



(12)发明专利

(10)授权公告号 CN 104011624 B

(45)授权公告日 2018.03.30

(21)申请号 201280063695.7

(72)发明人 I·M·索迪 A·纳韦 D·拉杰万

(22)申请日 2012.11.26

R·D·威尔斯 E·C·萨姆森

(65)同一申请的已公布的文献号

(74)专利代理机构 上海专利商标事务所有限公司 31100

申请公布号 CN 104011624 A

代理人 姬利永

(43)申请公布日 2014.08.27

(51)Int.CI.

G06F 1/32(2006.01)

(30)优先权数据

13/335,738 2011.12.22 US

(56)对比文件

(85)PCT国际申请进入国家阶段日

US 2007/0214289 A1, 2007.09.13,

2014.06.20

CN 101573677 A, 2009.11.04,

(86)PCT国际申请的申请数据

US 2007/0214289 A1, 2007.09.13,

PCT/US2012/066557 2012.11.26

US 2011/0231681 A1, 2011.09.22,

(87)PCT国际申请的公布数据

审查员 徐生芹

W02013/095869 EN 2013.06.27

(73)专利权人 英特尔公司

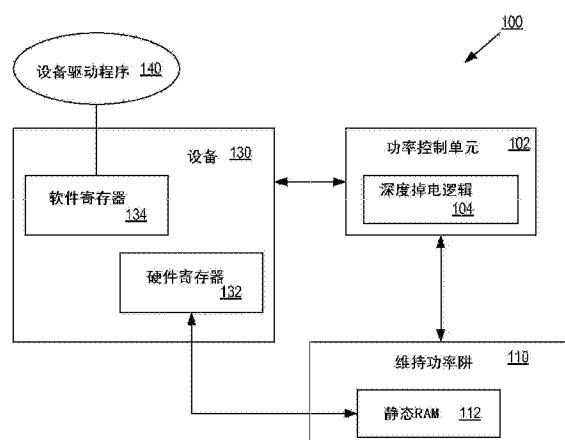
权利要求书2页 说明书12页 附图12页

地址 美国加利福尼亚州

(54)发明名称

包括设备中自主的基于硬件的深度掉电的
用于能效和节能的方法、装置和系统

(57)摘要

描述了包括允许设备的自主的基于硬件的
深度掉电的用于能效和节能的系统、装置和方法
的实施例。在一个实施例中，系统包括设备、静态
存储器、以及与该设备和该静态存储器耦合的功
率控制单元。该系统还包括该功率控制单元的深
度掉电逻辑，用于监视该设备的状态并且用于在
该设备空闲时将该设备转换到深度掉电状态。在
该系统中，该设备在深度掉电状态时比在空闲状
态时消耗更少的功率。

1. 一种用于高效能耗的系统,包括:

设备;

静态存储器;以及

与所述设备和所述静态存储器耦合的功率控制单元,所述功率控制单元包括深度掉电逻辑单元,所述深度掉电逻辑单元用于监视所述设备的状态以及在所述设备空闲时将所述设备转换到深度掉电状态,其中所述设备在所述深度掉电状态时比在空闲状态时消耗更少的功率,

其中,所述深度掉电逻辑单元用于监视以下至少一个度量:所述设备的状态、所述系统中各组件的状态、所述设备接收到的命令的模式或时序、所述设备所包含的软件/硬件寄存器的读/写操作的模式或时序、与另一设备或CPU核的事务的模式或时序,来确定所述设备跟随所述CPU核的深度掉电状态或者独立于所述CPU核而转换到/转换出所述深度掉电状态。

2. 如权利要求1所述的系统,其特征在于,还包括:

所述功率控制单元用于在所述设备被转换到所述深度掉电状态时将从所述设备获得的上下文数据存储在所述静态存储器中。

3. 如权利要求2所述的系统,其特征在于,还包括:

所述深度掉电逻辑单元用于响应于所述深度掉电逻辑单元监视到一事件而唤醒所述设备,其中所述深度掉电逻辑单元用于将所述设备转换到活动状态并且将所述上下文数据从所述静态存储器加载到所述设备中。

4. 如权利要求3所述的系统,其特征在于,所述事件是所述系统的处理器从空闲状态唤醒。

5. 如权利要求3所述的系统,其特征在于,所述事件是要由所述设备处理的事件,并且其中所述系统的处理器留在空闲状态中。

6. 如权利要求1所述的系统,其中所述深度掉电逻辑单元用于确定所述设备空闲并且在所述系统的处理器进入空闲状态时发起所述转换。

7. 如权利要求1所述的系统,其中所述深度掉电逻辑单元用于在所述系统的处理器处于活动状态时发起将所述设备转换到所述深度掉电状态的转换。

8. 如权利要求1所述的系统,其特征在于,所述静态存储器与维持功率阱耦合以便向所述静态存储器供电。

9. 如权利要求1所述的系统,其特征在于,所述设备是图形处理单元。

10. 如权利要求1所述的系统,其特征在于,所述深度掉电状态是C6状态。

11. 一种用于高效能耗的方法,包括:

监视设备的状态;

确定与所述设备相对应的空闲状态;以及

利用基于硬件的深度掉电逻辑,在确定所述设备空闲时将所述设备转换到深度掉电状态,其中所述设备在所述深度掉电状态时比在所述空闲状态时消耗更少的功率,

所述方法还包括:

监视以下至少一个度量:所述设备的状态、包括所述设备的系统中各组件的状态、所述设备接收到的命令的模式或时序、所述设备所包含的软件/硬件寄存器的读/写操作的模式

或时序、与另一设备或CPU核的事务的模式或时序，来确定所述设备跟随所述CPU核的深度掉电状态或者独立于所述CPU核而转换到/转换出所述深度掉电状态。

12. 如权利要求11所述的方法，其特征在于，还包括：

在所述设备被转换到所述深度掉电状态时将从所述设备获得的上下文数据存储在静态存储器中。

13. 如权利要求12所述的方法，其特征在于，还包括：

响应于监视到一事件而唤醒所述设备；

将所述设备转换到活动状态；以及

将所述上下文数据从所述静态存储器加载到所述设备中。

14. 如权利要求13所述的方法，其特征在于，所述事件是所述系统的处理器从空闲状态唤醒。

15. 如权利要求13所述的方法，其特征在于，所述事件是要由所述设备处理的事件，并且其中所述系统的处理器留在空闲状态中。

16. 如权利要求11所述的方法，其特征在于，在所述系统的处理器进入空闲状态时发起将所述设备转换到所述深度掉电状态的转换。

17. 如权利要求11所述的方法，当所述系统的处理器处于活动状态时发起将所述设备转换到所述深度掉电状态的转换。

18. 如权利要求11所述的方法，其特征在于，所述设备是图形处理单元。

19. 一种用于高效能耗的装置，包括：

静态存储器；以及

与设备和所述静态存储器耦合的功率控制单元，所述功率控制单元包括深度掉电逻辑单元，所述深度掉电逻辑单元用于监视所述设备的状态并且在所述设备空闲时将所述设备转换到深度掉电状态，其中所述设备在所述深度掉电状态时比在空闲状态时消耗更少的功率，

其中，所述深度掉电逻辑单元用于监视以下至少一个度量：所述设备的状态、所述装置中各组件的状态、所述设备接收到的命令的模式或时序、所述设备所包含的软件/硬件寄存器的读/写操作的模式或时序、与另一设备或CPU核的事务的模式或时序，来确定所述设备跟随所述CPU核的深度掉电状态或者独立于所述CPU核而转换到/转换出所述深度掉电状态。

20. 如权利要求19所述的装置，其特征在于，所述功率控制单元用于在所述设备被转换到所述深度掉电状态时将从所述设备获得的上下文数据存储在所述静态存储器中。

21. 一种非瞬态机器可读介质，其上存储有指令，所述指令在被执行时致使机器执行如权利要求11-18中任一项所述的方法。

22. 一种用于执行指令的设备，包括多个装置，每个装置用于执行如权利要求11-18中任一项所述的方法的相应步骤。

包括设备中自主的基于硬件的深度掉电的用于能效和节能的方法、装置和系统

技术领域

[0001] 本公开涉及集成电路中的能效和节能，并且具体但非排他性地涉及降低计算机处理系统中设备的功耗。

[0002] 背景

[0003] 半导体处理和逻辑设计领域的进步已经允许增加集成电路器件上可存在的逻辑数量。结果，计算机系统配置已从系统中的单个或多个集成电路演变至多硬件线程、多核、多设备和/或完善的各集成电路上的系统。附加地，随着集成电路的密度增长，计算系统的功率需求（从嵌入式系统至服务器）也已逐步升级。此外，软件效率低及其对硬件的需要也已造成计算设备能耗的增加。事实上，一些研究表明，计算设备消耗整个国家（例如美国）的电力供给的相当大百分比。结果，对于与集成电路关联的能效和节能具有至关重要的需求。随着服务器、台式计算机、笔记本电脑、超级本、平板计算机、移动电话、处理器、嵌入式系统等设备变得越来越盛行（从包含在典型计算机、汽车和电视机中乃至到生物技术），这些需求将增加。

附图说明

[0004] 本发明是通过示例说明的，而不仅局限于各个附图的图示，在附图中，类似的参考标号表示类似的元件，其中：

[0005] 图1是示出具有用于设备的深度掉电的系统的框图。

[0006] 图2A示出用于执行处理系统的设备的深度掉电的方法的实施例。

[0007] 图2B示出用于将设备从处理系统的深度掉电状态唤醒的方法的实施例。

[0008] 图3A是示出根据本发明的实施例的示例性有序流水线以及示例性寄存器重命名的无序发布/执行流水线两者的方框图。

[0009] 图3B是示出根据本发明的实施例的有序架构核的示例性实施例以及包括在处理器中的示例性寄存器重命名的无序发布/执行架构核两者的方框图。

[0010] 图4A-B描述更具体的示例性的有序核架构的框图。

[0011] 图5是处理器的框图。

[0012] 图6-9是示例性计算机架构的方框图。

[0013] 图10是根据本发明的实施例的对比使用软件指令变换器将源指令集中的二进制指令转换成目标指令集中的二进制指令的框图。

具体实施方式

[0014] 在下面的描述中，阐述了很多具体细节。然而，应当理解，本发明的各实施例可以在不具有这些具体细节的情况下得到实施。在其他实例中，未详细示出公知的电路、结构和技术以免混淆对本描述的理解。

[0015] 在说明书中对“一个实施例”、“实施例”、“示例实施例”等的引用指示所描述的实

施例可以包括特定特征、结构或特性，但并不一定每个实施例都需要包括该特定特征、结构或特性。此外，这样的短语不一定是指同一个实施例。此外，当结合一个实施例描述特定特征、结构或特性时，我们认为，可在本领域技术人员的学识范围内，与其他实施例相结合地影响这样的特征、结构或特性，无论是否对此明确描述。

[0016] 尽管参考特定集成电路中(诸如计算平台或微处理器中)的节能和能效描述了以下实施例，但是其它实施例适用于其它类型的集成电路和逻辑器件。本文描述的实施例的相似技术和教义可适用于可从更好的能效和节能中得益的其它类型的电路或半导体器件。例如，所披露的实施例不限于台式计算机系统，并也可用于其它设备，例如手持设备、片上系统(SoC)以及嵌入式应用。手持式设备的一些示例包括：蜂窝电话、互联网协议设备、数码相机、个人数字助理(PDA)、手持式PC。嵌入式应用典型包括：微控制器、数字信号处理器(DSP)、片上系统、网络计算机(NetPC)、机顶盒、网络中枢、广域网(WAN)交换机、或可执行如下所教导的功能和操作的任何其他系统。此外，本文描述的装置、方法和系统不限于物理计算设备，而是也涉及对节能和能效的软件优化。如将在以下描述中变得显而易见的，本文描述的方法、装置和系统的实施例(无论是关于硬件、固件、软件还是它们的组合)对于“绿色技术”(诸如，用在涵盖美国经济很大部分的产品中的节能和能效)的前景是至关重要的。

[0017] 以下是用于在处理系统中启用设备中的自主深度掉电(*autonomous deep power down*)的实施例，其有益于降低所述设备的能耗，以及降低该处理系统的总能耗。计算系统在活动时消耗功率，诸如在它们执行处理操作、显示数据等时。为了在计算系统不活动时节省功率，处理核可进入深度掉电状态。深度掉电状态允许处理核的电压被降低，这又降低了处理核的功耗。在本文讨论的实施例中，设备(诸如图形处理设备、显示器、外围部件互联(PCI)控制器、图像信号处理器、音频加速器、视频加速器等)被允许进入深度掉电状态而无需软件干预。在一个实施例中，监视该设备以确定该设备何时空闲。在一个实施例中，当该设备空闲下来时，专用静态随机存取存储器(静态RAM)存储该设备的上下文数据(例如，引导上下文、内部寄存器内的数据等)，该RAM的维持轨提供正常电量以允许该RAM保持上下文数据的存储，而该设备被设置到深度掉电状态。在一个实施例中，该深度掉电状态为C6功率状态。在一个实施例中，上下文数据的存储和深度掉电状态的进入允许该设备的内部电压被降低到任何值，包括0(这将完全关闭该设备)。从而，通过使设备进入深度掉电状态，该设备的功耗可被最小化，甚至被消除。

[0018] 在一个实施例中，进一步监视该设备以确定该设备何时将从该深度掉电状态唤醒。在一个实施例中，当该设备从深度掉电状态唤醒并且被再通电时，该上下文数据被从该专用静态RAM加载到该设备中。在一个实施例中，该上下文数据被加载到该设备中以使得该设备具有与该设备进入深度掉电状态时相同的配置，并且该设备进入和退出该深度掉电状态是对该处理系统和由该处理系统执行的软件应用透明的。而且，如下面将更详细地描述的，当确定一设备空闲并使其进入深度掉电状态时，以及当确定该设备要被唤醒以退出深度掉电状态时，可以通过不同的试探来确定。

[0019] 图1是示出具有用于设备的深度掉电的系统100的框图。在一个实施例中，系统100是更大的计算机处理系统(如下文中在图6-9中所例示的那些系统)的一部分。在一个实施例中，系统100包括维持功率阱110、功率控制单元102、以及设备130。

[0020] 如上面讨论的，设备130可以是图形处理设备、显示器、PCI控制器等。在一个实施

例中，设备130的操作受设备驱动程序140的控制。在一个实施例中，设备驱动程序140是使设备130能够与处理系统交互的软件、硬件或其组合。在一个实施例中，功率控制单元102包括深度掉电逻辑104，诸如基于硬件的有限状态机，以便监视设备130的状态。在一个实施例中，深度掉电逻辑104是常醒逻辑(always-aware logic)，其通过监视从设备驱动程序140接收的命令、对寄存器132、134执行的操作、功率控制操作、设备130和其他设备(未示出)或系统100的CPU核(未示出)之间的事务等来监视设备130的状态。在一个实施例中，深度掉电逻辑104基于所监视到的设备130的状态来确定设备130是否空闲(即，没有在接收命令、在给定时间内没有接收到寄存器读或写命令等)以及设备130是否应当进入深度掉电状态。

[0021] 在一个实施例中，功率控制单元102发起使设备130进入深度掉电状态的过程。在一个实施例中，功率控制单元102将设备130保持在寄存器132、134中的上下文数据以及其他上下文数据存储在静态RAM112中。在一个实施例中，静态RAM112是专用RAM。在一个实施例中，上下文数据提供了设备130进入深度掉电状态之前设备130所处的状态的快照。在一个实施例中，静态RAM112与维持功率阱110耦合，该维持功率阱向静态RAM112提供足以维持上下文数据的完整性的功率。然而，维持功率阱110提供给静态RAM112的功率小于操作处理器以及将设备130维持在空闲状态将需要的功率。在一个实施例中，通过将如果设备130留在空闲状态而不是深度掉电状态设备130所本应消耗的能量节省下来，使设备130进入深度掉电状态节省了系统100所利用的能量。在一个实施例中，每个设备(诸如设备130)有其自己的静态RAM和维持轨以用于进入和退出深度掉电状态。

[0022] 在一个实施例中，功率控制单元102的深度掉电逻辑104继续监视设备130以确定设备130何时进入唤醒状态(即，退出深度掉电状态，并且上电以用于正常操作)。在一个实施例中，深度掉电逻辑104监视寄存器132、133、设备驱动程序140等以确定设备130的一个或多个服务何时被请求。在一个实施例中，响应于深度掉电逻辑104确定设备130应当进入唤醒状态，功率控制单元102将上下文数据从静态RAM112加载到设备130中。在一个实施例中，该上下文数据使设备130返回到该设备进入深度掉电状态之前的状态，诸如通过恢复寄存器132、134中的值、设置设备配置等。设备130随后可在唤醒状态执行任务。

[0023] 在一个实施例中，如本文所讨论的，使设备130能够对处理系统中的其他设备或软件(如图6-9中所示的那些)透明地进入和退出深度掉电状态。从而，尽管设备130所处的系统不处于空闲或深度掉电状态，然而在设备130空闲时设备130仍可进入深度掉电状态，由此节省设备130的能量使用。而且，功率控制单元102可进一步基于若干试探来将设备130转换到进入或退出深度掉电状态。

[0024] 在一个实施例中，设备130在系统100的中央处理单元(CPU)核(未示出)进入空闲模式时将进入深度掉电状态。在一个实施例中，因为设备130将最终跟随CPU核进入空闲，所以功率控制单元102将设备130与CPU核步调一致地送入深度掉电状态。在另一实施例中，当设备130空闲，但是CPU核不空闲时，功率控制单元102仍可基于设备130的空闲性质将设备130送入深度掉电状态。但是没有将设备130锁定到CPU核的相同功率状态，使得功率控制单元102能够利用设备130空闲的时间段，设备130空闲的时间段可能明显大于CPU核的时间段，从而造成比CPU核或操作系统的唤醒速率频率显著更低的唤醒速率频率。而且，可以通过允许独立于CPU核进入深度掉电状态来减少设备130的能量使用。

[0025] 类似地，功率控制单元102可根据CPU核，或者独立于CPU核而使设备103进入唤醒

状态。在一个实施例中,当CPU核空闲时,设备130也将最终空闲,并且在CPU核再次变忙之前不会再次变得活动。从而,在一个实施例中,当CPU核退出深度掉电状态时,设备130再次步调一致地跟随CPU核并且也退出深度掉电状态。通过使设备130的部分唤醒时间与CPU核唤醒时间并行,可以减少对CPU的功耗和性能的冲击。在另一个实施例中,即便当CPU核进入唤醒状态时,设备130可留在深度掉电状态中。通过向设备130提供独立的唤醒时间段,功率可以节省达到设备130空闲时间的延长时间段。

[0026] 在一个实施例中,功率控制单元102的深度掉电逻辑104可决定功率控制单元102将哪个试探(即,跟随CPU核的深度掉电状态,或者独立于CPU核而转换到进入/退出深度掉电状态)应用于设备130。在一个实施例中,深度掉电逻辑104可基于设备130的状态、系统100的各组件的状态、设备130所接收的命令的模式或时序、寄存器132、134处的读/写操作的模式或时序、与另一设备或CPU核的事务的模式或时序等的一个或多个度量来“在运行中(on-the-fly)”决定。在一个实施例中,深度掉电逻辑104监视与设备104、其他设备和处理系统在一时间间隔内的各种度量来确定可采用哪种试探(即,跟随CPU核的深度掉电状态,或者独立于CPU核而转换到进入/退出深度掉电状态)来获得最大的功率/性能优化点。在一个实施例中,深度掉电逻辑104周期性地监视各种度量以便更新深度掉电逻辑104目前正在采用哪个试探。例如,深度掉电逻辑104可以确定对于给定间隔,设备130应当跟随处理器的深度掉电状态变化,但是对于不同的间隔确定设备130应当独立地进入和退出深度掉电状态。

[0027] 图2A示出用于发起处理系统的设备的深度掉电的方法200的实施例。在一个实施例中,该方法由处理逻辑(诸如基于硬件的有限状态机)执行。

[0028] 在一个实施例中,监视活动设备(诸如图形控制器、监视器、PCI接口等)(处理框202)。在一个实施例中,按照周期性地间隔或者持续地监视该设备。在一个实施例中,监视所请求的该设备的操作、该设备所执行的操作以及该设备和处理系统中的其他设备之间的事务。而且,还监视内部设备操作(诸如寄存器读/写操作和该设备的功率操作)。在一个实施例中,还监视处理系统中的其他设备的状态(诸如CPU核的状态)。

[0029] 在一个实施例中,处理逻辑确定该设备是否空闲(处理框204)。在一个实施例中,基于一个或多个试探确定该设备空闲。在一个实施例,当还确定CPU核处于空闲或深度掉电状态时,确定该设备空闲。在另一个实施例中,当该设备在给定时间量内(例如,在0.5毫秒内、基于该设备的功率/策略模式的时间间隔等)还未执行操作时确定该设备空闲。

[0030] 在一个实施例中,当处理逻辑确定该设备不空闲时(处理框204),该过程结束。然而,当处理逻辑确定该设备空闲时,处理逻辑将设备上下文数据存储在专用存储器中(处理框206),并且将该设备转换到深度掉电状态(处理框208)。在一个实施例中,该专用存储器是静态RAM存储器。在一个实施例中,该深度掉电状态为C6功率状态。

[0031] 图2B示出用于将设备从深度掉电状态唤醒的方法250的实施例。在一个实施例中,该方法由处理逻辑(诸如基于硬件的有限状态机)执行。

[0032] 在一个实施例中,监视处于深度掉电状态中的设备(诸如图形控制器、监视器、PCI接口等)(处理框252)。如同上面所讨论的,可监视关于该设备的各方面,以及关于其他设备和计算机处理系统组件的各方面。

[0033] 在一个实施例中,处理逻辑确定是否应当将该设备从深度掉电状态唤醒(处理框

254)。在一个实施例中,处理逻辑将根据如本文所讨论的一个或多个试探使该设备进入唤醒状态。在一个实施例中,当CPU核从空闲或深度掉电状态转换到唤醒状态时,该设备将进入唤醒状态。在另一实施例中,处理逻辑将在事务请求该设备的动作时使该设备进入唤醒状态。例如,当另一设备或CPU请求该设备执行操作时,当该设备接收到寄存器读/写请求时等。

[0034] 当处理逻辑确定该设备不应当离开深度掉电状态时,该过程结束。然而,当处理逻辑确定该设备应当唤醒时,处理逻辑发起该设备通电到活动状态(处理框256)并且将上下文数据加载到该设备中(处理框258)。在一个实施例中,该上下文数据是来自设备寄存器的数据以及在所述设备进入深度掉电之前由所述设备保持的其他数据。

[0035] 示例性核架构、处理器和计算机架构

[0036] 处理器核可以用出于不同目的的不同方式在不同的处理器中实现。例如,这样的核的实现可以包括:1)旨在用于通用计算的通用有序核;2)旨在用于通用计算的高性能通用无序核;3)主要旨在用于图形和/或科学(吞吐量)计算的专用核。不同处理器的实现可包括:包括旨在用于通用计算的一个或多个通用有序核和/或旨在用于通用计算的一个或多个通用无序核的CPU;以及2)包括主要旨在用于图形和/或科学(吞吐量)的一个或多个专用核的协处理器。这样的不同处理器导致不同的计算机系统架构,其可包括:1)在与CPU分开的芯片上的协处理器;2)在与CPU相同的封装中但分开的管芯上的协处理器;3)与CPU在相同管芯上的协处理器(在该情况下,这样的协处理器有时被称为诸如集成图形和/或科学(吞吐量)逻辑等专用逻辑,或被称为专用核);以及4)可以将所描述的CPU(有时被称为应用核或应用处理器)、以上描述的协处理器和附加功能包括在同一管芯上的片上系统。接着描述示例性核架构,随后描述示例性处理器和计算机架构。

[0037] 示例性核架构

[0038] 有序和无序核框图

[0039] 图3A是示出根据本发明的实施例的示例性有序流水线以及示例性寄存器重命名的无序发布/执行流水线两者的方框图。图3B是示出根据本发明的实施例的有序架构核的示例性实施例以及包括在处理器中的示例性寄存器重命名的无序发布/执行架构核两者的方框图。图3A-B中的实线框示出了有序流水线和有序核,而虚线框中的可选附加项示出了寄存器重命名的、无序发布/执行流水线和核。给定有序方面是无序方面的子集的情况下,将描述无序方面。

[0040] 在图3A中,处理器流水线300包括取出级302、长度解码级304、解码级306、分配级308、重命名级310、调度(也称为分派或发布)级312、寄存器读取/存储器读取级314、执行级316、写回/存储器写入级318、异常处理级322和提交级324。

[0041] 图3B示出了包括耦合到执行引擎单元350的前端单元330的处理器核390,且执行引擎单元和前端单元两者都耦合到存储器单元370。核390可以是精简指令集合计算(RISC)核、复杂指令集合计算(CISC)核、超长指令字(VLIW)核或混合或替代核类型。作为又一选项,核390可以是专用核,诸如例如网络或通信核、压缩引擎、协处理器核、通用计算图形处理器单元(GPGPU)核、或图形核等等。

[0042] 前端单元330包括耦合到指令高速缓存单元334的分支预测单元332,该指令高速缓存单元334被耦合到指令转换后备缓冲器(TLB)336,该指令转换后备缓冲器336被耦合到

指令取出单元338,指令取出单元338被耦合到解码单元340。解码单元340(或解码器)可解码指令,并生成从原始指令解码出的、或以其他方式反映原始指令的、或从原始指令导出的一个或多个微操作、微代码进入点、微指令、其他指令、或其他控制信号作为输出。解码单元340可使用各种不同的机制来实现。合适的机制的示例包括但不限于查找表、硬件实现、可编程逻辑阵列(OLA)、微代码只读存储器(ROM)等。在一个实施例中,核390包括存储(例如,在解码单元340中或否则在前端单元330内的)某些宏指令的微代码的微代码ROM或其他介质。解码单元340耦合至执行引擎单元350中的重命名/分配器单元352。

[0043] 执行引擎单元350包括重命名/分配器单元352,该重命名/分配器单元352耦合至引退单元354和一个或多个调度器单元356的集合。调度器单元356表示任何数目的不同调度器,包括预留站、中央指令窗等。调度器单元356被耦合到物理寄存器组单元358。每个物理寄存器组单元358表示一个或多个物理寄存器组,其中不同的物理寄存器组存储一种或多种不同的数据类型,诸如标量整数、标量浮点、打包整数、打包浮点、向量整数、向量浮点、状态(例如,作为要执行的下一指令的地址的指令指针)等。在一个实施例中,物理寄存器组单元358包括向量寄存器单元、写掩码寄存器单元和标量寄存器单元。这些寄存器单元可以提供架构向量寄存器、向量掩码寄存器、和通用寄存器。物理寄存器组单元358与引退单元354重叠以示出可以用来实现寄存器重命名和无序执行的各种方式(例如,使用记录器缓冲器和引退寄存器组;使用将来的文件、历史缓冲器和引退寄存器组;使用寄存器映射和寄存器池等等)。引退单元354和物理寄存器组单元358被耦合到执行群集360。执行群集360包括一个或多个执行单元362的集合和一个或多个存储器访问单元364的集合。执行单元362可以对各种类型的数据(例如,标量浮点、打包整数、打包浮点、向量整型、向量浮点)执行各种操作(例如,移位、加法、减法、乘法)。尽管某些实施例可以包括专用于特定功能或功能集合的多个执行单元,但其他实施例可包括全部执行所有函数的仅一个执行单元或多个执行单元。调度器单元356、物理寄存器组单元358和执行群集360被示为可能有多个,因为某些实施例为某些类型的数据/操作(例如,标量整型流水线、标量浮点/打包整型/打包浮点/向量整型/向量浮点流水线,和/或各自具有其自己的调度器单元、物理寄存器单元和/或执行群集的存储器访问流水线——以及在分开的存储器访问流水线的情况下,实现其中仅该流水线的执行群集具有存储器访问单元364的某些实施例)创建分开的流水线。还应当理解,在分开的流水线被使用的情况下,这些流水线中的一个或多个可以为无序发布/执行,并且其余流水线可以为有序发布/执行。

[0044] 存储器访问单元364的集合被耦合到存储器单元370,该存储器单元370包括耦合到数据高速缓存单元374的数据TLB单元372,其中数据高速缓存单元374耦合到二级(L2)高速缓存单元376。在一个示例性实施例中,存储器访问单元364可包括加载单元、存储地址单元和存储数据单元,其中的每一个均耦合至存储器单元370中的数据TLB单元372。指令高速缓存单元334还耦合到存储器单元370中的二级(L2)高速缓存单元376。L2高速缓存单元376被耦合到一个或多个其他级的高速缓存,并最终耦合到主存储器。

[0045] 作为示例,示例性寄存器重命名的、无序发布/执行核架构可以如下实现流水线300:1)指令取出338执行取出和长度解码级302和304;2)解码单元340执行解码级306;3)重命名/分配器单元352执行分配级308和重命名级310;4)调度器单元356执行调度级312;5)物理寄存器组单元358和存储器单元370执行寄存器读取/存储器读取级314;执行群集360

执行执行级316;6)存储器单元370和物理寄存器组单元358执行写回/存储器写入级318;7)各单元可牵涉到异常处理级322;以及8)引退单元354和物理寄存器组单元358执行提交级324。

[0046] 核390可支持一个或多个指令集合(例如,x86指令集合(具有与较新版本一起添加的某些扩展);加利福尼亚州桑尼维尔市的MIPS技术公司的MIPS指令集合;加利福尼州桑尼维尔市的ARM控股的ARM指令集合(具有诸如NEON等可选附加扩展)),其中包括本文中描述的各指令。在一个实施例中,核390包括支持打包数据指令集合扩展(例如,AVX1、AVX2)的逻辑,由此允许被许多多媒体应用使用的操作将使用打包数据来执行。

[0047] 应当理解,核可支持多线程化(执行两个或更多个并行的操作或线程的集合),并且可以按各种方式来完成该多线程化,此各种方式包括时分多线程化、同步多线程化(其中单个物理核为物理核正在同步多线程化的各线程中的每一个线程提供逻辑核)、或其组合(例如,时分取出和解码以及此后诸如用Intel®超线程化技术来同步多线程化)。

[0048] 尽管在无序执行的上下文中描述了寄存器重命名,但应当理解,可以在有序架构中使用寄存器重命名。尽管所示出的处理器的实施例还包括分开的指令和数据高速缓存单元334/374以及共享L2高速缓存单元376,但替换实施例可以具有用于指令和数据两者的单个内部高速缓存,诸如例如一级(L1)内部高速缓存或多个级别的内部缓存。在某些实施例中,该系统可包括内部高速缓存和在核和/或处理器外部的外部高速缓存的组合。或者,所有高速缓存都可以在核和/或处理器的外部。

[0049] 具体的示例性有序核架构

[0050] 图4A-B示出更具体的示例性有序核架构的方块图,该核可以是芯片中的若干逻辑块(包括具有相同类型和/或不同类型的其他核)中的一个。这些逻辑块通过高带宽的互连网络(例如,环形网络)与某些固定的功能逻辑、存储器I/O接口和其它必要的I/O逻辑通信,这取决于应用。

[0051] 图4A是根据本发明的各实施例的单个处理器核连同它与管芯上互连网络402的连接以及其二级(L2)高速缓存404的本地子集的框图。在一个实施例中,指令解码器400支持具有打包数据指令集扩展的x86指令集。L1高速缓存406允许对标量和向量单元中的高速缓存存储器的低等待时间访问。尽管在一个实施例中(为了简化设计),标量单元408和向量单元410使用分开的寄存器集合(分别为标量寄存器412和向量寄存器414),并且在这些寄存器之间转移的数据被写入到存储器并随后从一级(L1)高速缓存406读回,但是本发明的替换实施例可以使用不同的方法(例如使用单个寄存器集合或包括允许数据在这两个寄存器组之间传输而无需被写入和读回的通信路径)。

[0052] L2高速缓存的本地子集404是全局L2高速缓存的一部分,该全局L2高速缓存被划分成多个分开的本地子集,即每个处理器核一个本地子集。每个处理器核具有到其自己的L2高速缓存404的本地子集的直接访问路径。被处理器核读出的数据被存储在其L2高速缓存子集404中,并且可以与其他处理器核访问其自己的本地L2高速缓存子集并行地被快速访问。被处理器核写入的数据被存储在其自己的L2高速缓存子集404中,并在必要的情况下从其它子集清除。环形网络确保共享数据的一致性。环形网络是双向的,以允许诸如处理器核、L2高速缓存和其它逻辑块之类的代理在芯片内彼此通信。每个环形数据路径为每个方向1012位宽。

[0053] 图4B是根据本发明的各实施例的图4A中的处理器核的一部分的展开图。图4B包括作为L1高速缓存404的L1数据高速缓存406A部分,以及关于向量单元410和向量寄存器414的更多细节。具体地说,向量单元410是16宽向量处理单元(VPU)(见16宽ALU428),该单元执行整型、单精度浮点以及双精度浮点指令中的一个或多个。该VPU通过混合单元420支持对寄存器输入的混合、通过数值转换单元422A-B支持数值转换,并通过复制单元424支持对存储器输入的复制。写掩码寄存器426允许断言所得的向量写入。

[0054] 具有集成存储器控制器和图形器件的处理器

[0055] 图5是根据本发明的实施例的可具有一个以上核、可具有集成存储器控制器、并且可具有集成图形的处理器500的方框图。图5的实线框示出了处理器500,处理器500具有单个核502A、系统代理(agent)510、一组一个或多个总线控制器单元516,而可选附加的虚线框示出了替代的处理器500,其具有多个核502A-N、系统代理单元510中的一组一个或多个集成存储器控制器单元514以及专用逻辑508。

[0056] 因此,处理器500的不同实现可包括:1) CPU,其中专用逻辑508是集成图形和/或科学(吞吐量)逻辑(其可包括一个或多个核),并且核502A-N是一个或多个通用核(例如,通用的有序核、通用的无序核、这两者的组合);2) 协处理器,其中核502A-N是主要旨在用于图形和/或科学(吞吐量)的多个专用核;以及3) 协处理器,其中核502A-N是多个通用有序核。因此,处理器500可以是通用处理器、协处理器或专用处理器,诸如例如网络或通信处理器、压缩引擎、图形处理器、GPGPU(通用图形处理单元)、高吞吐量的集成众核(MIC)协处理器(包括30个或更多核)、或嵌入式处理器等。该处理器可以被实现在一个或多个芯片上。处理器500可以是一个或多个衬底的一部分,和/或可以使用诸如例如BiCMOS、CMOS或NMOS等的多个加工技术中的任何一个技术将其实现在一个或多个衬底上。

[0057] 存储器层次结构包括在各核内的一个或多个级别的高速缓存、一个或多个共享高速缓存单元506的集合、以及耦合至集成存储器控制器单元514的集合的外部存储器(未示出)。该共享高速缓存单元506的集合可以包括一个或多个中间级高速缓存,诸如二级(L2)、三级(L3)、四级(L4)或其他级别的高速缓存、末级高速缓存(LLC)、和/或其组合。尽管在一个实施例中,基于环的互连单元512将集成图形逻辑508、共享高速缓存单元506的集合以及系统代理单元510/集成存储器控制器单元514互连,但替代实施例可使用任何数量的公知技术来将这些单元互连。在一个实施例中,在一个或多个高速缓存单元506与核502-A-N之间维持一致性。

[0058] 在某些实施例中,核502A-N中的一个或多个核能够多线程化。系统代理510包括协调和操作核502A-N的那些组件。系统代理单元510可包括例如功率控制单元(PCU)和显示单元。PCU可以是或包括调整核502A-N和集成图形逻辑508的功率状态所需的逻辑和组件。显示单元用于驱动一个或多个外部连接的显示器。

[0059] 核502A-N在架构指令集合方面可以是同构的或异构的;即,这些核502A-N中的两个或更多个核可能能够执行相同的指令集合,而其他核可能能够执行该指令集合的仅仅子集或不同的指令集合。

[0060] 示例性计算机架构

[0061] 图6-9是示例性计算机架构的方块图。本领域已知的对膝上型设备、台式机、手持PC、个人数字助理、工程工作站、服务器、网络设备、网络集线器、交换机、嵌入式处理器、数

字信号处理器(DSP)、图形设备、视频游戏设备、机顶盒、微控制器、蜂窝电话、便携式媒体播放器、手持设备以及各种其他电子设备的其他系统设计和配置也是合适的。一般来说，能够纳入本文中所公开的处理器和/或其它执行逻辑的多个系统和电子设备一般都是合适的。

[0062] 现在参考图6,所示出的是根据本发明一个实施例的系统600的框图。系统600可以包括一个或多个处理器610、615,这些处理器耦合到控制器中枢620。在一个实施例中,控制器中枢620包括图形存储器控制器中枢(GMCH)690和输入/输出中枢(IOH)650(其可以在分开的芯片上);GMCH690包括存储器和图形控制器,存储器640和协处理器645耦合到该存储器和图形控制器;IOH650将输入/输出(I/O)设备660耦合到GMCH690。替换地,存储器和图形控制器中的一个或两个集成在处理器(如本文中所描述的)内,存储器640和协处理器645直接耦合到处理器610,且控制器中枢620与IOH650在单一芯片中。

[0063] 附加处理器615的任选性质用虚线表示在图6中。每一处理器610、615可包括本文中描述的处理核中的一个或多个,并且可以是处理器500的某一版本。

[0064] 存储器640可以是例如动态随机存取存储器(DRAM)、相变存储器(PCM)或这两者的组合。对于至少一个实施例,控制器中枢620经由诸如前端总线(FSB)之类的多分支总线(multi-drop bus)、诸如快速通道互连(QPI)之类的点对点接口、或者类似的连接695与处理器610、615进行通信。

[0065] 在一个实施例中,协处理器645是专用处理器,诸如例如高吞吐量MIC处理器、网络或通信处理器、压缩引擎、图形处理器、GPGPU、或嵌入式处理器等等。在一个实施例中,控制器中枢620可以包括集成图形加速器。

[0066] 按照包括体系结构、微体系结构、热、功耗特征等等优点的度量谱,物理资源610、615之间存在各种差别。

[0067] 在一个实施例中,处理器610执行控制一般类型的数据处理操作的指令。嵌入在这些指令中的可以是协处理器指令。处理器610识别如具有应当由附连的协处理器645执行的类型的这些协处理器指令。因此,处理器610在协处理器总线或者其他互连上将这些协处理器指令(或者表示协处理器指令的控制信号)发布到协处理器645。协处理器645接受并执行所接收的协处理器指令。

[0068] 现在参照图7,所示出的是根据本发明实施例的更具体的第一示例性系统700的框图。如图7所示,多处理器系统700是点对点互连系统,并包括经由点对点互连750耦合的第一处理器770和第二处理器780。处理器770和780中的每一个都可以是处理器500的某一版本。在本发明的一个实施例中,处理器770和780分别是处理器610和615,而协处理器738是协处理器645。在另一实施例中,处理器770和780分别是处理器610和协处理器645。

[0069] 处理器770和780被示为分别包括集成存储器控制器(IMC)单元772和782。处理器770还包括作为其总线控制器单元的一部分的点对点(P-P)接口776和778;类似地,第二处理器780包括点对点接口786和788。处理器770、780可以使用点对点(P-P)电路778、788经由P-P接口750来交换信息。如图7所示,IMC772和782将各处理器耦合至相应的存储器,即存储器732和存储器734,这些存储器可以是本地附连至相应的处理器的主存储器的一部分。

[0070] 处理器770、780可各自经由使用点对点接口电路776、794、786、798的各个P-P接口752、754与芯片组790交换信息。芯片组790可以可选地经由高性能接口739与协处理器738交换信息。在一个实施例中,协处理器738是专用处理器,诸如例如高吞吐量MIC处理器、网

络或通信处理器、压缩引擎、图形处理器、GPGPU、或嵌入式处理器等等。

[0071] 共享高速缓存(未示出)可以被包括在任一处理器之内或被包括两个处理器外部但仍经由P-P互连与这些处理器连接,从而如果将某处理器置于低功率模式时,可将任一处理器或两个处理器的本地高速缓存信息存储在该共享高速缓存中。

[0072] 芯片组790可经由接口796耦合至第一总线716。在一个实施例中,第一总线716可以是外围部件互连(PCI)总线,或诸如PCI Express总线或其它第三代I/O互连总线之类的总线,但本发明的范围并不受此限制。

[0073] 如图7所示,各种I/O设备714可以连同总线桥718耦合到第一总线716,总线桥718将第一总线716耦合至第二总线720。在一个实施例中,诸如协处理器、高吞吐量MIC处理器、GPGPU的处理器、加速器(诸如例如图形加速器或数字信号处理器(DSP)单元)、场可编程门阵列或任何其他处理器的一个或多个附加处理器715被耦合到第一总线716。在一个实施例中,第二总线720可以是低引脚计数(LPC)总线。各种设备可以被耦合至第二总线720,在一个实施例中这些设备包括例如键盘/鼠标722、通信设备727以及诸如可包括指令/代码和数据730的盘驱动器或其它大容量存储设备的存储单元728。此外,音频I/O724可以被耦合至第二总线720。注意,其它架构是可能的。例如,取代图7的点对点架构,系统可以实现多点总线或其它这类架构。

[0074] 现在参考图8,示出了根据本发明的一个实施例的更具体的第二示例性系统800的方框图。图7和图8中的相同部件用相同附图标记表示,并从图8中省去了图7中的某些方面,以避免使图8的其它方面变得难以理解。

[0075] 图8示出处理器770、780可分别包括集成存储器和I/O控制逻辑("CL")772和782。因此,CL772、782包括集成存储器控制器单元并包括I/O控制逻辑。图8不仅示出存储器732、734耦合至CL772、782,而且还示出I/O设备814也耦合至控制逻辑772、782。传统I/O设备815被耦合至芯片组790。

[0076] 现在参照图9,所示出的是根据本发明一个实施例的SoC900的框图。在图5中,相似的部件具有同样的附图标记。另外,虚线框是更先进的SoC的可选特征。在图9中,互连单元(多个)902被耦合至:应用处理器910,该应用处理器包括一个或多个核202A-N的集合以及共享高速缓存单元506;系统代理单元510;总线控制器单元516;集成存储器控制器单元514;一组或一个或多个协处理器920,其可包括集成图形逻辑、图像处理器、音频处理器和视频处理器;静态随机存取存储器(SRAM)单元930;直接存储器存取(DMA)单元932;以及用于耦合至一个或多个外部显示器的显示单元940。在一个实施例中,协处理器920包括专用处理器,诸如例如网络或通信处理器、压缩引擎、GPGPU、高吞吐量MIC处理器、或嵌入式处理器等等。

[0077] 本文公开的机制的各实施例可以被实现在硬件、软件、固件或这些实现方法的组合中。本发明的实施例可实现为在可编程系统上执行的计算机程序或程序代码,该可编程系统包括至少一个处理器、存储系统(包括易失性和非易失性存储器和/或存储元件)、至少一个输入设备以及至少一个输出设备。

[0078] 可将程序代码(诸如图7中示出的代码730)应用于输入指令,以执行本文描述的各功能并生成输出信息。输出信息可以按已知方式被应用于一个或多个输出设备。为了本申请的目的,处理系统包括具有诸如例如数字信号处理器(DSP)、微控制器、专用集成电路

(ASIC)或微处理器之类的处理器的任何系统。

[0079] 程序代码可以用高级程序化语言或面向对象的编程语言来实现,以便与处理系统通信。程序代码也可以在需要的情况下用汇编语言或机器语言来实现。事实上,本文中描述的机制不仅限于任何特定编程语言的范围。在任一情形下,语言可以是编译语言或解释语言。

[0080] 至少一个实施例的一个或多个方面可以由存储在机器可读介质上的表示性指令来实现,指令表示处理器中的各种逻辑,指令在被机器读取时使得该机器制作用于执行本文所述的技术的逻辑。被称为“IP核”的这些表示可以被存储在有形的机器可读介质上,并被提供给多个客户或生产设施以加载到实际制造该逻辑或处理器的制造机器中。

[0081] 这样的机器可读存储介质可以包括但不限于通过机器或设备制造或形成的物品的非瞬态、有形安排,其包括存储介质,诸如硬盘;任何其它类型的盘,包括软盘、光盘、紧致盘只读存储器(CD-ROM)、紧致盘可重写(CD-RW)的以及磁光盘;半导体器件,例如只读存储器(ROM)、诸如动态随机存取存储器(DRAM)和静态随机存取存储器(SRAM)的随机存取存储器(RAM)、可擦除可编程只读存储器(E PROM)、闪存、电可擦除可编程只读存储器(E EPROM);相变存储器(PCM);磁卡或光卡;或适于存储电子指令的任何其它类型的介质。

[0082] 因此,本发明的各实施例还包括非瞬态、有形机器可读介质,该介质包含指令或包含设计数据,诸如硬件描述语言(HDL),它定义本文中描述的结构、电路、装置、处理器和/或系统特性。这些实施例也被称为程序产品。

[0083] 仿真(包括二进制变换、代码变形等)

[0084] 在某些情况下,指令转换器可用来将指令从源指令集转换至目标指令集。例如,指令转换器可以变换(例如使用静态二进制变换、包括动态编译的动态二进制变换)、变形、仿真或以其它方式将指令转换成将由核来处理的一个或多个其它指令。指令转换器可以用软件、硬件、固件、或其组合实现。指令转换器可以在处理器上、在处理器外、或者部分在处理器上部分在处理器外。

[0085] 图10是根据本发明的实施例的对比使用软件指令变换器将源指令集中的二进制指令变换成目标指令集中的二进制指令的框图。在所示的实施例中,指令转换器是软件指令转换器,但作为替代该指令转换器可以用软件、固件、硬件或其各种组合来实现。图10示出了用高级语言1002的程序可以使用x86编译器1004来编译,以生成可以由具有至少一个x86指令集核的处理器1016原生执行的x86二进制代码1006。具有至少一个x86指令集核的处理器1016表示任何处理器,这些处理器能通过兼容地执行或以其他方式处理以下内容来执行与具有至少一个x86指令集核的英特尔处理器基本相同的功能:1)英特尔x86指令集核的指令集的本质部分,或2)被定向为在具有至少一个x86指令集核的英特尔处理器上运行的应用或其它程序的目标代码版本,以便取得与具有至少一个x86指令集核的英特尔处理器基本相同的结果。x86编译器1004表示用于生成x86二进制代码1006(例如,目标代码)的编译器,该二进制代码0可通过或不通过附加的链接处理在具有至少一个x86指令集核的处理器1016上执行。类似地,图10示出用高级语言1002的程序可以使用替代的指令集编译器1008来编译,以生成可以由不具有至少一个x86指令集核的处理器1014(例如具有执行加利福尼亚州桑尼维尔市的MIPS技术公司的MIPS指令集,和/或执行加利福尼亚州桑尼维尔市的ARM控股公司的ARM指令集的核的处理器)原生执行的替代指令集二进制代码1010。指令

转换器1012被用来将x86二进制代码1006转换成可以由不具有x86指令集核的处理器1014原生执行的代码。该转换后的代码不大可能与替换性指令集二进制代码1010相同,因为能够这样做的指令转换器难以制造;然而,转换后的代码将完成一般操作并由来自替代指令集的指令构成。因此,指令转换器1012通过仿真、模拟或任何其它过程来表示允许不具有x86指令集处理器或核的处理器或其它电子设备执行x86二进制代码1006的软件、固件、硬件或其组合。

[0086] 同样,尽管附图中的流程图示出本发明的某些实施例的特定操作顺序,应该理解该顺序是示例性的(例如,可选实施例可按不同顺序执行操作、组合某些操作、使某些操作重叠等)。

[0087] 在以上描述中,出于解释的目的,阐明了众多具体细节以提供对本发明的实施例的透彻理解。然而,对本领域技术人员而言将是明显的是,不用这些具体细节中的一些也可实践一个或多个其他实施例。提供所描述的具体实施例不是为了限制本发明而是为了说明本发明的实施例。本发明的范围不是由所提供的具体示例确定,而是仅由所附权利要求确定。

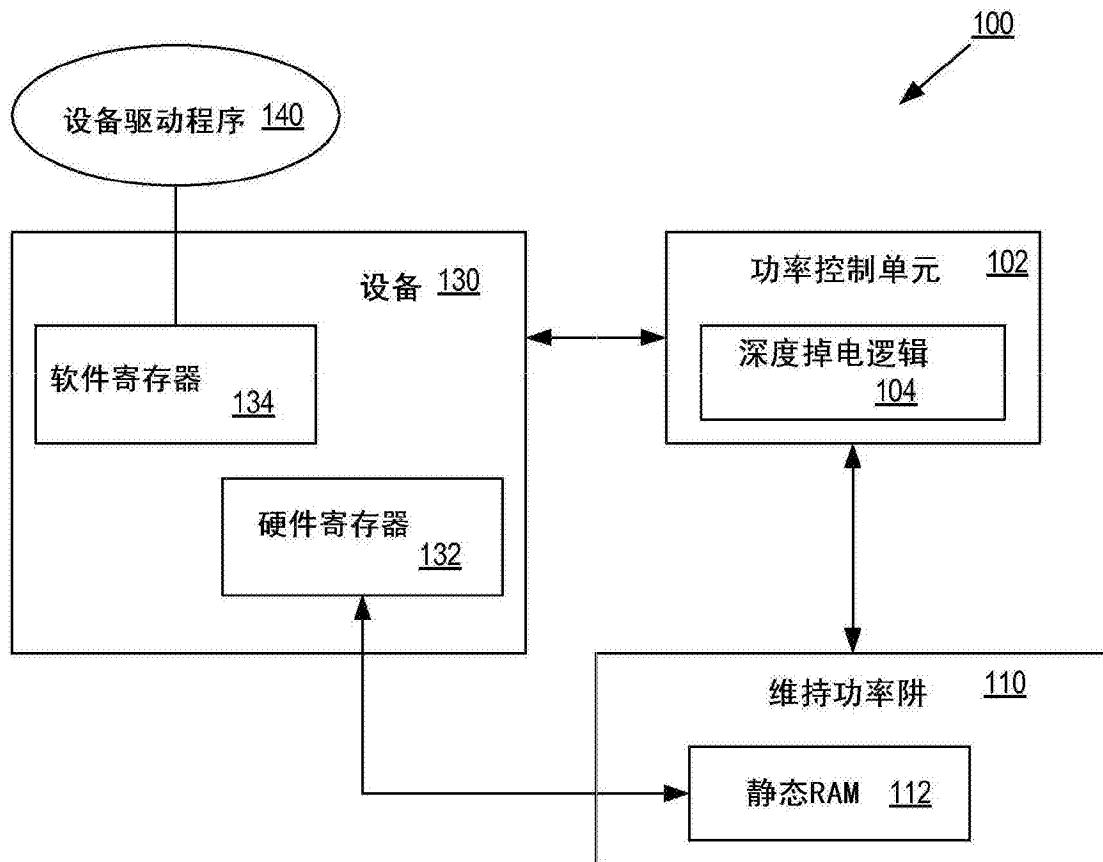


图1

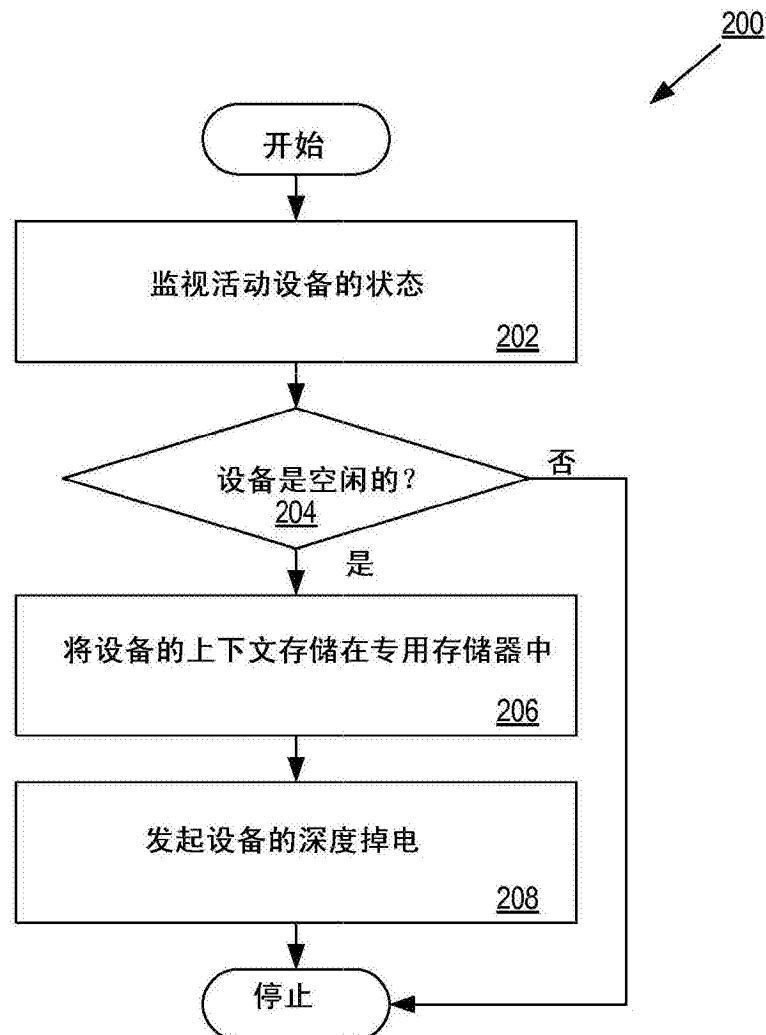


图2A

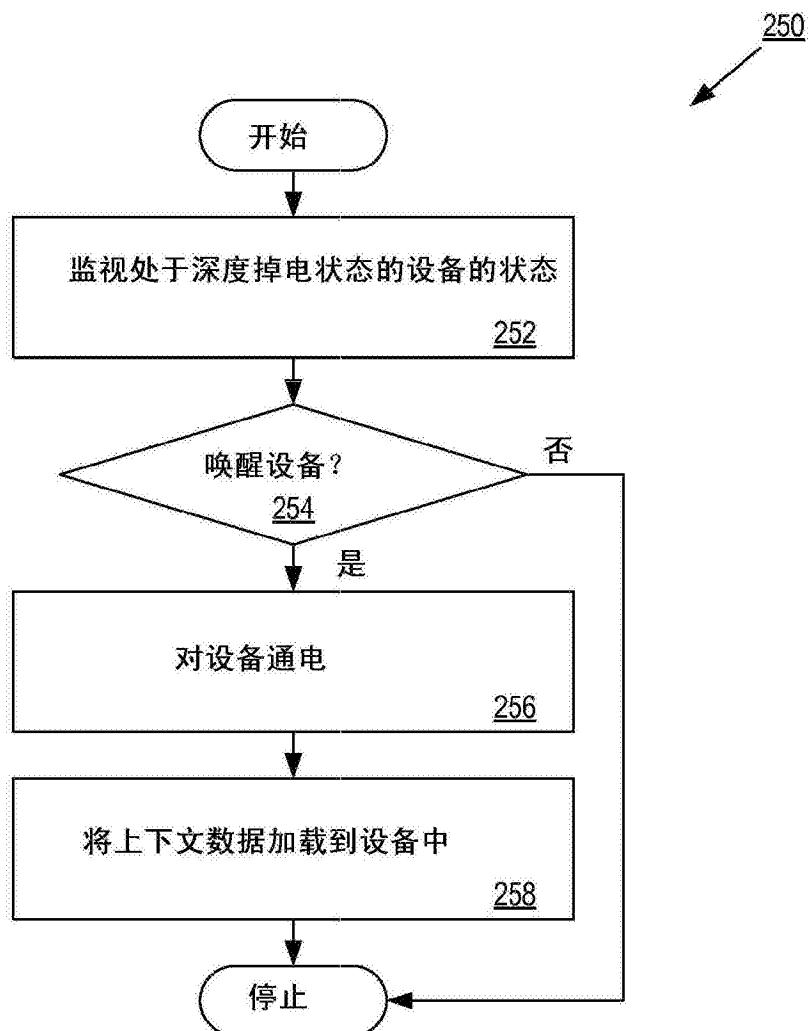


图2B

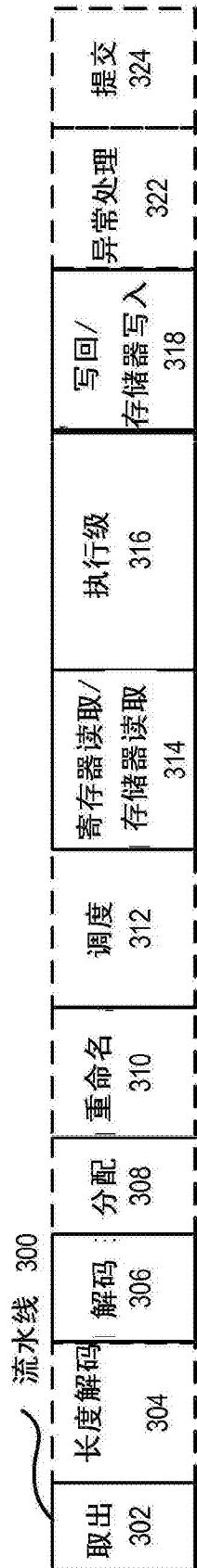


图3A

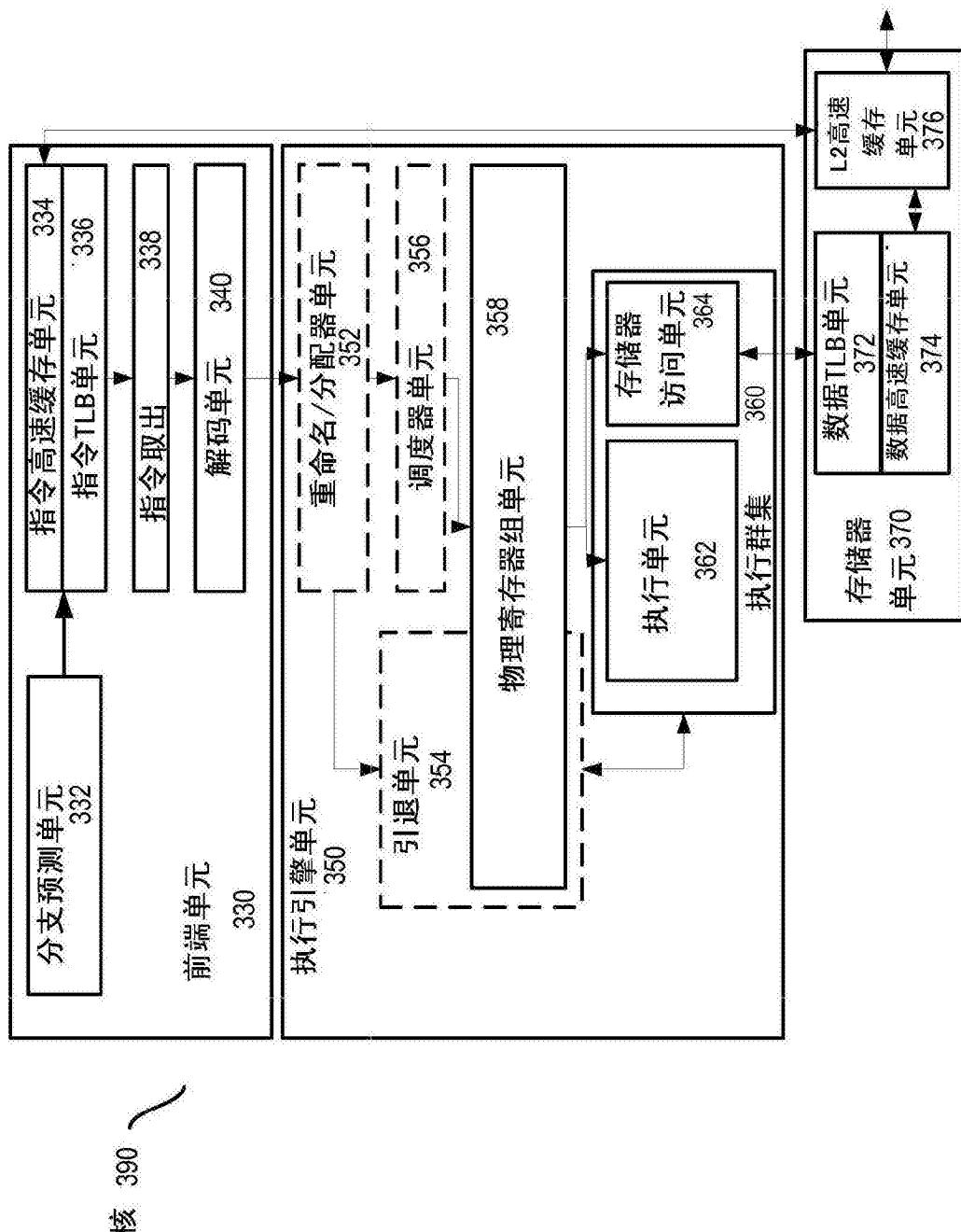


图3B

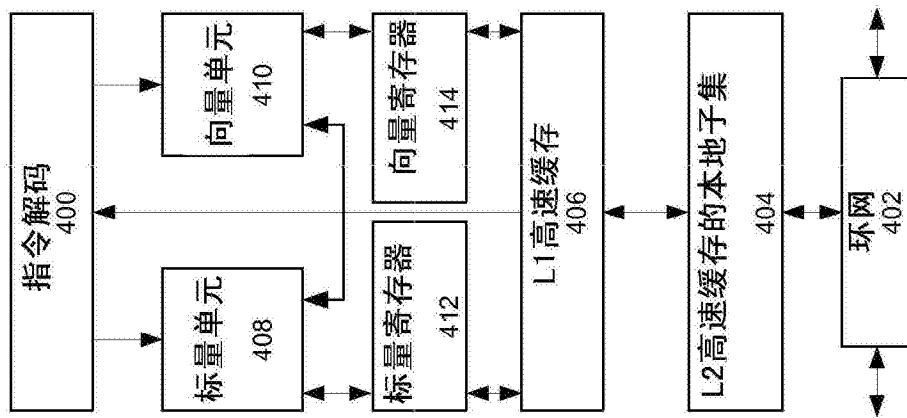


图 4A

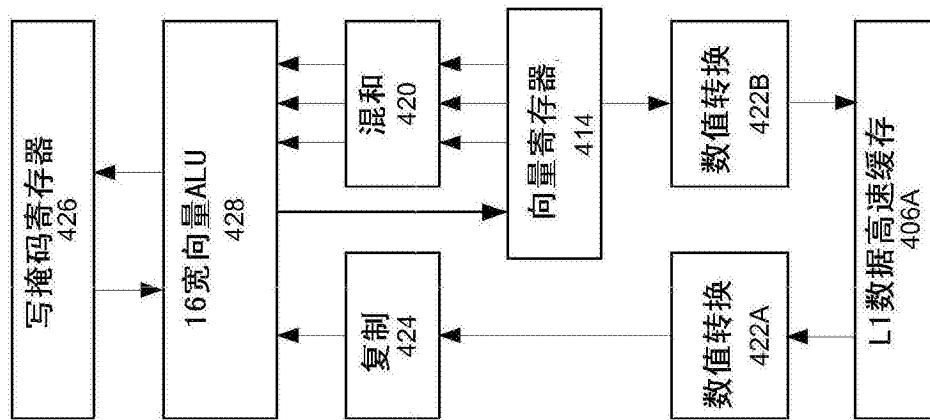


图 4B

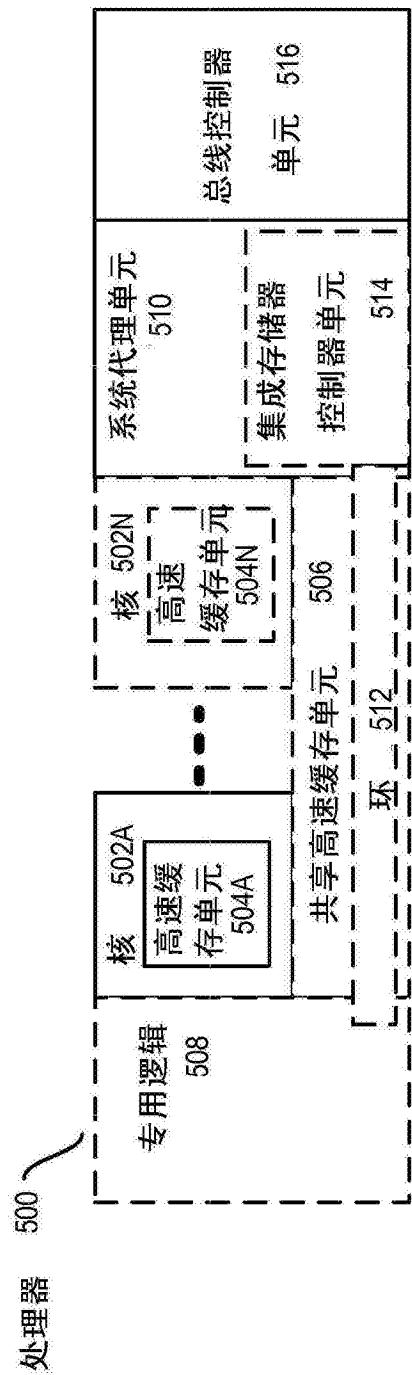


图5

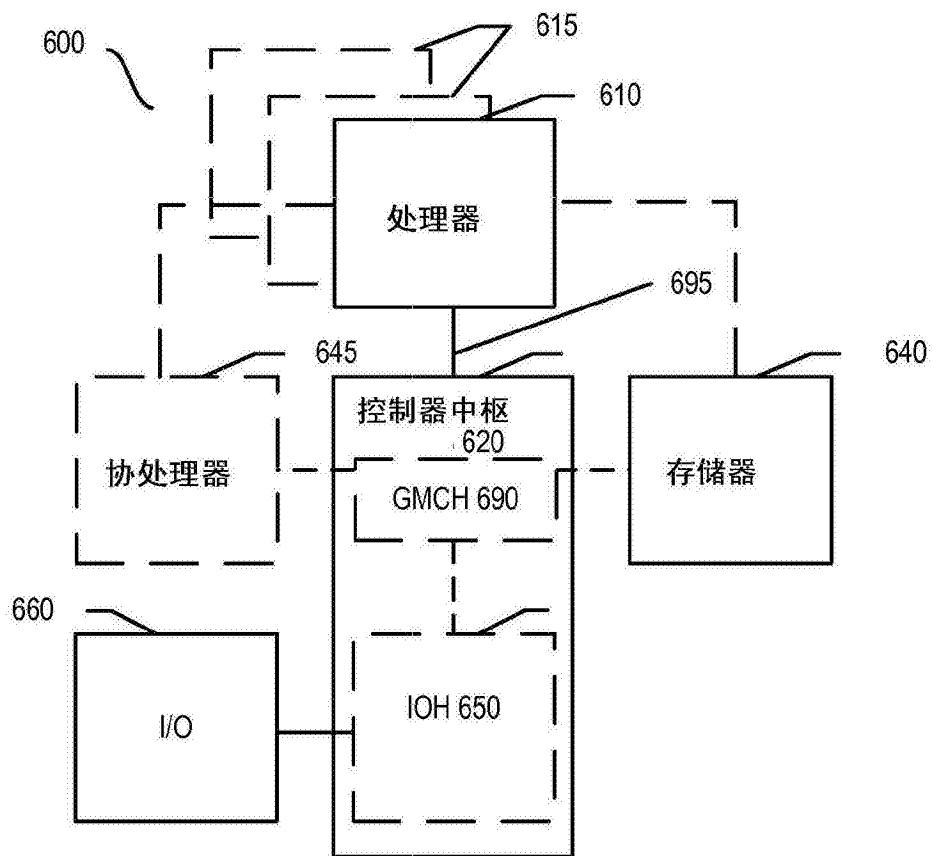


图6

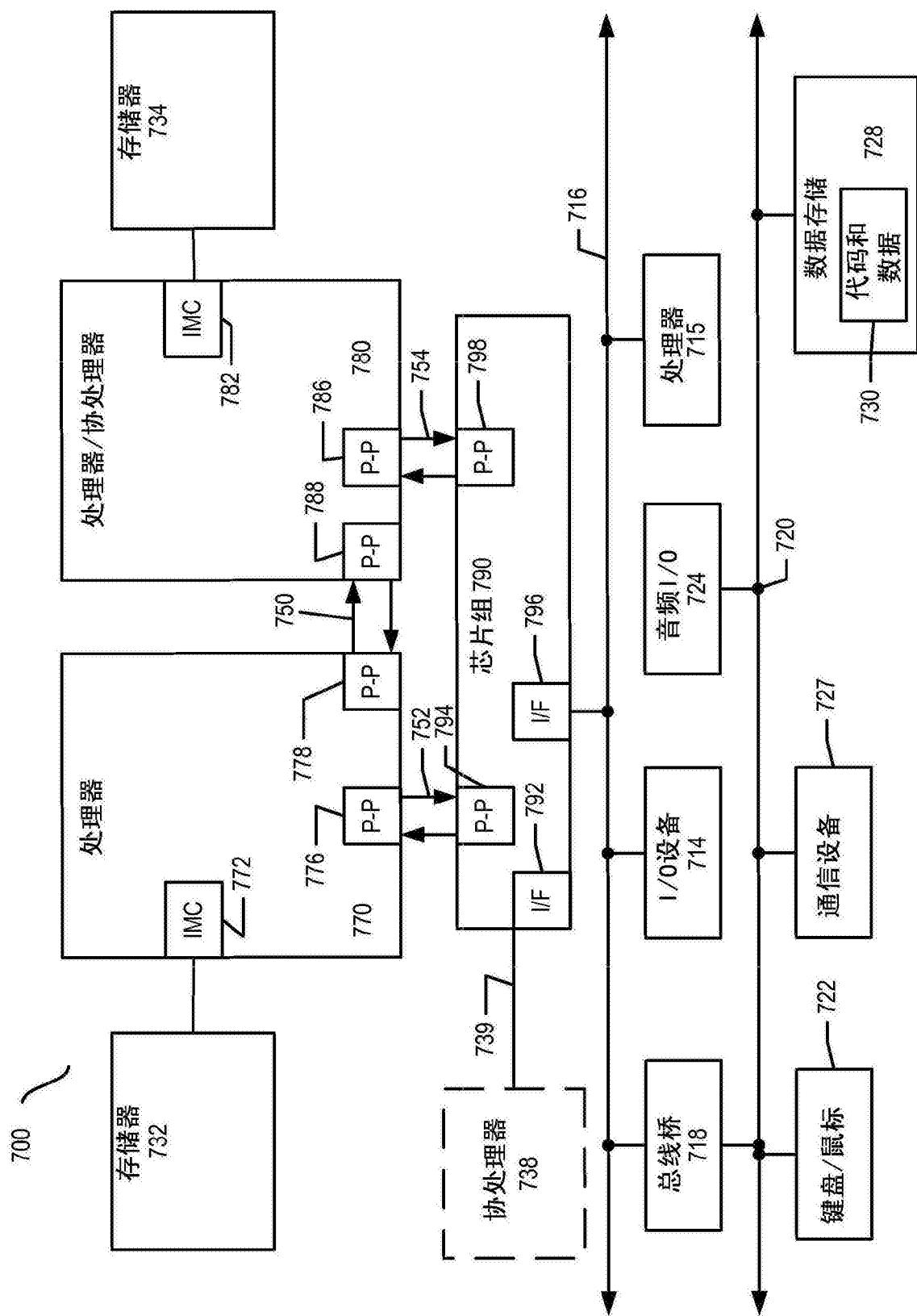


图 7

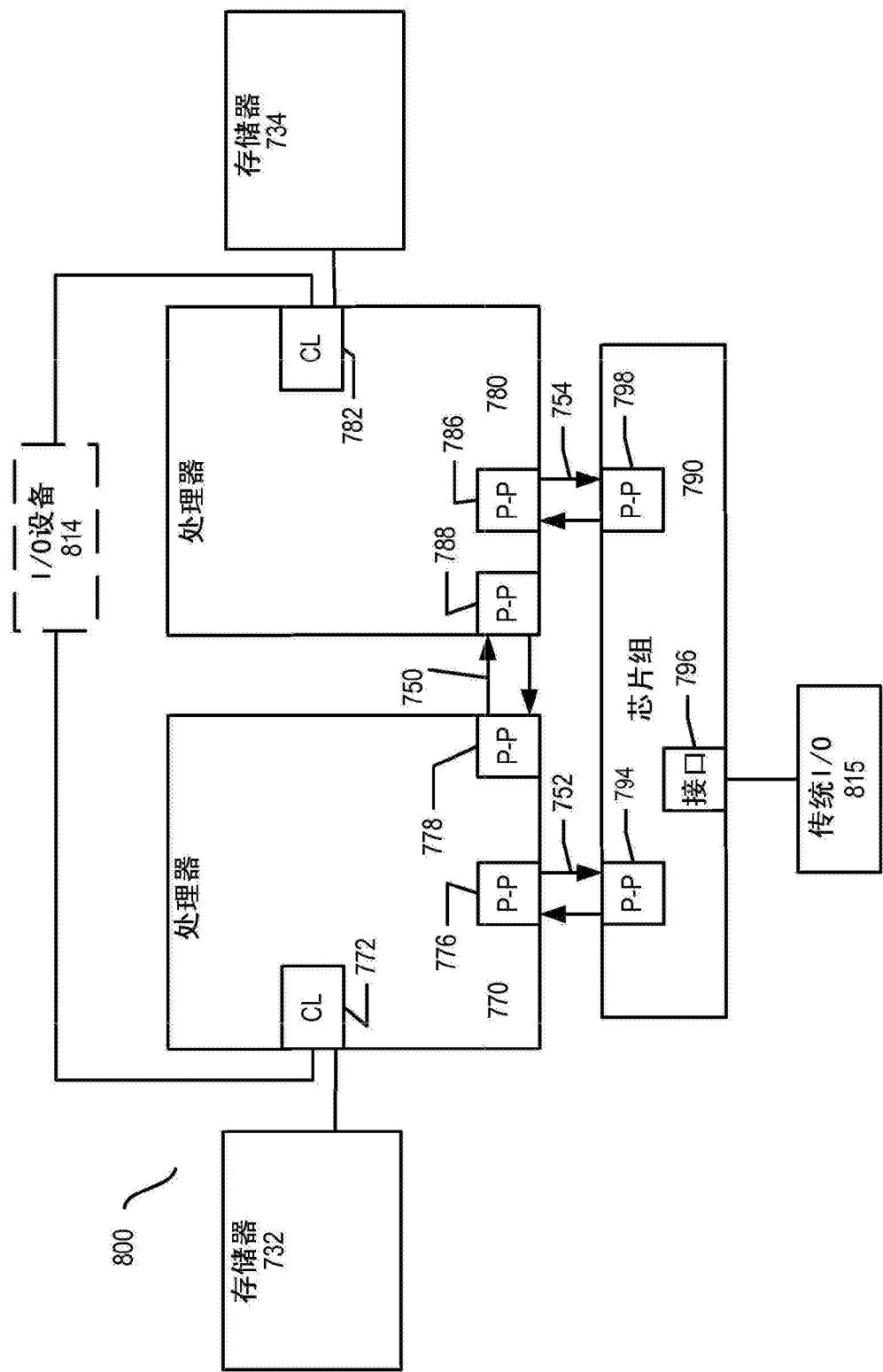


图8

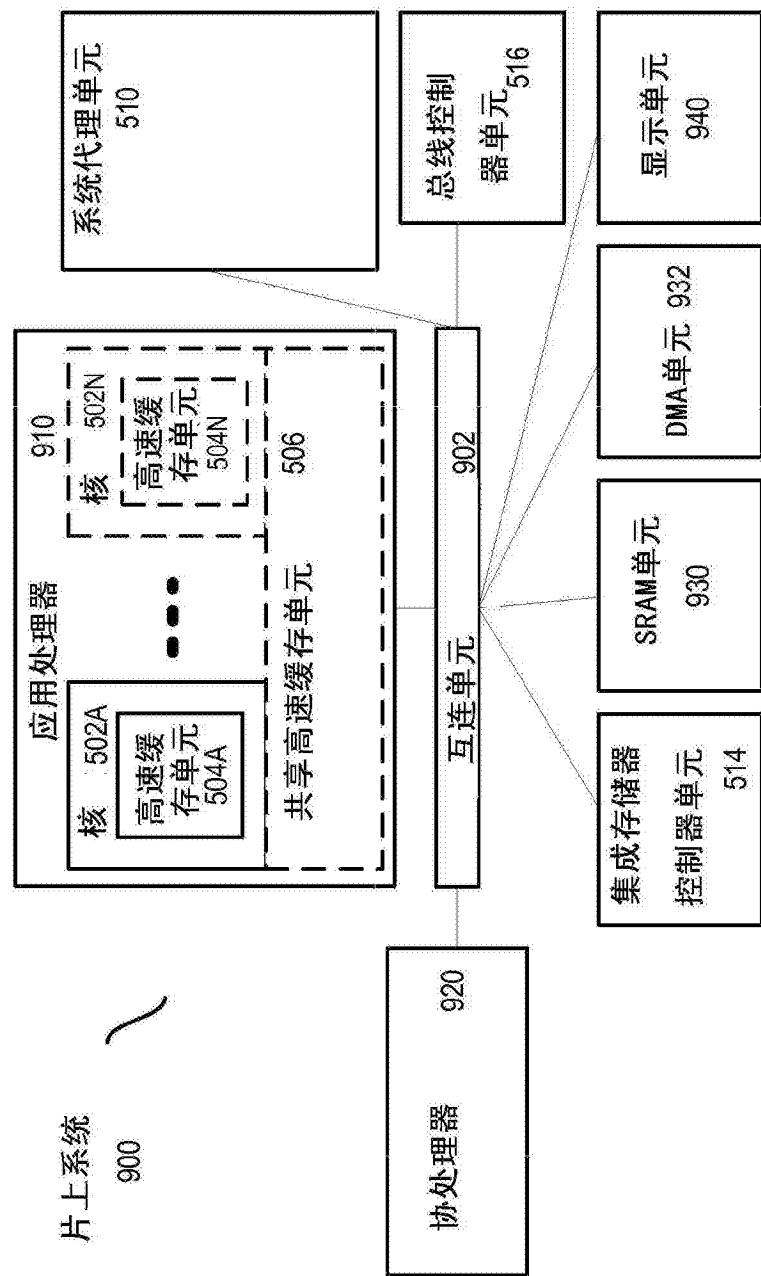


图9

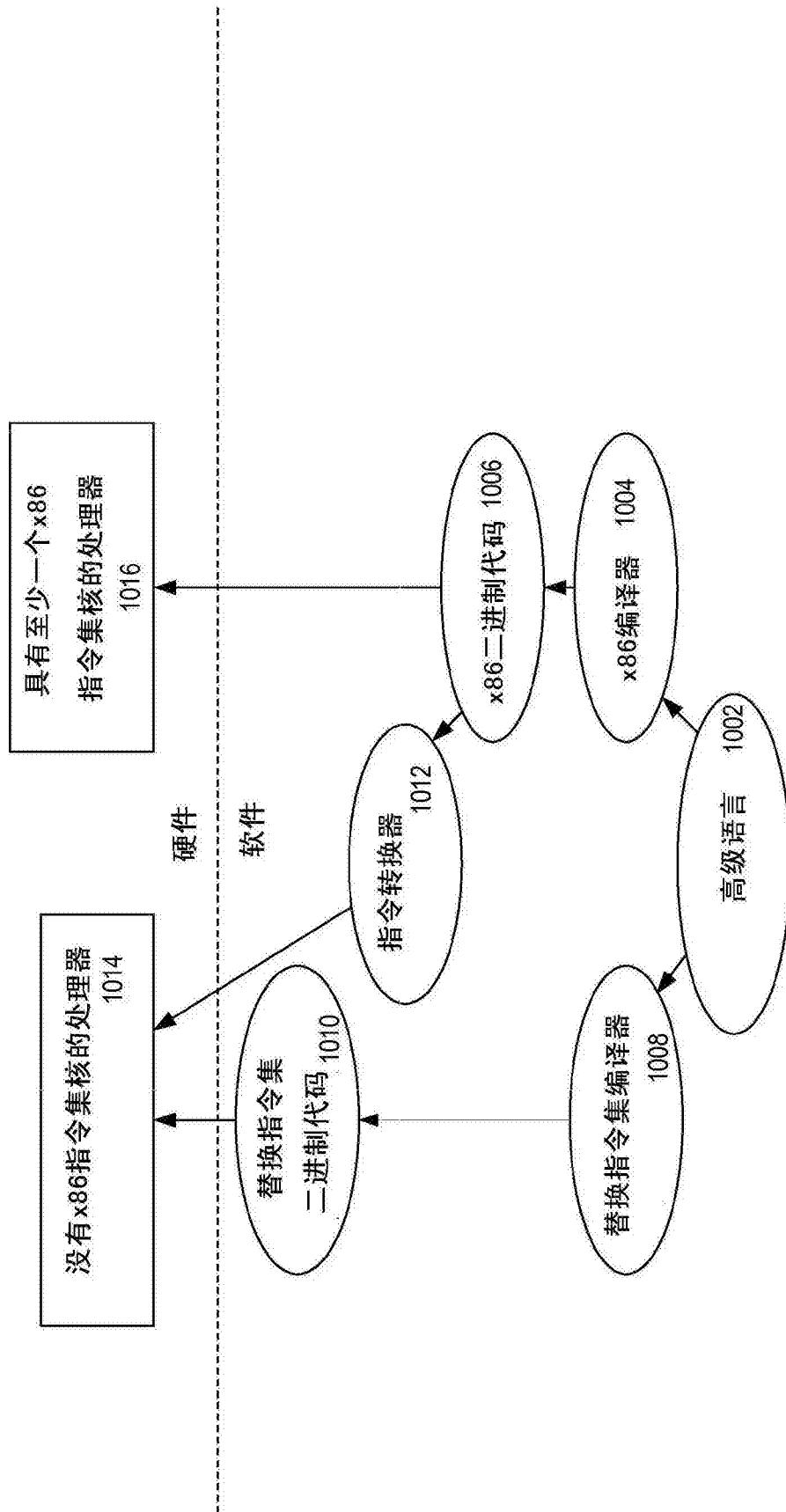


图 10