之前每一CPU的架构式寄存器上下文(亦即,PSW、CPU定时器、时钟比较器、通用寄存器、浮点寄存器及浮点控制寄存器、向量寄存器、控制寄存器、存取寄存器、前缀寄存器及TOD可编程寄存器等的内容)在启用多线程之后变为每一相应核心的具有TID 0的CPU的架构式寄存器上下文。类似地,当由于加载正常或加载与转储密钥的激活而停用多线程时,具MT能力的配置的每一核心的具有TID 0的CPU的架构式寄存器上下文变为每一相应CPU的架构式寄存器上下文。

[0072] 当由于加载正常或加载与转储密钥操作的激活而停用多线程工具时,可保留具有非0线程标识的所有CPU的架构式寄存器上下文。如果在无介入清除重设的情况下随后重新启用多线程工具,则恢复具有非0线程标识的所有CPU的架构式寄存器上下文。

[0073] 当在已由加载正常或加载与转储密钥的激活停用之后重新启用多线程时,如果参数寄存器的位中的PSMTID的值不同于前述启用中使用的值,则具有非0线程ID的所有CPU的架构式寄存器上下文可为不可预测的。

[0074] 存储系统信息指令可用于将关于配置的一个或多个组件的信息存储至系统信息块(SYSIB)中。SYSIB可包括MT安装字段、MT通用字段、总CPU/核心计数、被配置CPU/核心计数、待用CPU/核心计数、保留CPU/核心计数及其他字段。MT安装字段可指示是否安装多线程工具且亦可指示用于第一核心类型(例如,特殊核心类型)的最高支持TID。MT通用字段可指示用于第二核心类型(例如,通用核心类型)的最高支持TID。MT通用字段中的最高支持TID可限于小于或等于MT安装字段中的最高支持TID。总CPU/核心计数可指示配置中的通用CPU(不论在被配置、待用或保留状态下)或包括通用CPU的核心的总数。被配置CPU/核心计数可指示在被配置状态下(亦即,在配置及准备执行程序中)的通用CPU或包括通用CPU的核心的数目。待用CPU/核心计数指示在待用状态下(亦即,不可用于执行程序直至置于被配置状态下)的通用CPU或包括通用CPU的核心的数目。保留CPU/核心计数指示在保留状态下(亦即,不可用于执行程序且不能置于被配置状态下)的通用CPU或包括通用CPU的核心的数目。

[0075] 图8描绘根据一个实施例的存储多线程能力信息的一个实例。在线程(诸如,核心800A的线程1)中执行的程序可自配置850的诸如LPAR的存储器801取回存储系统信息(STSI)指令830。STSI指令的执行可导致系统信息块(SYSIB)802的存储(832)。在图8的实例中,SYSIB 802包括指示配置850是否支持多线程的MT安装识别符804。SYSIB 802亦包括可作为用于特殊核心的每核心最大TID 806及用于通用核心的最大TID 808提供的核心800A/800B的最高支持线程的最大线程识别符。SYSIB 802亦可包括当前程序指定最大线程识别符(PSMTID)809。当前PSMTID 809反映如由程序在配置850中启用的多线程模式。如果STSI指令830在基本机器级别处执行,则可不定义当前PSMTID 809。

[0076] 在线程(诸如核心800B的线程2)中执行的程序亦可自配置850的存储器801取回服务调用(SERVC)指令834,其中指令指定读取系统控制程序信息(读取SCP信息,或RSCPI)命令。RSCPI命令的执行可引起服务调用控制块(SCCB)810存储(836)于存储器801中。在一个例示性实施例中,通过RSCPI命令的执行而存储的SCCB 810提供可不可用于SYSIB 802中的类似及额外信息。在图8的实例中,SCCB 810包括指示核心800B是否支持多线程的MT安装识别符812。SCCB 810亦包括可被提供为用于特殊核心的每核心最大TID 814及用于通用核心的最大TID 816的核心800B的最高支持线程的最大线程识别符。SCCB 810的812至816的值等于可在SYSIB 802中可存取的值804至808。另外,SCCB 810可包括核心800B的最高支持线

程的最后设定程序指定最大线程识别符，其亦被称为最后设定程序指定最大线程识别符(PSMTID)818。SCCB 810亦可包括作为PSMTID支持屏蔽820的关于设定MT命令可接受的PSMTID值的屏蔽。当小于由每核心最大TID 814定义的数目为所期望的时，PSMTID支持屏蔽820可用以识别所支持CPU/线程。

[0077] 应理解，核心800A及800B包括未在此实例中描绘的其他方面。此外，SYSIB 802及SCCB 810可包括超出在图8的实例中描绘的那些值的额外值。

[0078] 图9描绘根据一个实施例的用于确定多线程能力的过程流900。在块902处，核心执行取回多线程能力信息(RMTCI)指令，其可(例如)为SERVC指令或STSI指令中的任一者。在块904处，获得识别配置的多线程能力的线程识别信息。在块906处，存储所获得的线程识别信息。在块908处，基于所获得的线程识别信息判定配置先前是否已使多线程被启用。

[0079] 如先前所描述，SERVC指令被配置为将线程识别信息存储在存储器中的响应块(例如，图8的SCCB 810)中，且STSI指令被配置为将线程识别信息存储在存储器中的SYSIB(例如，图8的SYSIB 802)中。所获得的线程信息可包括指示核心是否支持多线程的MT安装识别符(例如，图8的MT安装识别符804或812)。所获得的线程信息亦可包括核心的最高支持线程的最大线程识别符(例如，图8的最大TID值806、808、814或816)。所获得的线程信息可包括当前程序指定最大线程识别符(例如，图8的当前PSMTID 809)及最后设定程序指定最大线程识别符(例如，图8的PSMTID 818)。响应块可包括指示个体地支持的特定线程识别符的位掩码(例如，图8的PSMTID支持屏蔽820)。配置先前已使MT被启用的判定可基于最后设定程序指定最大线程识别符中的非0值(例如，最后设定PSMTID>0)。在一个例示性实施例中，配置支持多个核心类型。

[0080] 在例示性实施例中，寄存器及诸如程序计数器值的值(其可包括于寄存器中或单独地管理)被捕获为线程上下文。当在MT模式中发生地址扩展时，额外线程上下文变为可存取。如先前参看图6所描述，针对配置中的每一核心形成CPU地址。CPU地址可由存储CPU地址指令检验，所述指令出现在其他结构中且用于各种SIGP命令中。当MT未被启用时，此寻址方案保持不变。当MT被启用时，CPU地址经历扩展程序。如先前所描述，CPU地址的非MT启用部分可向左移位足够的位以容纳TID。举例而言，如果操作系统发出具有PSMTID值1的SIGP设定MT命令，则CPU地址将向左移位1位；如果PSMTID为2或3，则CPU地址将向左移位2位，如果PSMTID为4至7，则CPU地址将向左移位3位，等等。

[0081] 当随后停用多线程(由于由加载正常操作引起的清除重设或CPU重设)时，CPU地址缩短发生。具MT能力的CPU地址可向右移位用于启用MT的SIGP设定MT命令中的相同数目个PSMTID位，且地址的线程ID部分消失。可在MT模式期间存取的线程上下文可驻留在一个或多个位置中，诸如图10中描绘的实例。在图10的实例中，配置1000包括核心1002且可包括其他核心(未描绘)。存储器1006可包括作为配置1000的一部分的配置存储器1005及与配置1000分离的主机/固件存储器1007。主机/固件存储器1007可包括由主机维护的状态描述块1008，其可存储用于线程(例如，图10中的线程n)的线程上下文1010。卫星块1012可被锚定至作为主机/固件存储器1007的一部分的存储器1006中的状态描述块1008，其中卫星块1012可包括作为线程上下文1010的替代或结合线程上下文1010的线程上下文1014。每一线程可具有对应状态描述块1008及(可选地)卫星块1012，其中可存储线程上下文1010或线程上下文1014。作为另一替代例，硬件上下文寄存器1016可用于将线程上下文1018存储在(例

如)核心1002中。线程上下文1010、1014及1018的实例可组合地或单独地用作存储选项。可在实施例中使用替代存储选项。与在何处维护线程上下文无关,在地址缩短后,线程上下文可不再直接可存取,但可由转储程序而保留以用于存取。

[0082] 当MT被停用时,CPU地址缩短过程使核心的线程1至n不再可寻址;类似地,包括架构式寄存器的线程上下文不再为程序可见。如果MT由于由非清除加载操作产生的CPU重设而停用,则保留线程1至n的寄存器上下文;如果配置返回至MT模式,则可随后检验这些数据。用于每一客体线程的寄存器上下文可由主机维护于如图10中所描绘的线程状态描述块1008中(或如在向量寄存器的状况下,维护于锚定在状态描述中的卫星块1012中)。

[0083] MT停用期间的线程1至n的上下文的保持为用于在OS失败之后待转储的线程的状态的诊断特性。在OS失败之后,操作者可选择运行独立转储(SADMP)程序以在失败时捕获系统的存储器及线程上下文。然而,加载SADMP程序可引起配置恢复至启用ST模式的缺省架构模式,因此停用MT。但因为SADMP由非清除加载操作加载,所以保留每一核心的线程1至n的寄存器上下文。SADMP可通过查验SERVC读取SCP信息命令的响应块的结果而判定MT是否在被转储的配置中被启用。此数目可随后用作至SIGP设定MT命令的输入以在与前述相同的级别处重新启用MT。

[0084] 图11描绘根据一个实施例的多线程寄存器保留的一个实例。诸如图11的计算机系统1100的系统可包括多个配置1102及1104。在图11的实例中,配置1102包括核心1106及核心1108,且配置1104包括核心1110及核心1112。配置1102及1104中的每一者可在不同时间独立地在ST模式与MT模式之间切换。计算机系统1100的配置1102及1104中的每一者可配置有不同数目个最大线程ID以在配置1102及1104中的每一者处支持同时启用的不同数目个线程。在图11的实例中,核心1106及1108均支持最多两个线程,同时配置1102在MT模式1114下,而核心1110及1112均支持最多四个线程,同时配置1104在MT模式1116下。

[0085] 当在配置1102中启用MT模式1114时,TID 0及TID 1两者可存取为单独线程上下文,诸如线程上下文1115的单独实例。在时间1118处,可通过配置1102的加载正常操作或非清除重设而停用MT模式1114,其将核心1106及1108切换至ST模式1120中。归因于如先前所描述的地址缩短,可在ST模式1120下存取TID0寄存器;然而,可在MT模式1114下存取的TID1寄存器被保留但不再可存取。举例而言,TID1寄存器可体现为图10的线程上下文1010、1014或1018,其中在切换至ST模式1120后借助地址扩展可用的地址在地址缩短之后不再可存取。

[0086] 在配置1104使MT模式1116被启用时,TID0、TID1、TID2及TID3寄存器可存取为独立线程上下文,诸如图10的线程上下文1010、1014或1018的单独实例。在此实例中,TID0表示主要线程且TID1至TID3表示针对核心1110及核心1112中的每一者单独地维护的次要线程。在时间1122处,可针对配置1104借助清除重设而停用MT模式1116,其将核心1110及1112两者切换至ST模式1124中。在时间1122处的清除重设可清除所有寄存器TID0、TID1、TID2及TID3。归因于如先前所描述的地址缩短,可在ST模式1124下存取TID0寄存器;然而,可在MT模式1116下存取的TID1、TID2及TID3寄存器被保留于已清除状态下但不再可存取。如图11中所描绘,可在不同时间1118及1122处在每一配置1102及1104上以区域化至每一配置1102及1104的效果独立地执行操作。因此,配置1102可在ST模式1120下而配置1104在MT模式1116下,且ST/MT模式无需针对计算机系统1100的所有配置而对准。

[0087] 图12描绘根据一个实施例的用于多线程寄存器保留的过程流1200。在块1202处，基于由MT模式中的核心判定待在核心中停用MT，执行自MT模式至ST模式的切换。MT模式的主要线程可被维护为ST模式的仅有线程。使包括次要线程的程序可存取寄存器值及程序计数器值的一个或多个线程上下文不可由应用程序存取。在块1204处，基于切换，判定操作类型(例如，清除对非清除)以清除程序可存取寄存器值抑或保留程序可存取寄存器值。在块1206处，基于非清除操作，判定保留程序可存取寄存器值。在块1208处，基于清除操作，判定清除程序可存取寄存器。

[0088] 如先前所描述，线程上下文的程序可存取寄存器值及程序计数器值可包括程序通用寄存器、浮点寄存器、控制寄存器、存取寄存器、前缀寄存器及TOD可编程寄存器。控制寄存器可包括浮点控制寄存器、运行时插装控件、CPU测量控件及其类似者。可包括于线程上下文中的寄存器的其他实例包括程序状态字(例如，包括程序计数器/指令地址、条件代码及用以控制指令定序及判定CPU状态的其他信息)、向量寄存器、CPU定时器、时钟比较器、中断事件地址寄存器及本领域公知的其他寄存器。如先前所描述，PSMTID基于引起启用MT的最后成功地执行的信号处理器指令而设定。基于切换至MT模式，基于对应次要线程被重新启用而使程序可存取寄存器值不可由应用程序存取。举例而言，在图11中自ST模式1120切换回至MT模式1114允许TID1寄存器被存取，且TID1可被重新启用。线程上下文可被维护在以下各者中的任一者中：状态描述块、锚定至存储器中的状态描述块的卫星块，或上下文寄存器，诸如图10的线程上下文1010、1014或1018。

[0089] 主要线程上下文可包括主要线程的程序可存取寄存器值及程序计数器值，例如，用于图11的配置1104的TID0及TID0寄存器，其中主要线程上下文在ST模式1124及MT模式1116两者下可由应用程序存取。次要线程上下文可包括次要线程的程序可存取寄存器值及程序计数器值，例如，用于图11的配置1104的TID1至TID3及TID1至TID3寄存器。

[0090] 图13描绘根据一个实施例的多线程寄存器恢复的一个实例。图13的实例包括具有单一配置1302的计算机系统1300。配置1302包括核心1304、核心1306及核心1308。在此实例中，核心1304至1308中的每一者包括最多四个线程(TID0、TID1、TID2及TID3)。在MT模式1310下，TID0至TID3的所有线程上下文可用于核心1304至1308中。在时间1312处，可通过配置1302的加载正常操作或非清除重设而停用MT模式1310，其将核心1304至1308切换至ST模式1314中。在ST模式1314下，TID0寄存器保持可存取，且TID1至TID3寄存器不可存取的但针对核心1304至1308中的每一者保留。在时间1316处，可通过SIGP设定MT命令的执行而重新启用MT以进入被重新继续的MT模式1318。在被重新继续的MT模式1318下，针对核心1304至1308中的每一者恢复对TID1至TID3寄存器的线程上下文的存取。此通过转储程序(诸如，独立转储程序1320)实现对所有线程寄存器(包括TID1至TID3寄存器)的检验以保存线程上下文信息以供分析。

[0091] 图14描绘如可由独立转储(SADMP)程序(诸如，图13的独立转储程序1320)使用以在操作系统失败之后捕获线程的架构式寄存器上下文的根据一个实施例的用于多线程寄存器恢复的过程流1400。在块1405处，经由非清除加载操作(例如，加载正常或加载与转储)加载SADMP程序。非清除加载操作隐含地引起配置恢复至ST模式，诸如用于图13的配置1302的ST模式1314。SADMP程序可接着在块1410处通过使用STSI或SERVC指令查询MT工具是否可用于配置中。如果已安装MT，则在块1415处SADMP程序查询针对配置设定的最后设定程序指

定最大线程标识(PSMTID)。如果先前从未针对配置设定MT,则最后设定PSMTID值将为0。SADMP程序可在最后设定PSMTID无论为何值(即使其为0)的情况下在块1420处接着执行指令以重新启用多线程。如果在块1410处查询揭露未安装MT,则不尝试在块1415处查询最后设定PSMTID值或在块1420处重新启用MT。

[0092] SADMP程序尝试用信号通知配置中的每一其他CPU(线程)将其架构式寄存器上下文保存在存储器中的预定义位置中。如果在加载SADMP之前先前未启用MT,则CPU地址为正常的非扩展格式。如果先前启用MT,则CPU地址为包括核心ID及线程ID的扩展格式。SADMP在块1425处以为0的CPU地址(N)开始,且在块1430处判定CPU地址是否表示执行SADMP所在的CPU。如果是,则跳过该CPU/线程,且在块1450处将N递增至下一值。如果N不同于当前CPU地址,则在块1435处用信号通知该CPU/线程(例如)通过SIGP在地址处存储状态命令或SIGP停止及存储状态命令的执行将其架构式寄存器上下文存储在存储器中。如果配置包括向量工具,则在SIGP在地址处存储额外状态命令亦可被执行以存储CPU/线程的向量寄存器的内容。在块1440处作出关于块1435的发信号是否成功的判定。如果成功,则在块1445处SADMP程序可将CPU/线程的寄存器上下文存储在磁带或磁盘上的转储文件中,且在块1450处处理通过递增N而继续。如果块1435的发信号如由块1440判定为未成功(例如,如果线程不可操作),则其被跳过,且在块1450处处理通过递增N而继续。在块1450处使发信号时所使用的CPU地址(N)的值递增,且在块1455处作出关于N现是否大于用于配置的最高可能CPU地址的判定。如果N不大于用于配置的最高可能CPU地址,则在块1430处处理通过判定N是否表示执行SADMP程序所借助的当前CPU/线程而继续。如果N大于用于配置的最高可能CPU地址,则在块1460处已完成架构式寄存器上下文恢复及转储。

[0093] 尽管针对配置的一个核心而描述图14,但应理解,图14的过程流1400可延伸以跨越包括多个核心的配置的所有核心遍历最大CPU地址而运行。可在配置中进行额外调节以支持并不支持MT的OS或感知MT但不采用MT的程序的转储。举例而言,可在加载不支持配置中的MT的OS之前执行清除重设以防止MT感知独立转储程序尝试自配置转储任何次要线程。作为另一实例,感知MT但不采用MT的程序可在执行用于配置的独立转储程序之前发出具有为0的对应最大线程id的设定MT命令。

[0094] 技术效果及益处包括支持单线程操作模式及多线程操作模式两者的多线程计算机系统中的地址扩展及缩短。

[0095] 本文中描述的系统使软件能够通过要求OS显式地“加入宣告(opt in)”以利用MT硬件而减轻硬件可变性。当OS理解执行环境的MT性质时,OS具有显式地管理每核心线程密度的能力(对于其最好能力,给定工作负载分派模式)。OS具有如下选项:即使当较少地利用计算资源时仍维持高线程密度,由此减轻在其他MT实施上看到的总技术容量的许多可变性。作为维持高线程密度的直接结果,事务响应时间及计费方面两者可更一致。

[0096] 各实施例包括一种用于多线程计算机系统中的地址扩展及缩短的系统、方法及计算机程序产品。根据一个方面,一种计算机系统包括配置,所述配置具有能在单线程(ST)模式与多线程(MT)模式之间配置的核心。所述ST模式处理主要线程,且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程。多线程工具被配置为控制对所述配置的利用以执行一种方法,所述方法包括在所述ST模式中使用核心地址值存取所述主要线程及自所述ST模式切换至所述MT模式。在所述MT模式中使用扩展后的地址值存取所述

主要线程或所述一个或多个次要线程中的一者，其中所述扩展后的地址值包括与线程地址值串接的所述核心地址值。

[0097] 根据另一方面，提供一种用于配置中的地址调整的计算机实施的方法。所述配置包括能在ST模式与MT模式之间配置的核心，其中所述ST模式处理主要线程，且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程。在所述ST模式中使用核心地址值存取所述主要线程。执行自所述ST模式至所述MT模式的切换。在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者。所述扩展后的地址值包括与线程地址值串接的所述核心地址值。

[0098] 另一方面包括一种用于配置中的地址调整的计算机程序产品。所述配置包括能在ST模式与MT模式之间配置的核心，其中所述ST模式处理主要线程，且所述MT模式处理所述主要线程及所述核心的共享资源上的一个或多个次要线程。所述计算机程序产品包括具有程序指令的计算机可读存储介质，其中所述计算机可读存储介质并非信号。所述程序指令能由处理电路读取以使所述处理电路执行一种方法，所述方法包括在所述ST模式中使用核心地址值存取所述主要线程，及自所述ST模式切换至所述MT模式。在所述MT模式中使用扩展后的地址值存取所述主要线程或所述一个或多个次要线程中的一者，其中所述扩展后的地址值包括与线程地址值串接的所述核心地址值。

[0099] 除上文所描述的特征中的一者或多者的外，或作为替代例，其他实施例可包括其中所述扩展后的地址值包括移位后的核心地址值，所述移位后的核心地址值被移位基于所请求最大线程识别符的量。

[0100] 除上文所描述的特征中的一者或多者的外，或作为替代例，其他实施例可包括其中所述线程地址值被串接至所述核心地址值的低阶位以形成所述扩展后的地址值。

[0101] 除上文所描述的特征中的一者或多者的外，或作为替代例，其他实施例可包括在所述MT模式与所述ST模式之间切换，且进一步其中基于所述核心处于相应ST模式或MT模式中而选择所述核心地址值或所述扩展后的地址值中的一者。

[0102] 除上文所描述的特征中的一者或多者的外，或作为替代例，其他实施例可包括其中在所述ST模式中使用标准格式地址，且所述核心基于停用所述MT模式而自所述MT模式恢复至所述ST模式。

[0103] 除上文所描述的特征中的一者或多者的外，或作为替代例，其他实施例可包括其中基于停用所述MT模式，仅所述主要线程为可存取的且所述一个或多个次要线程并非可存取的。

[0104] 除上文所描述的特征中的一者或多者的外，或作为替代例，其他实施例可包括其中自所述MT模式至所述ST模式的恢复进一步包括使所述扩展后的地址值移位及去除所述线程地址值。

[0105] 除上文所描述的特征中的一者或多者的外，或作为替代例，其他实施例可包括其中基于程序指定最大线程识别符确定所述线程地址值中的线程识别符位的数目。

[0106] 本文中所用的术语，仅仅是为了描述特定的实施例，而不意图限定本发明。本文中所用的单数形式的“一”和“该”，旨在也包括复数形式，除非上下文中明确地另行指出。还要知道，“包含”和/或“包括”在本说明书中使用时，说明存在所指出的特征、整体、步骤、操作、单元和/或组件，但是并不排除存在或增加一个或多个其它特征、整体、步骤、操作、单元和/

或组件，以及/或者它们的组合。

[0107] 以下的权利要求中的对应结构、材料、操作以及所有功能性限定的装置(means)或步骤的等同替换，旨在包括任何用于与在权利要求中具体指出的其它单元相组合地执行该功能的结构、材料或操作。所给出的对一个或多个实施例的描述其目的在于示意和描述，并非是穷尽性的，也并非是要将本发明限定到所表述的形式。对于所属技术领域的普通技术人员来说，显然可以作出许多修改和变型而不偏离本发明的精神和范围。对实施例的选择和描述，是为了最好地解释本发明的原理和实际应用，使所属技术领域的普通技术人员能够明了，本发明可以有适合所要的特定用途的具有各种改变的各种实施例。

[0108] 本发明的各种实施例的描述已为达成说明的目的而呈现，但不意欲为穷尽性的或限于所揭示的实施例。在不脱离所描述实施例的范围及精神的情况下，许多修改及变化对于本领域技术人员将显而易见。本文中所使用的术语被选择以最好地解释实施例的原理、实际应用或对市场中找到的技术的技术改进，或使其他本领域技术人员能够理解本文所揭示的实施例。

[0109] 现参看图15，总体上展示根据一个实施例的包括计算机可读存储介质1502及程序指令1504的计算机程序产品1500。

[0110] 本发明可以是系统、方法和/或计算机程序产品。计算机程序产品可以包括计算机可读存储介质，其上载有用于使处理器实现本发明的各个方面的计算机可读程序指令。

[0111] 计算机可读存储介质可以是可以保持和存储由指令执行设备使用的指令的有形设备。计算机可读存储介质例如可以是但不限于电存储设备、磁存储设备、光存储设备、电磁存储设备、半导体存储设备或者上述的任意合适的组合。计算机可读存储介质的更具体的例子(非穷举的列表)包括：便携式计算机盘、硬盘、随机存取存储器(RAM)、只读存储器(ROM)、可擦式可编程只读存储器(EPROM或闪存)、静态随机存取存储器(SRAM)、便携式压缩盘只读存储器(CD-ROM)、数字多功能盘(DVD)、记忆棒、软盘、机械编码设备、例如其上存储有指令的打孔卡或凹槽内凸起结构、以及上述的任意合适的组合。这里所使用的计算机可读存储介质不被解释为瞬时信号本身，诸如无线电波或者其他自由传播的电磁波、通过波导或其他传输媒介传播的电磁波(例如，通过光纤电缆的光脉冲)、或者通过电线传输的电信号。

[0112] 这里所描述的计算机可读程序指令可以从计算机可读存储介质下载到各个计算/处理设备，或者通过网络、例如因特网、局域网、广域网和/或无线网下载到外部计算机或外部存储设备。网络可以包括铜传输电缆、光纤传输、无线传输、路由器、防火墙、交换机、网关计算机和/或边缘服务器。每个计算/处理设备中的网络适配卡或者网络接口从网络接收计算机可读程序指令，并转发该计算机可读程序指令，以供存储在各个计算/处理设备中的计算机可读存储介质中。

[0113] 用于执行本发明操作的计算机程序指令可以是汇编指令、指令集架构(ISA)指令、机器指令、机器相关指令、微代码、固件指令、状态设置数据、或者以一种或多种编程语言的任意组合编写的源代码或目标代码，所述编程语言包括面向对象的编程语言—诸如Smalltalk、C++等，以及常规的过程式编程语言—诸如“C”语言或类似的编程语言。计算机可读程序指令可以完全地在用户计算机上执行、部分地在用户计算机上执行、作为一个独立的软件包执行、部分在用户计算机上部分在远程计算机上执行、或者完全在远程计算机

或服务器上执行。在涉及远程计算机的情形中，远程计算机可以通过任意种类的网络—包括局域网(LAN)或广域网(WAN)—连接到用户计算机，或者，可以连接到外部计算机(例如利用因特网服务提供商来通过因特网连接)。在一些实施例中，通过利用计算机可读程序指令的状态信息来个性化定制电子电路，例如可编程逻辑电路、现场可编程门阵列(FPGA)或可编程逻辑阵列(PLA)，该电子电路可以执行计算机可读程序指令，从而实现本发明的各个方面。

[0114] 这里参照根据本发明实施例的方法、装置(系统)和计算机程序产品的流程图和/或框图描述了本发明的各个方面。应当理解，流程图和/或框图的每个方框以及流程图和/或框图中各方框的组合，都可以由计算机可读程序指令实现。

[0115] 这些计算机可读程序指令可以提供给通用计算机、专用计算机或其它可编程数据处理装置的处理器，从而生产出一种机器，使得这些指令在通过计算机或其它可编程数据处理装置的处理器执行时，产生了实现流程图和/或框图中的一个或多个方框中规定的功能/动作的装置。也可以把这些计算机可读程序指令存储在计算机可读存储介质中，这些指令使得计算机、可编程数据处理装置和/或其他设备以特定方式工作，从而，存储有指令的计算机可读介质则包括一个制造品，其包括实现流程图和/或框图中的一个或多个方框中规定的功能/动作的各个方面的指令。

[0116] 也可以把计算机可读程序指令加载到计算机、其它可编程数据处理装置、或其它设备上，使得在计算机、其它可编程数据处理装置或其它设备上执行一系列操作步骤，以产生计算机实现的过程，从而使得在计算机、其它可编程数据处理装置、或其它设备上执行的指令实现流程图和/或框图中的一个或多个方框中规定的功能/动作。

[0117] 附图中的流程图和框图显示了根据本发明的多个实施例的系统、方法和计算机程序产品的可能实现的体系架构、功能和操作。在这点上，流程图或框图中的每个方框可以代表一个模块、程序段或指令的一部分，所述模块、程序段或指令的一部分包含一个或多个用于实现规定的逻辑功能的可执行指令。在有些作为替换的实现中，方框中所标注的功能也可以以不同于附图中所标注的顺序发生。例如，两个连续的方框实际上可以基本并行地执行，它们有时也可以按相反的顺序执行，这依所涉及的功能而定。也要注意的是，框图和/或流程图中的每个方框、以及框图和/或流程图中的方框的组合，可以用执行规定的功能或动作的专用的基于硬件的系统来实现，或者可以用专用硬件与计算机指令的组合来实现。

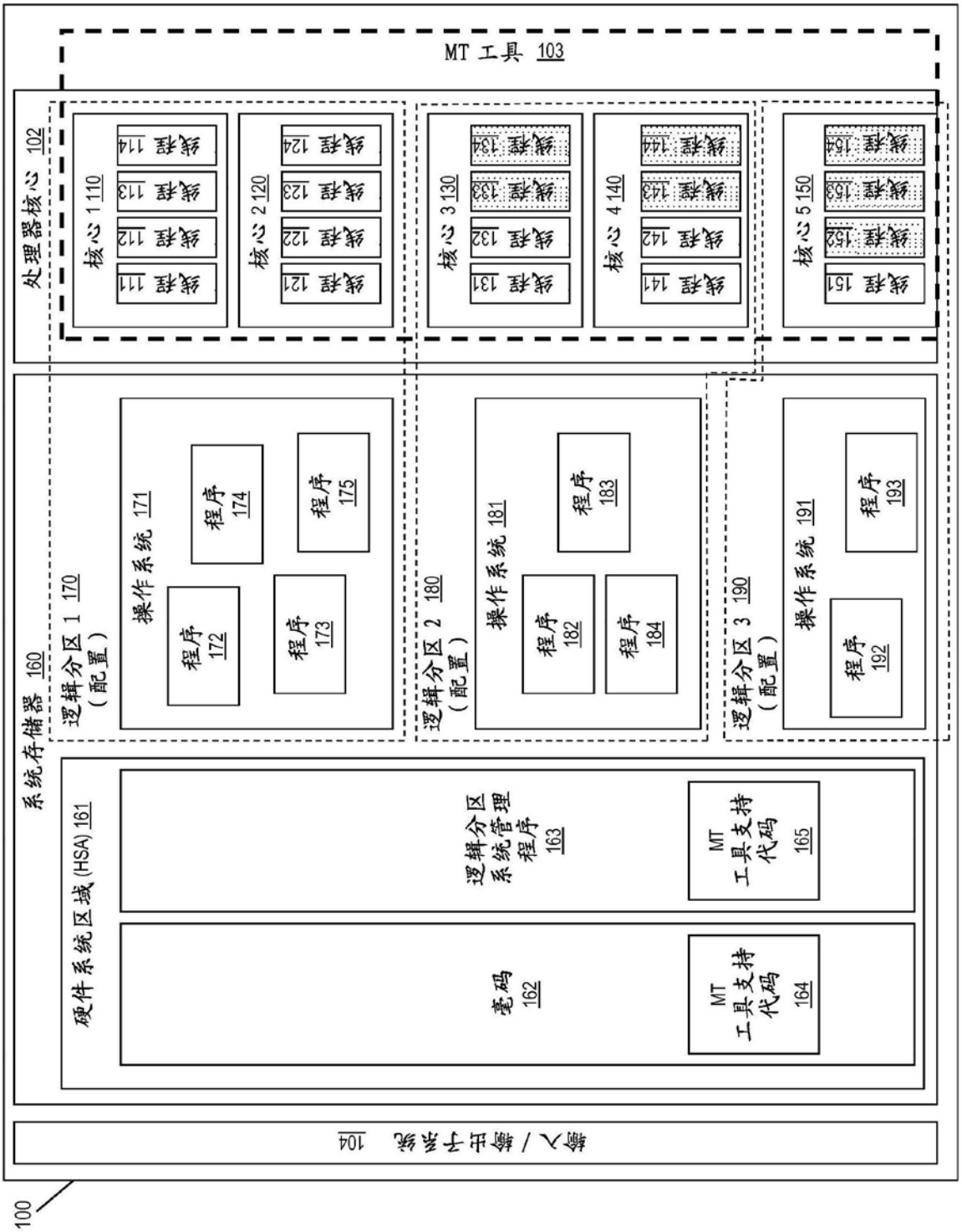

100
输入/输出子系统 104
系统存储器 160
硬件系统区域(HSA) 161
逻辑分区系统管理程序 163
MT 工具支持代码 165
毫码 162
MT 工具支持代码 164
逻辑分区 1 170 (配置)
操作系统 171
程序 172
程序 173
程序 174
程序 175
逻辑分区 2 180 (配置)
操作系统 181
程序 182
程序 184
程序 183
逻辑分区 3 190 (配置)
操作系统 191
程序 192
程序 193
处理器核心 102
核心 1 110
核心 2 120
核心 3 130
核心 4 140
核心 5 150
MT 工具 103

图1A

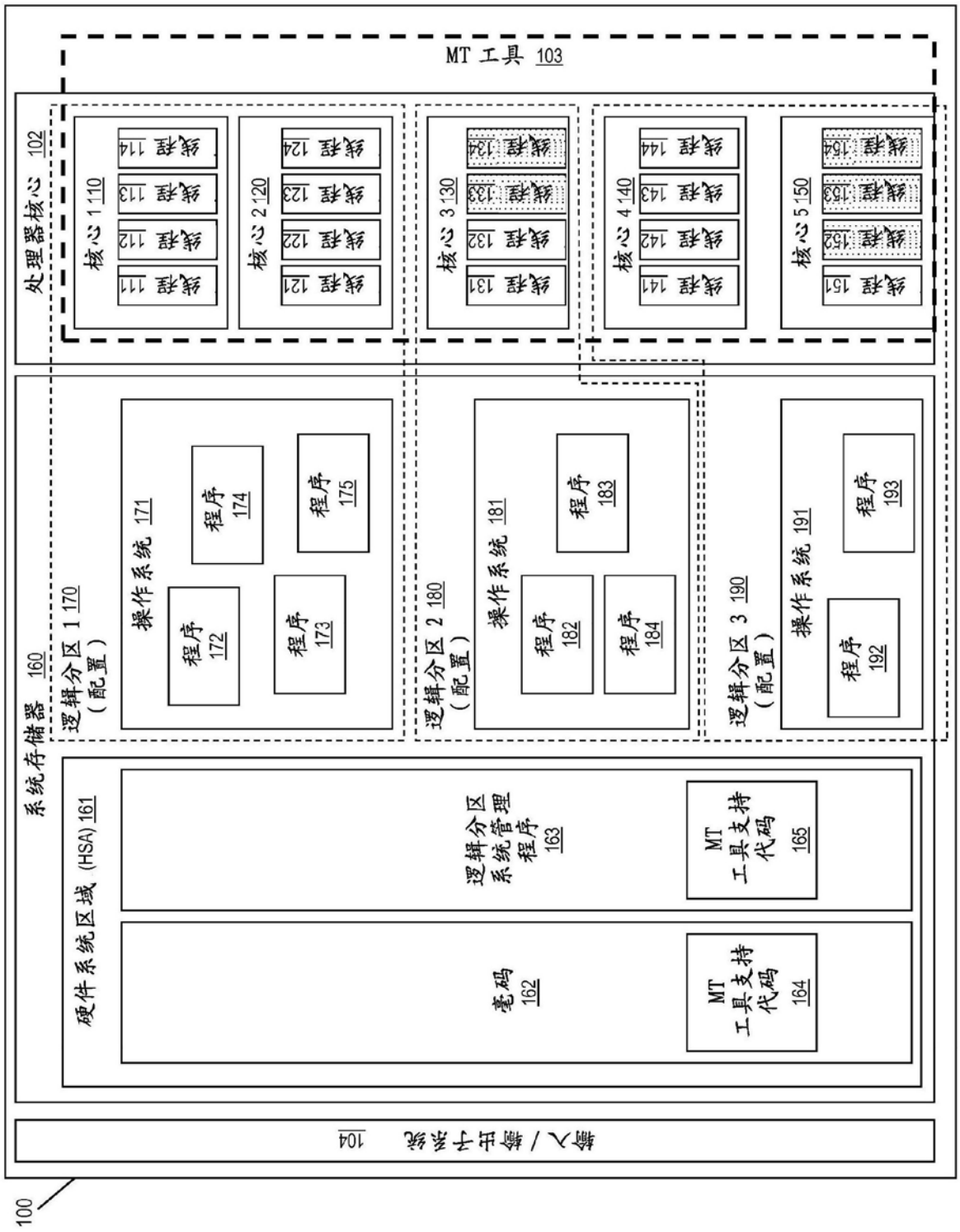

MT 工具 103
处理器核心 102
核心 1 110
核心 2 120
核心 3 130
核心 4 140
核心 5 150
线程 111
线程 112
线程 113
线程 114
线程 121
线程 122
线程 123
线程 124
线程 131
线程 132
线程 133
线程 134
线程 141
线程 142
线程 143
线程 144
线程 151
线程 152
线程 153
线程 154
系统存储器 160
逻辑分区 1 170
（配置）
操作系统 171
程序 172
程序 173
程序 174
程序 175
逻辑分区 2 180
（配置）
操作系统 181
程序 182
程序 183
程序 184
逻辑分区 3 190
（配置）
操作系统 191
程序 192
程序 193
硬件系统区域 (HSA) 161
逻辑分区系统管理程序 163
MT 工具支持代码 165
毫码 162
MT 工具支持代码 164
输入 / 输出 104
100

图1B

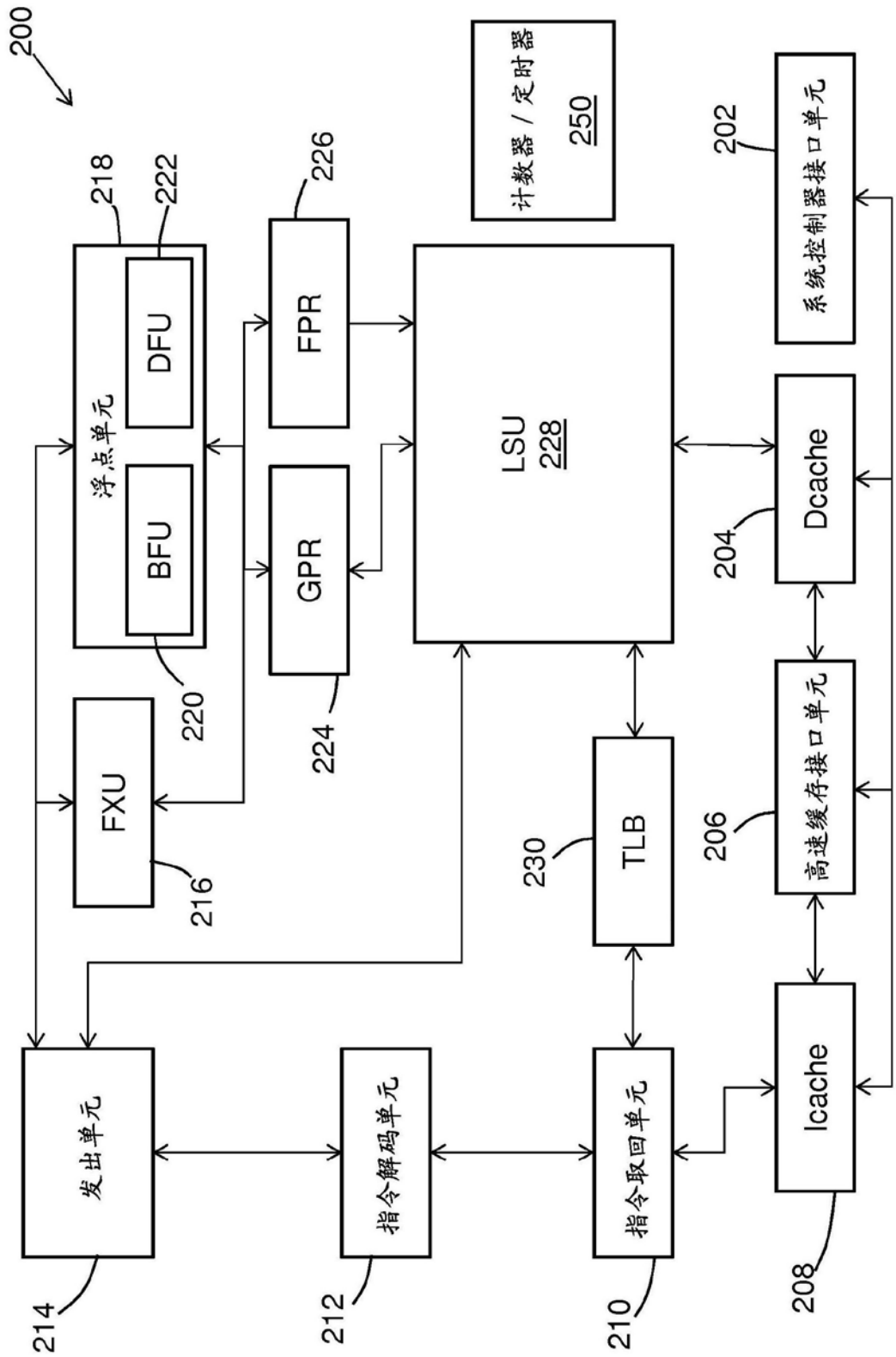

200
218
222
226
计数器 / 定时器
250
202
系统控制器接口单元
浮点单元
DFU
BFU
FPR
GPR
LSU
228
Dcache
204
220
224
FXU
216
230
TLB
206
高速缓存接口单元
发出单元
指令解码单元
指令取回单元
Icache
214
212
210
208

图2

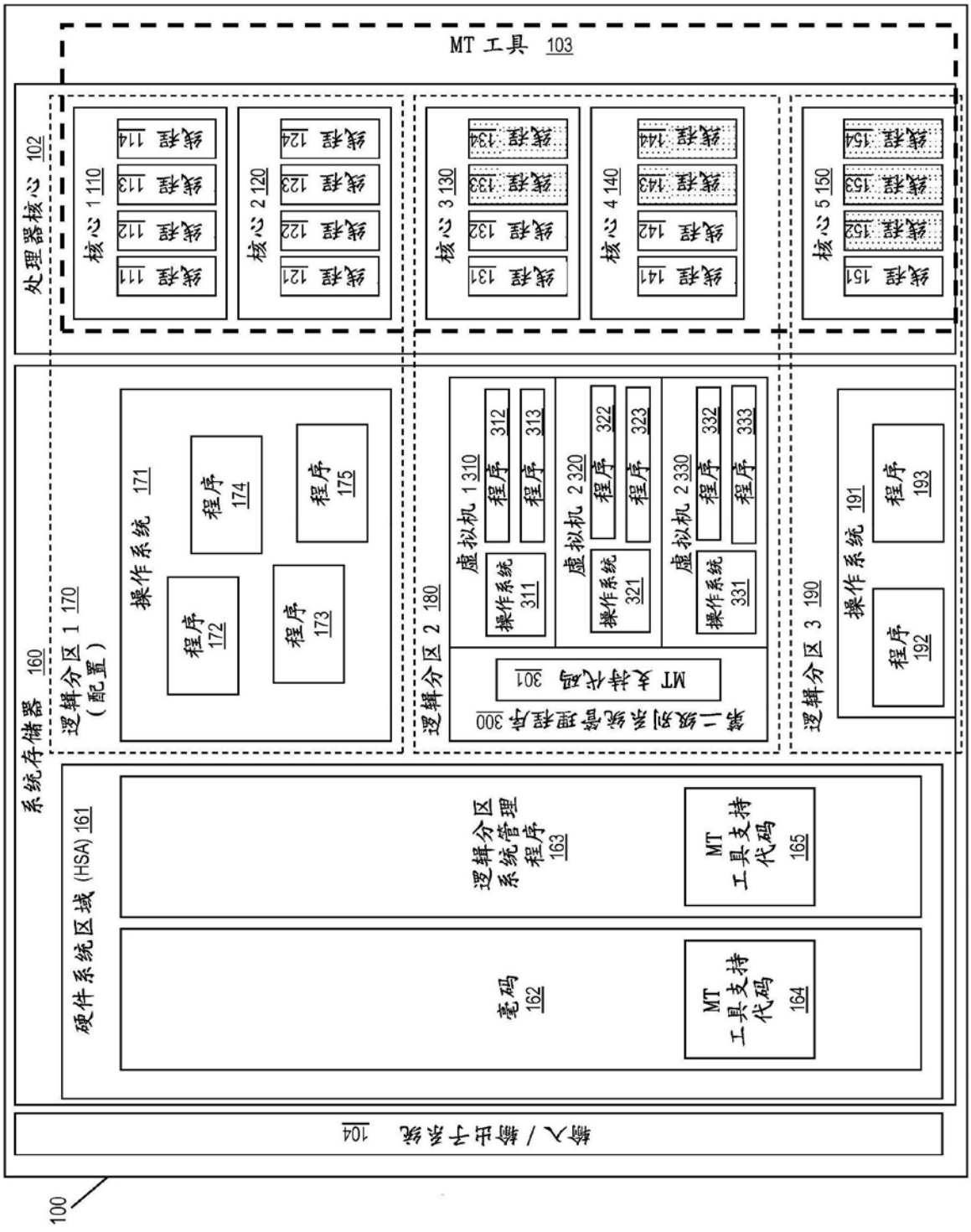

MT 工具 103
处理器核心 102
核心 1 110
核心 2 120
核心 3 130
核心 4 140
核心 5 150
线程 121
线程 122
线程 123
线程 124
线程 125
线程 126
线程 127
线程 128
线程 129
线程 131
线程 132
线程 133
线程 134
线程 135
系统存储器 160
逻辑分区 1 170
(配置)
操作系统 171
程序 172
程序 173
程序 174
程序 175
逻辑分区 2 180
虚拟机 1 310
操作系统 311
程序 312
程序 313
虚拟机 2 320
操作系统 321
程序 322
程序 323
虚拟机 2 330
操作系统 331
程序 332
程序 333
MT 特定代码 301
逻辑分区管理程序 300
逻辑分区 3 190
操作系统 191
程序 192
程序 193
硬件系统区域 (HSA) 161
逻辑分区系统管理程序 163
MT 工具支持代码 165
电码 162
MT 工具支持代码 164
输入/输出 I/O 设备 108
100

图3

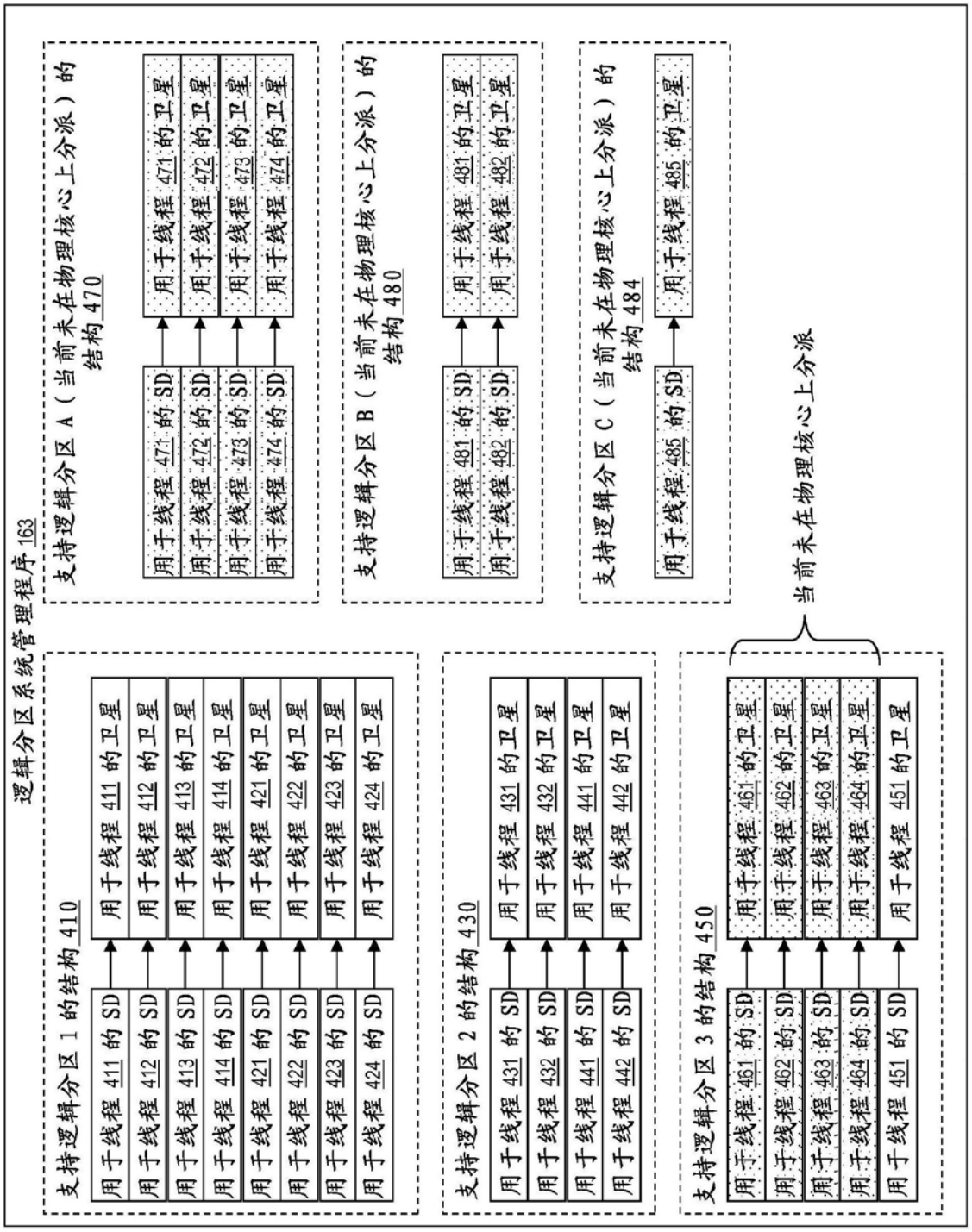

逻辑分区系统管理程序 163
支持逻辑分区 1 的结构 410
用于线程 411 的 SD
用于线程 412 的 SD
用于线程 413 的 SD
用于线程 414 的 SD
用于线程 421 的 SD
用于线程 422 的 SD
用于线程 423 的 SD
用于线程 424 的 SD
用于线程 411 的卫星
用于线程 412 的卫星
用于线程 413 的卫星
用于线程 414 的卫星
用于线程 421 的卫星
用于线程 422 的卫星
用于线程 423 的卫星
用于线程 424 的卫星
支持逻辑分区 2 的结构 430
用于线程 431 的 SD
用于线程 432 的 SD
用于线程 441 的 SD
用于线程 442 的 SD
用于线程 431 的卫星
用于线程 432 的卫星
用于线程 441 的卫星
用于线程 442 的卫星
支持逻辑分区 3 的结构 450
用于线程 461 的 SD
用于线程 462 的 SD
用于线程 463 的 SD
用于线程 464 的 SD
用于线程 451 的 SD
用于线程 461 的卫星
用于线程 462 的卫星
用于线程 463 的卫星
用于线程 464 的卫星
用于线程 451 的卫星
当前未在物理核心上分派
支持逻辑分区 A（当前未在物理核心上分派）的结构 470
用于线程 471 的 SD
用于线程 472 的 SD
用于线程 473 的 SD
用于线程 474 的 SD
用于线程 471 的卫星
用于线程 472 的卫星
用于线程 473 的卫星
用于线程 474 的卫星
支持逻辑分区 B（当前未在物理核心上分派）的结构 480
用于线程 481 的 SD
用于线程 482 的 SD
用于线程 481 的卫星
用于线程 482 的卫星
支持逻辑分区 C（当前未在物理核心上分派）的结构 484
用于线程 485 的 SD
用于线程 485 的卫星

图4

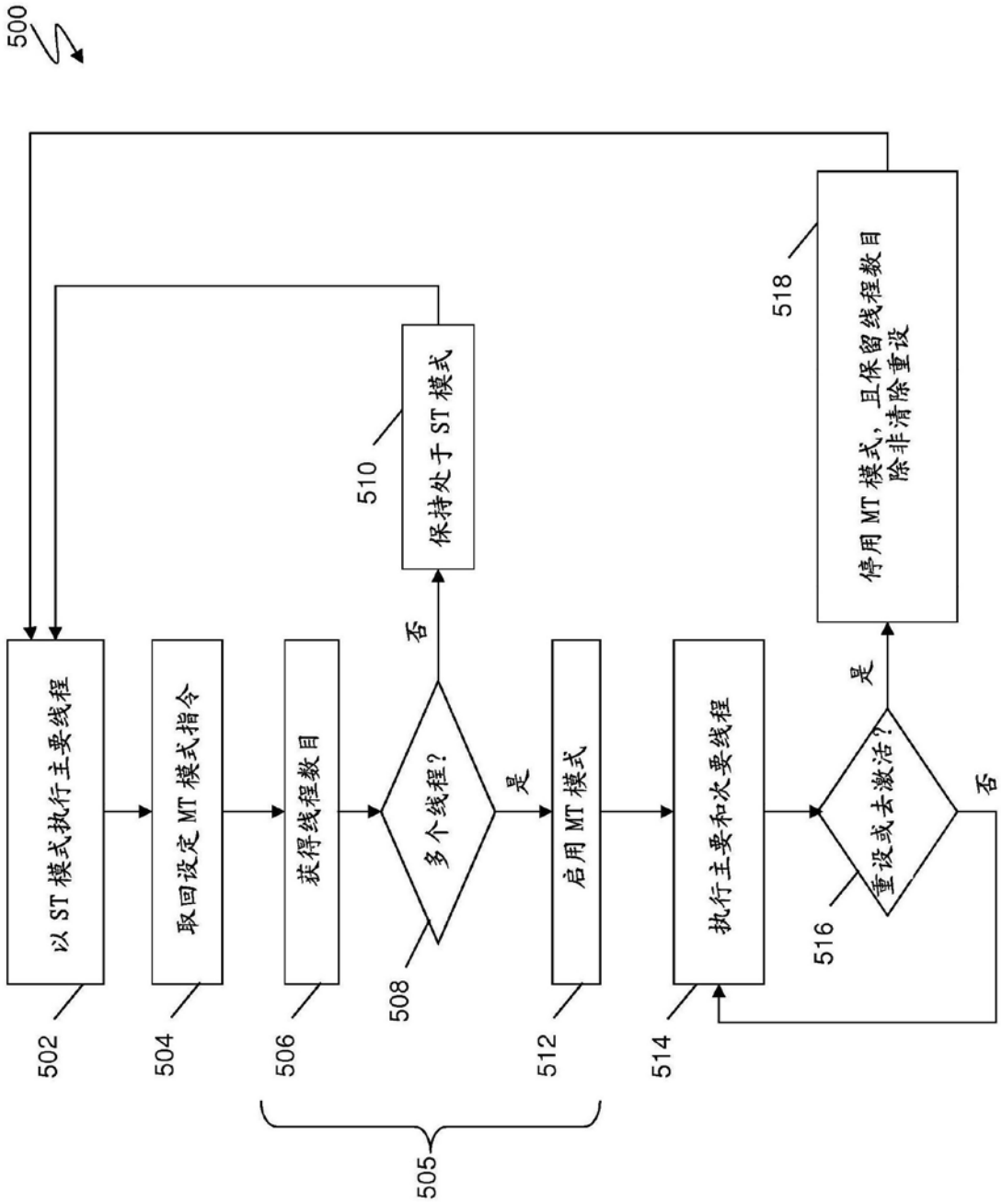

500
以 ST 模式执行主要线程
502
取回设定 MT 模式指令
504
获得线程数目
506
多个线程?
508
否
保持处于 ST 模式
510
是
启用 MT 模式
512
505
执行主要和次要线程
514
重设或去激活?
516
是
停用 MT 模式，且保留线程数目
除非清除重设
518
否

图5

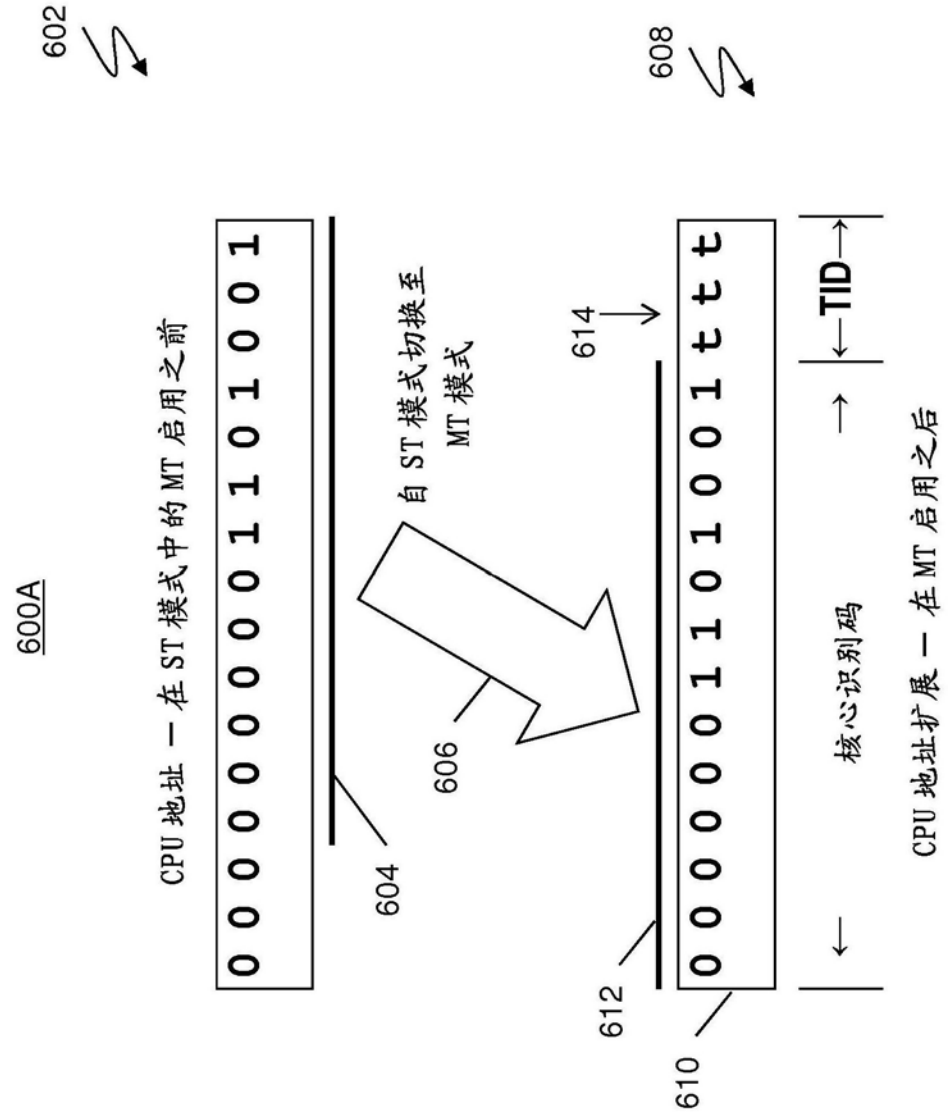

602
608
600A
CPU 地址 — 在 ST 模式中的 MT 启用之前
0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
自 ST 模式切换至
MT 模式
606
604
614
612
610
0 0 0 0 0 0 0 1 1 0 1 0 0 1 t t t
核心识别码
TID
CPU 地址扩展 — 在 MT 启用之后

图6A

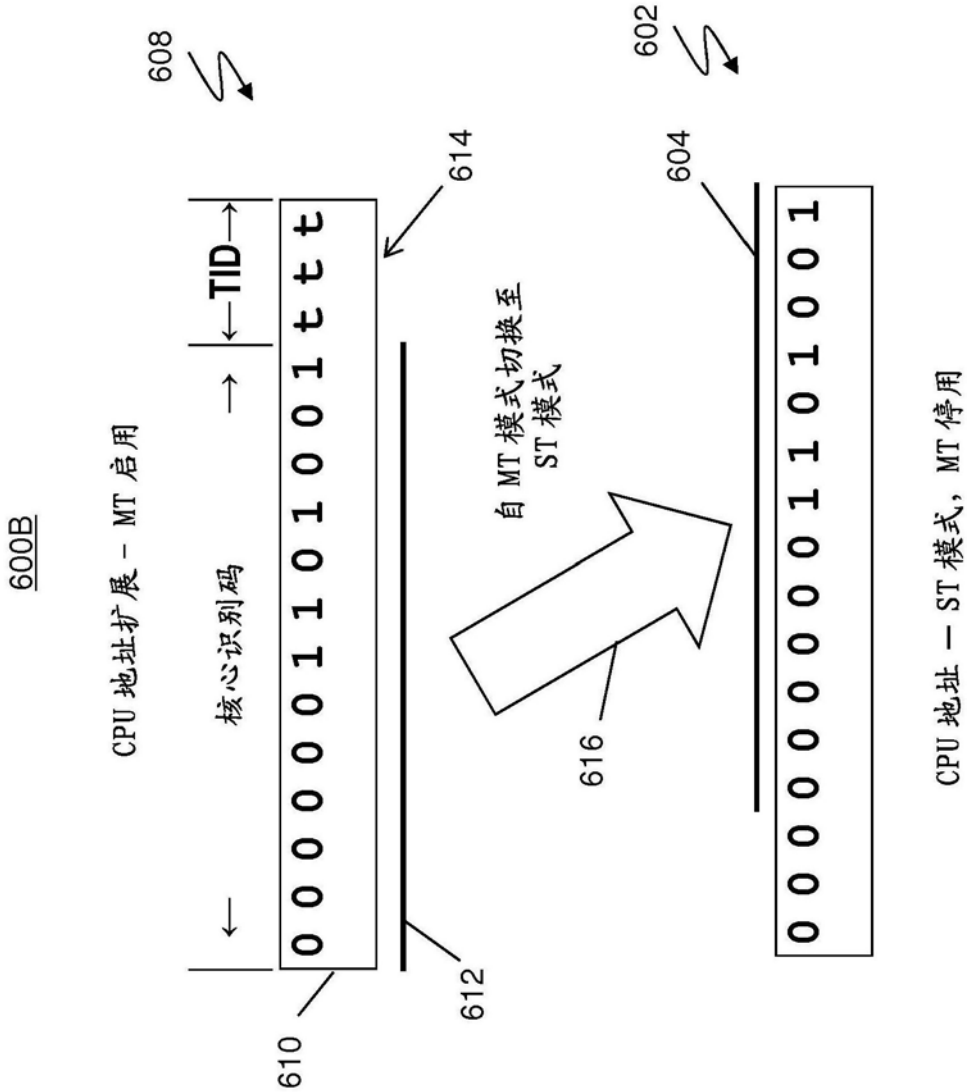

600B
CPU 地址扩展 - MT 启用
核心识别码
TID
0 0 0 0 0 0 0 1 1 0 1 0 0 1 t t t t
610
612
614
608
自 MT 模式切换至
ST 模式
616
602
604
0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
CPU 地址 — ST 模式，MT 停用

图6B

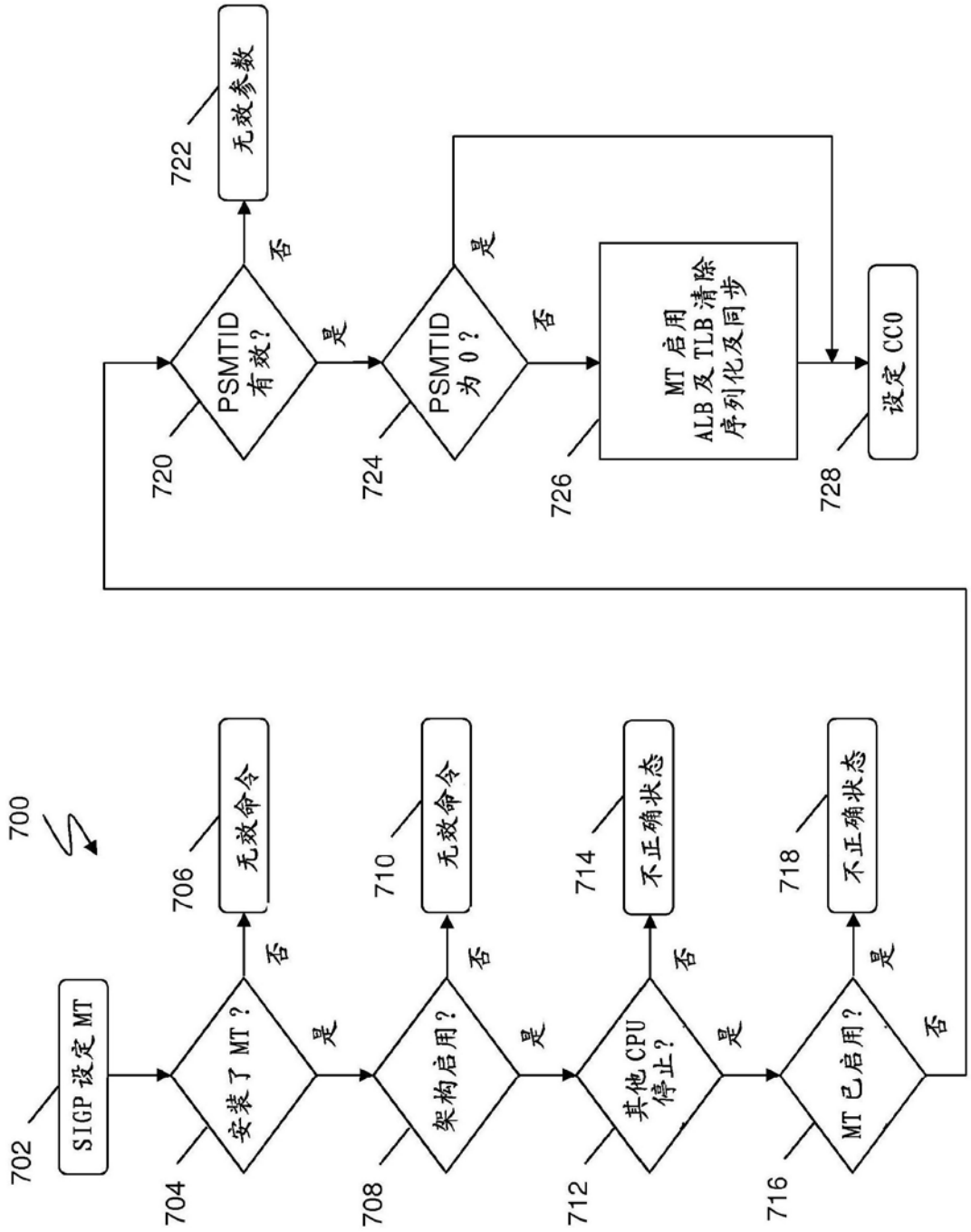

700
SIGP 设定 MT
702
安装了 MT？
704
是
否
无效命令
706
架构启用？
708
是
否
无效命令
710
其他 CPU 停止？
712
是
否
不正确状态
714
MT 已启用？
716
是
否
不正确状态
718
PSMTID 有效？
720
是
否
无效参数
722
PSMTID 为 0？
724
是
否
MT 启用 ALB 及 TLB 清除序列化及同步
726
设定 CC0
728

图7

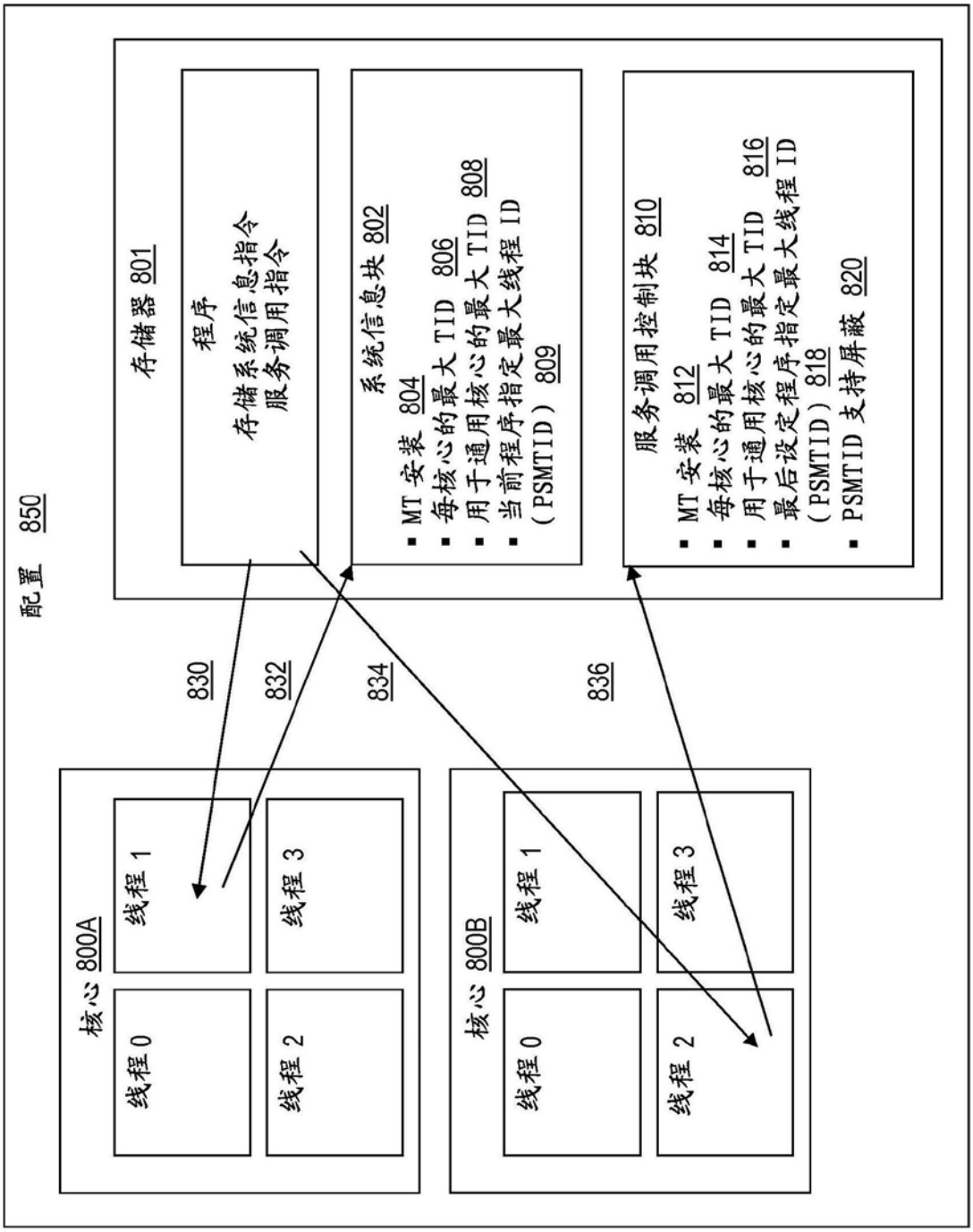

配置 850
存储器 801
程序
存储系统信息指令
服务调用指令
系统信息块 802
MT 安装 804
每核心的最大 TID 806
用于通用核心的最大 TID 808
当前程序指定最大线程 ID
(PSMTID) 809
服务调用控制块 810
MT 安装 812
每核心的最大 TID 814
用于通用核心的最大 TID 816
最后设定程序指定最大线程 ID
(PSMTID) 818
PSMTID 支持屏蔽 820
830
832
834
836
核心 800A
线程 0
线程 1
线程 2
线程 3
核心 800B
线程 0
线程 1
线程 2
线程 3

图8

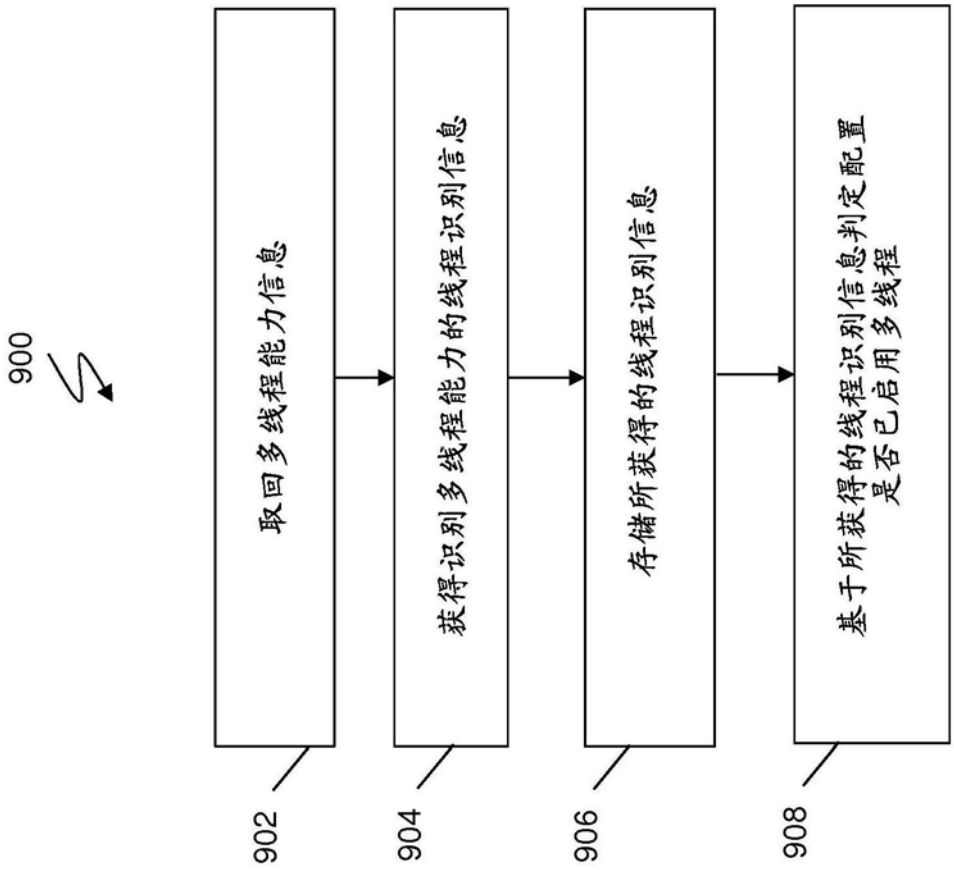

900
取回多线程能力信息
获得识别多线程能力的线程识别信息
存储所获得的线程识别信息
基于所获得的线程识别信息判定配置是否已启用多线程
902
904
906
908

图9

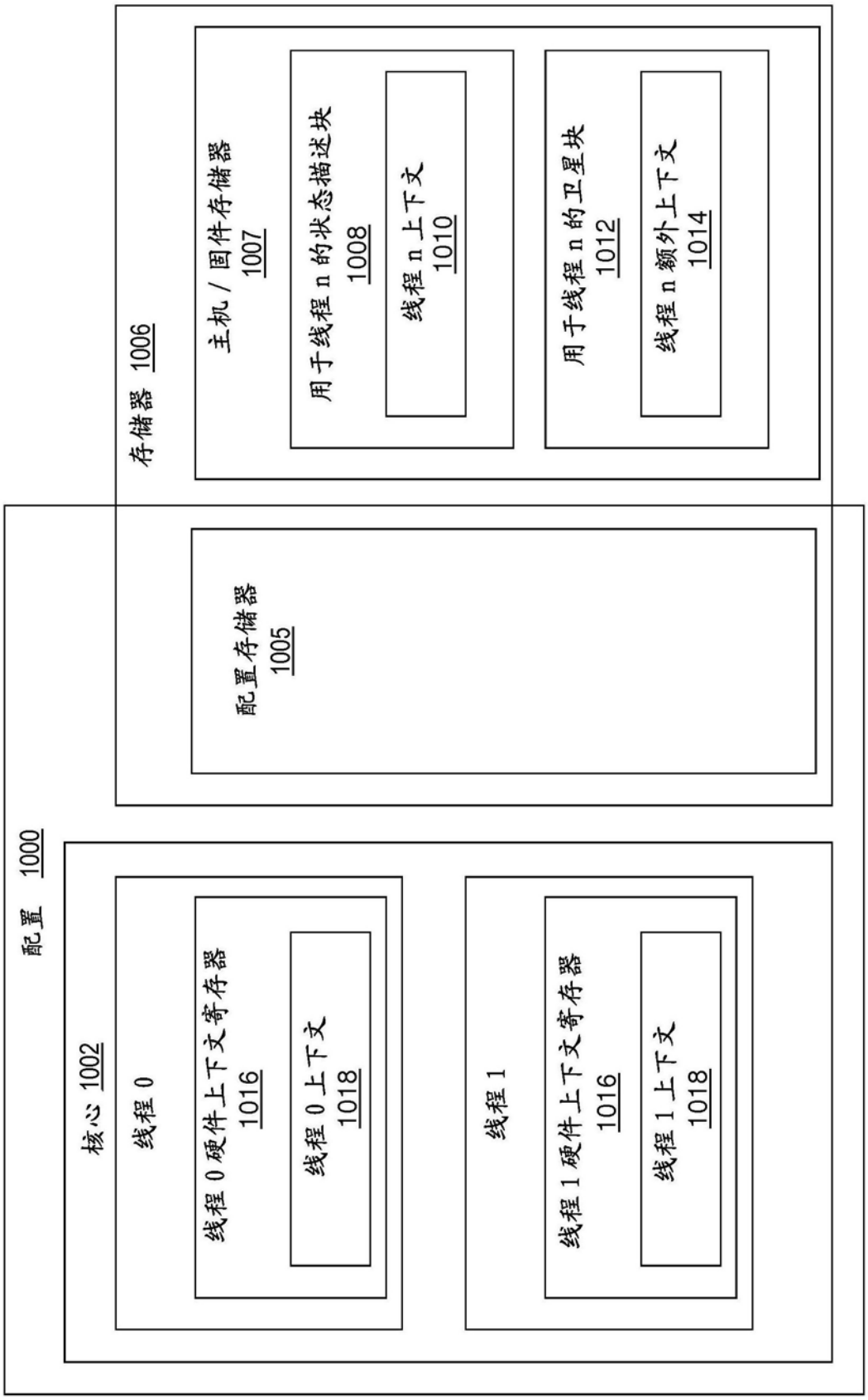

配置 1000
核心 1002
线程 0
线程 0 硬件上下文寄存器 1016
线程 0 上下文 1018
线程 1
线程 1 硬件上下文寄存器 1016
线程 1 上下文 1018
存储器 1006
配置存储器 1005
主机 / 固件存储器 1007
用于线程 n 的状态描述块 1008
线程 n 上下文 1010
用于线程 n 的卫星块 1012
线程 n 额外上下文 1014

图10

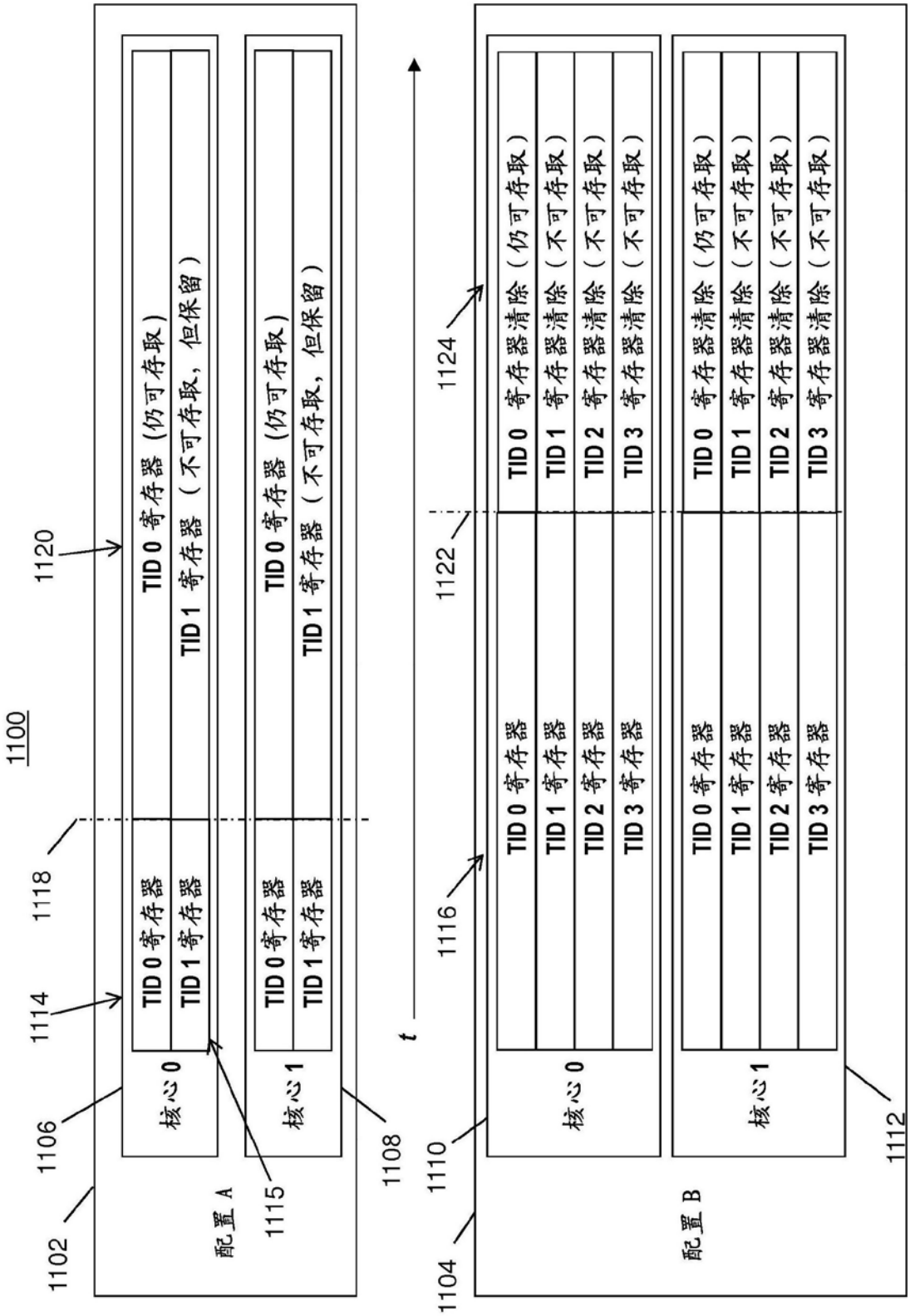

1100
1102
1104
1106
1108
1110
1112
1114
1115
1116
1118
1120
1122
1124
核心 0
核心 1
TID 0 寄存器
TID 1 寄存器
TID 0 寄存器 (仍可存取)
TID 1 寄存器 (不可存取，但保留)
TID 0 寄存器 (仍可存取)
TID 1 寄存器 (不可存取，但保留)
配置 A
配置 B
t
TID 0 寄存器清除 (仍可存取)
TID 1 寄存器清除 (不可存取)
TID 2 寄存器清除 (不可存取)
TID 3 寄存器清除 (不可存取)
TID 0 寄存器
TID 1 寄存器
TID 2 寄存器
TID 3 寄存器

图11

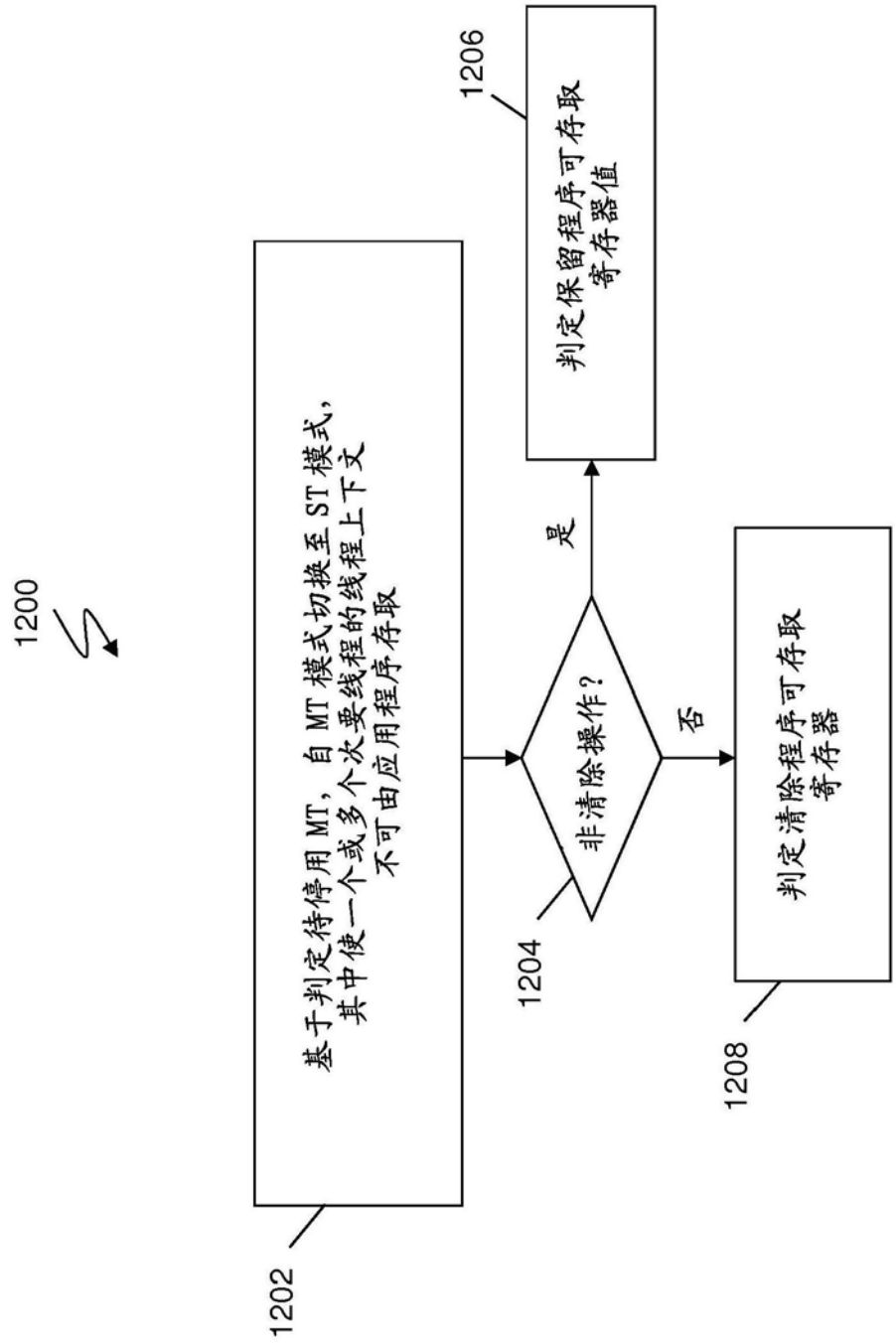

1200
基于判定待停用 MT，自 MT 模式切换至 ST 模式，
其中使一个或多个次要线程的线程上下文
不可由应用程序存取
1202
非清除操作？
1204
是
否
判定保留程序可存取
寄存器值
1206
判定清除程序可存取
寄存器
1208

图12

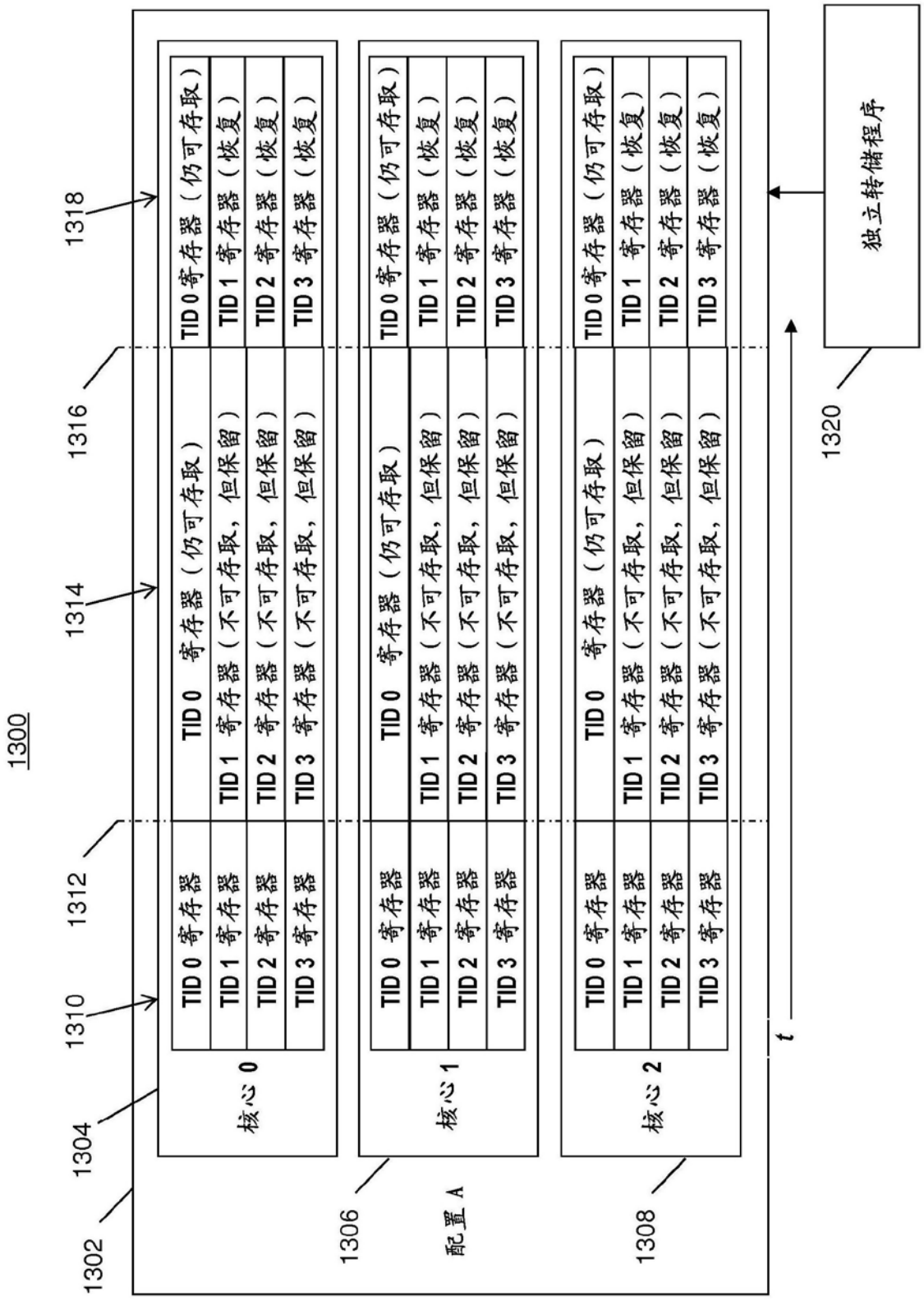

1300
1302
1304
1306
1308
1310
1312
1314
1316
1318
1320
配置 A
核心 0
核心 1
核心 2
TID 0 寄存器
TID 1 寄存器
TID 2 寄存器
TID 3 寄存器
TID 0 寄存器（仍可存取）
TID 1 寄存器（不可存取，但保留）
TID 2 寄存器（不可存取，但保留）
TID 3 寄存器（不可存取，但保留）
TID 0寄存器（仍可存取）
TID 1 寄存器（恢复）
TID 2 寄存器（恢复）
TID 3 寄存器（恢复）
t
独立转储程序

图13

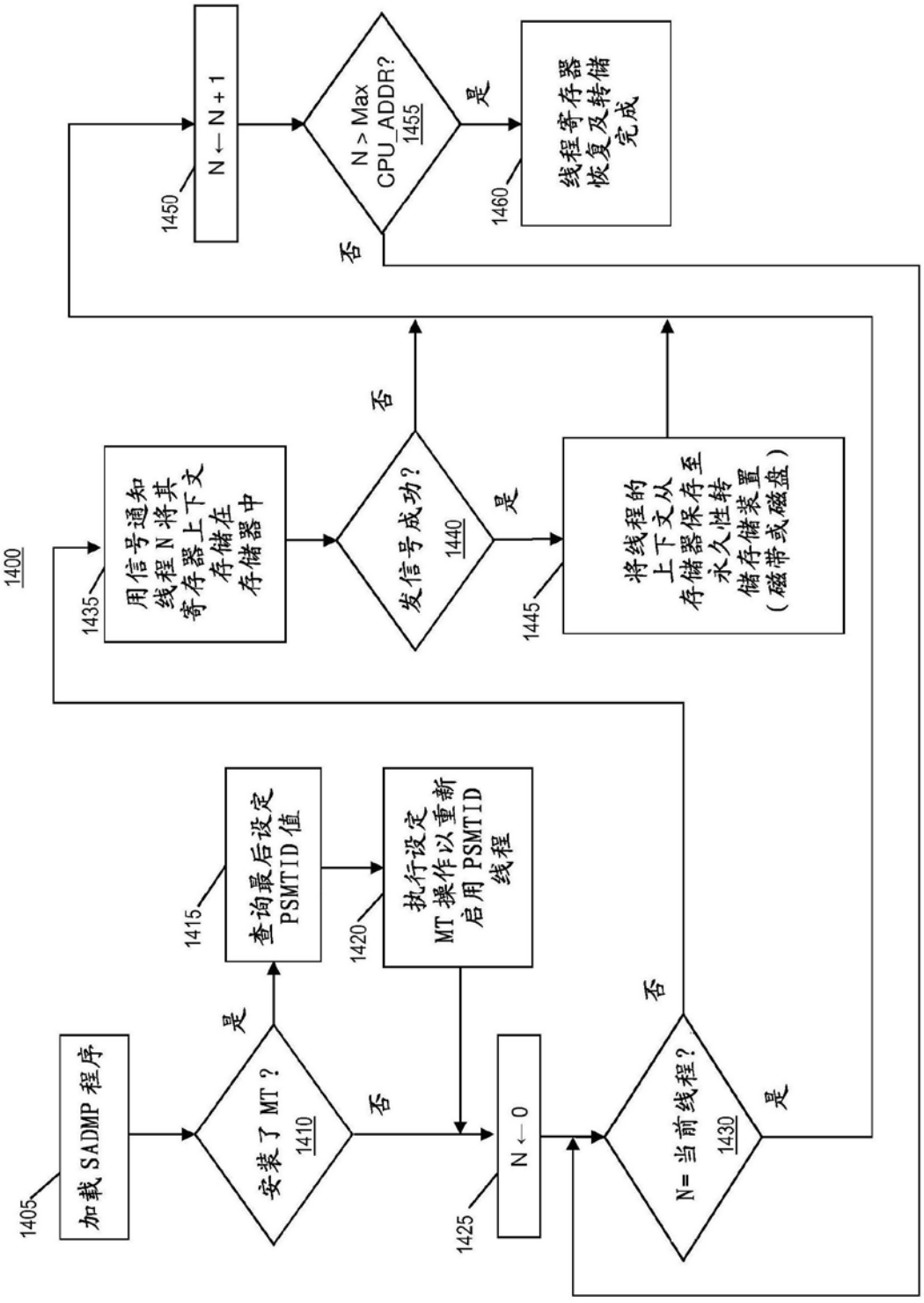

1400
1405
加载 SADMP 程序
安装了 MT ?
1410
是
否
1415
查询最后设定 PSMTID 值
1420
执行设定 MT 操作以重新启用 PSMTID 线程
1425
N ← 0
N= 当前线程?
1430
是
否
1435
用信号通知线程 N 将其寄存器上下文存储在存储器中
发信号成功?
1440
是
否
1445
将线程的上下文从存储器保存至永久性转储存储装置（磁带或磁盘）
N ← N + 1
1450
N > Max CPU_ADDR?
1455
是
否
1460
线程寄存器恢复及转储完成

图14

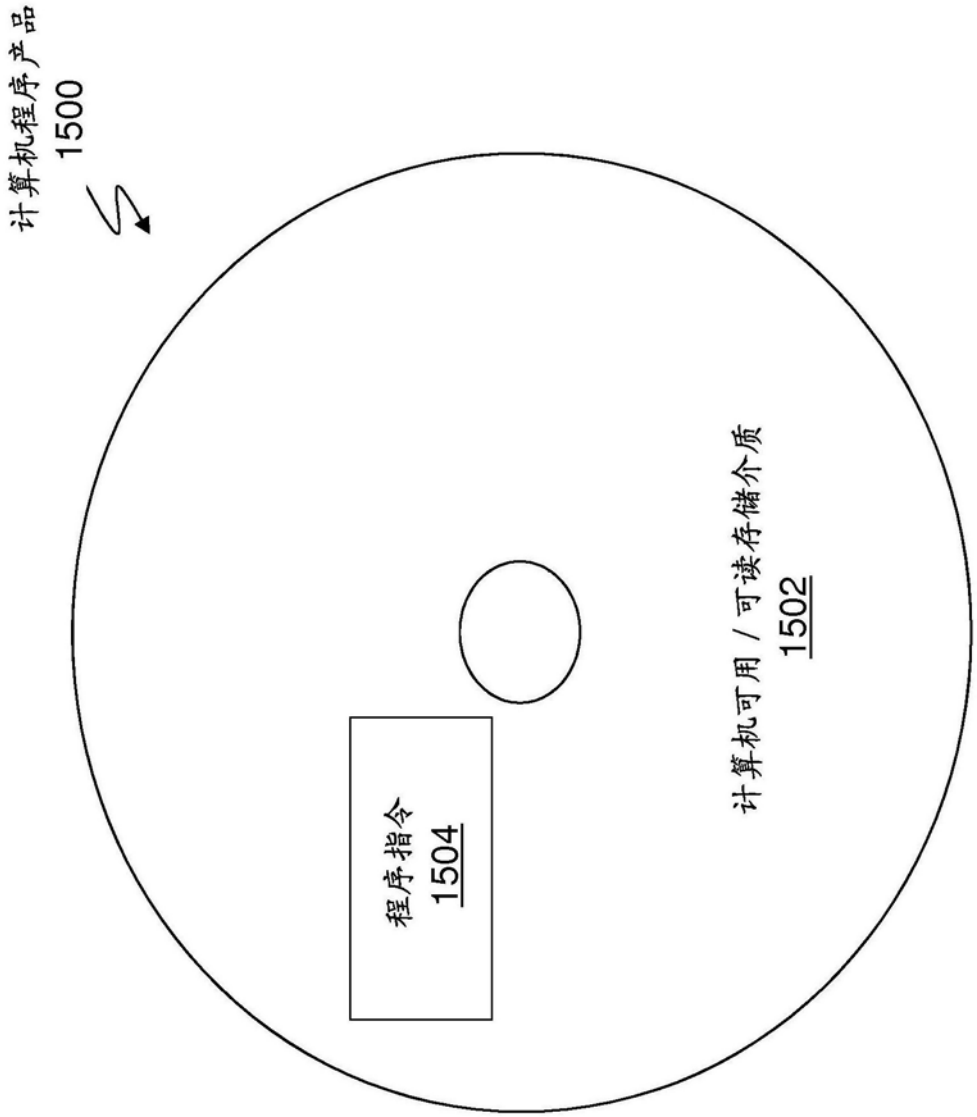


图15