(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 109074260 A
(43)申请公布日 2018.12.21

(21)申请号 201780026331.4

(22)申请日 2017.04.25

(30)优先权数据
62/328,976 2016.04.28 US
15/224,469 2016.07.29 US

(85)PCT国际申请进入国家阶段日
2018.10.26

(86)PCT国际申请的申请数据
PCT/US2017/029224 2017.04.25

(87)PCT国际申请的公布数据
WO2017/189463 EN 2017.11.02

(71)申请人 微软技术许可有限责任公司
地址 美国华盛顿州

(72)发明人 A·L·史密斯 J·S·格雷

(74)专利代理机构 北京市金杜律师事务所 11256
代理人 王茂华

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

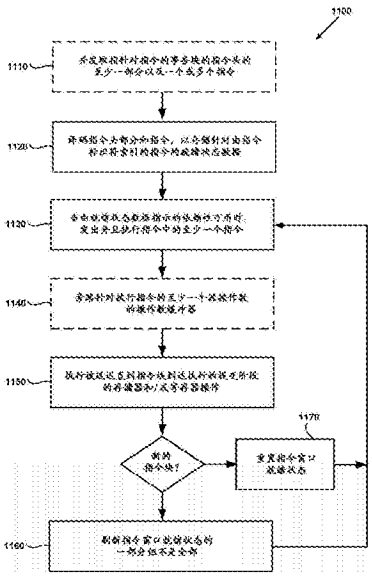
权利要求书2页 说明书23页 附图13页

(54)发明名称

乱序的基于块的处理器和指令调度器

(57)摘要

公开了用于实现包括现场可编程门阵列实现的基于块的处理器的装置和方法。在所公开的技术的一个示例中，基于块的处理器包括被配置为针对指令的事务块生成译码的就绪依赖性的指令译码器，其中指令中的每个指令与编码在事务块中的不同指令标识符相关联。处理器还包括被配置为从指令的事务块的集合乱序地发出指令的指令调度器。指令基于确定针对指令的译码的就绪状态依赖性被满足而被发出。确定包括访问具有译码的就绪依赖性的存储装置，该译码的就绪依赖性利用被编码在指令的事务块中的相应指令标识符而被索引。



CN 109074260 A

1.一种包括基于块的处理器的装置,所述基于块的处理器包括:

指令译码器,其被配置为针对指令的事务块的至少部分生成译码的就绪依赖性,所述指令中的每个指令与编码在所述事务块中的不同指令标识符相关联;以及

指令调度器,其被配置为从指令的所述事务块不按程序顺序地发出指令,其中:

所述指令基于确定针对所述指令的所述译码的就绪依赖性被满足而被发出,并且

所述确定包括:访问存储装置,所述存储装置存储使用被编码在指令的所述事务块中的相应指令标识符的所述译码的就绪依赖性。

2.根据权利要求1所述的装置,其中所述确定包括:使用由执行的指令信号通知的指令标识符,来生成用于访问所述存储装置的索引。

3.根据权利要求1或权利要求2所述的装置,其中所述指令处理器还包括指令取指单元,所述指令取指单元被配置为:取指针对指令的所述事务块的头的至少部分,并且并发地取指指令的所述事务块的指令的至少部分。

4.根据权利要求3所述的装置,其中所述指令取指单元包括存储所取指的头的第一块存储器、和存储所取指的指令的第二块存储器。

5.根据权利要求1-4中的任一项所述的装置,其中所述指令调度器被耦合到数据操作数缓冲器,所述数据操作数缓冲器存储用于由所述指令在后续时钟周期中的执行而生成的数据。

6.根据权利要求5所述的装置,还包括旁路逻辑,所述旁路逻辑允许数据操作数被转发以用于由指令在紧跟着的后续时钟周期中的执行,所述旁路逻辑允许所述数据操作数被转发而不将所述数据操作数存储在所述数据操作数缓冲器中。

7.根据权利要求5所述的装置,其中所述数据操作数缓冲器被配置为每个时钟周期为不超过一个的指令来存储操作数数据,所述装置还包括:

旁路逻辑,其允许在不同数据操作数被存储在所述数据操作数缓冲器中时、针对不同指令的数据操作数在相同时钟周期中被转发到所述处理器的执行单元。

8.一种被配置为执行基于块的处理器指令集的可重新配置逻辑器件,所述器件包括:

多个多输入查找表(LUT);

指令高速缓存,其被配置为从指令块接收指令,所述指令由所述处理器从耦合到所述可重新配置逻辑器件的存储器取指;

指令调度器,其被配置为存储由编码在指令块中的指令标识符索引的就绪状态数据,所述指令调度器还被配置为在所述就绪状态数据指示针对指令的所有依赖性被满足时发出指令;

一个或多个执行单元,其被配置为执行由发出的指令指定的操作;

数据高速缓存,其被配置为存储通过使用所述执行单元执行指令而从所述存储器读取的数据和/或向所述存储器写入的数据,所述数据高速缓存被配置为延迟完成存储器操作直到所述指令块到达执行的提交阶段;以及

寄存器文件,其被配置为存储由所述指令的寄存器操作数指定的数据。

9.根据权利要求8所述的可重新配置逻辑器件,其中所述指令调度器和所述寄存器文件利用使用所述多个LUT的部分形成的随机存取存储器(RAM)来实现。

10.根据权利要求8或权利要求9所述的可重新配置逻辑器件,其中所述指令调度器被

耦合到：

译码的指令字存储器，其被配置为针对所接收的所述指令的至少部分存储译码的指令控制数据；以及

多个操作数缓冲器，其被配置为存储用于执行所接收的所述指令的操作数数据。

11.根据权利要求8-10中的任一项所述的可重新配置逻辑器件，其中所述器件还被配置为通过刷新并且重新执行所述指令块来执行指令块的后续实例，并且其中所述就绪状态数据包括在所述刷新后不被清除的译码的就绪状态信息、以及在所述刷新后被清除的活动就绪状态数据。

12.根据权利要求8-11中的任一项所述的可重新配置逻辑器件，其中所述指令调度器被配置为重新使用针对指令的所存储的所述就绪状态数据的至少部分以用于执行所述指令块的后续实例，其中所述处理器被配置为不重新取指并且不重新译码所述指令以用于执行所述后续实例。

13.根据权利要求8-12中的任一项所述的可重新配置逻辑器件，其中所述指令调度器被配置为通过将所存储的所述就绪状态数据与通过执行另外的指令块而生成的一个或多个信号相比较，来确定所有指令的依赖性被满足。

14.一种形成具有可配置逻辑器件的基于块的处理器的方法，所述方法包括：

产生配置比特流，所述配置比特流包括用于利用所述可配置逻辑器件来实现针对所述基于块的处理器的电路的配置信息，针对所述基于块的处理器的所述电路包括：

乱序指令调度器，其被配置为基于存储在存储器中的就绪状态数据来发出指令，所述就绪状态数据通过唯一地识别事务指令块的每个相应指令的指令标识符而被索引。

15.根据权利要求14所述的方法，还包括将所述基于块的处理器的硬件描述语言规格映射到网表，所述网表包括要利用所述可配置逻辑器件来实现的硬件元件的描述，其中所述网表用于产生所述配置比特流。

乱序的基于块的处理器和指令调度器

背景技术

[0001] 由于摩尔定律所预测的持续的晶体管缩放(transistor scaling)，微处理器已经从晶体管数的持续增加、集成电路成本、制造资本、时钟频率、以及能量效率中受益，而相关联的处理器指令集架构(ISA)却变化很小。然而，从在过去40年里驱动半导体工业的光刻缩放实现的益处正在放缓或者甚至反转。精简指令集计算(RISC)架构已经成为处理器设计中的主导典范很多年。乱序的超标量实现尚未在面积或性能方面展现出持续改进。因此，存在对于为扩展性能改进而在处理器ISA中改进的足够机会。

发明内容

[0002] 公开了用于配置、操作并且编译用于基于块的处理器架构(BB-ISA)(包括显式数据图形执行(EDGE)架构)的代码的方法、装置以及计算机可读存储设备。用于例如改进处理器性能和/或降低能耗的解决方案的所描述的技术和工具可以分离地被实现，或者在彼此的各种组合中被实现。如下面将更充分地描述的，所描述的技术和工具可以被实现在以下各项中：数字信号处理器、微处理器、专用集成电路(ASIC)、软处理器(例如，使用可重新配置逻辑被实现在现场可编程门阵列(FPGA)中的微处理器核)、可编程逻辑、或者其他适合的逻辑电路。如对于本领域的普通技术人员而言将显而易见的，所公开的技术可以被实现在各种计算平台中，包括但不限于服务器、大型机、手机、智能电话、手持式设备、手持式计算机、个人数字助理(PDA)、触摸屏平板设备、平板计算机、可穿戴计算机、以及膝上型计算机。

[0003] 基于块的处理器架构的软处理器实现可以改进设计生产力。例如，以适合的描述语言(例如，C、SystemC、SystemVerilog或Verilog)编写的基于块的软件处理器的描述可以经历逻辑合成以生成被映射到FPGA的门极网表。针对FPGA生成用于对FPGA编程的比特流。软件到硬件中的昂贵的初始端口代替地变成针对软处理器的简单交叉编译，并且大多数设计轮是快速重新编译。应用瓶颈然后可以被卸载到被暴露为新指令、功能单元、自主加速器、存储器或互连件的定制硬件。

[0004] 所公开的技术的某些示例允许在不降低复杂性和开销的情况下配置高指令级并行(ILP)、乱序(OoO)超标量的软处理器。在一些示例中，为了面积和能量高效的高ILP执行而提供显式数据图形执行(EDGE)指令集架构。EDGE架构及其编译器共同解决了大部分的寄存器重命名、CAM和复杂性，从而实现用于比按序标量RISC仅仅多几百个的FPGA查找表(“LUT”)的乱序处理器。

[0005] 该所公开的技术介绍了EDGE ISA，并且探索了与按序RISC相比EDGE微架构如何。公开了用于构建FPGA中的小型快速数据流指令调度器的方法和装置。

附图说明

[0006] 图1图示了如可以根据所公开的技术的一些示例采用的包括多个处理器核的示例基于块的处理器。

[0007] 图2图示了如可以在所公开的技术的某些示例中使用的用于实现基于块的处理器

的示例微架构。

[0008] 图3是概述如可以在所公开的技术的一些示例中使用的示例FPGA微架构的块图。

[0009] 图4图示了如可以在所公开的技术的某些示例中使用的可重新配置逻辑块中的示例可重新配置逻辑。

[0010] 图5图示了如可以在所公开的技术的一些示例中使用的示例基于块的处理器头和指令。

[0011] 图6图示了如可以在所公开的技术的某些示例中使用的示例源代码部分和对应的指令块。

[0012] 图7图示了根据所公开的技术的可以用于基于块的处理器的某些示例的指令格式的示例。

[0013] 图8是图示了如可以在所公开的技术的某些示例中使用的、基于块的处理器中的处理器核的执行状态的进展的示例的流程图。

[0014] 图9是图示了如可以在所公开的技术的某些示例中使用的、包括基于块的处理器和存储器的示例配置的块图。

[0015] 图10是图示了如可以在所公开的技术的某些示例中使用的、发出具有编码的指令标识符的指令的示例方法的流程图。

[0016] 图11是概述如可以在所公开的技术的某些示例中执行的、当由就绪状态数据指示的依赖性可用时发出并且执行指令的示例方法的流程图。

[0017] 图12是概述如可以在所公开的技术的某些示例中执行的、产生用于实现基于块的处理器的配置比特流的示例方法的流程图。

[0018] 图13是图示了用于实现所公开的技术的某些实施例的适合的计算环境的块图。

具体实施方式

[0019] I.总体考虑

[0020] 在未旨在以任何方式进行限制的代表性实施例的上下文中阐述了本公开内容。

[0021] 如在本申请中所使用的，除非上下文清楚地指明相反，否则单数形式“一”、“一种”和“该”包括复数形式。此外，术语“包括”意味着“包含”。而且，术语“耦合的”涵盖机械的、电的、磁的、光学的、以及将多个项耦合或链接在一起的其他实用方式，并且不排除耦合项之间的中间元件的存在。另外，如在此所使用的，术语“和/或”意味着短语中的任何一项或多项的组合。

[0022] 在此所描述的系统、方法和装置不应当以任何方式被解释为限制性的。相反，本公开涉及单独以及以彼此的各种组合和子组合形式的各种所公开的实施例的所有新颖和非显而易见的特征和方面。所公开的系统、方法和装置既不限于任何特定方面或者特征或者其组合，所公开的内容和方法也不要求任何一个或多个特定优点存在或者问题被解决。此外，所公开的实施例的任何特征或者方面可以以彼此的各种组合和子组合被使用。

[0023] 虽然为了方便呈现而以特定顺序的排序描述所公开的方法中的一些方法的操作，但是应当理解，除非特定排序由下面阐述的特定语言所要求，否则说明书的这种方式涵盖重新布置。例如，顺序地描述的操作可以在一些情况下被重新布置或者并发地执行。此外，出于简单的缘故，附图可能未示出所公开的内容和方法能够结合其他内容和方法使用的各

种方式。此外，说明书有时使用类似“产生”、“生成”、“显示”、“接收”、“发射”、“验证”、“执行”和“发起”的术语来描述所公开的方法。这些术语是所执行的实际操作的高层描述。对应于这些术语的实际操作将取决于特定实现而变化，并且是由本领域的普通技术人员可容易地辨别的。

[0024] 参考本公开的装置或者方法在此所呈现的操作理论、科学原理或者其他理论描述已经出于更好的理解的目的而被提供，并且不旨在限制范围。所附的权利要求中的装置和方法不限于以由这样的操作理论所描述的方式实现的那些装置和方法。

[0025] 所公开的方法中的任一方法可以被实现为被存储在一个或多个计算机可读介质(例如，计算机可读介质(诸如一个或多个光学介质光盘、易失性存储器部件(诸如DRAM或SRAM))或非易失性存储器部件(诸如硬盘驱动器))上并且在计算机(例如，任何市售的计算机，包括智能电话或者包括计算硬件的其他移动设备)上被执行的计算机可执行指令。用于实现所公开的技术的计算机可执行指令中的任一指令以及在所公开的实施例的实现期间创建和使用的任何数据，可以被存储在一个或多个计算机可读介质(例如，计算机可读存储介质)上。计算机可执行指令可以是例如专用软件应用、或者经由网络浏览器或其他软件应用(诸如远程计算应用)而访问或下载的软件应用的一部分。这样的软件可以例如在单个本地计算机(例如，具有在任何适合的市售的计算机上执行的通用和/或基于块的处理器)上被执行，或者在使用一个或多个网络计算机的网络环境(例如，经由因特网、广域网、局域网、客户端服务器网络(诸如云计算网络)、或者其他这样的网络)中被执行。

[0026] 为了清晰起见，仅描述了基于软件的实现的某些选择的方面。省略了在本领域中众所周知的其他细节。例如，应当理解，所公开的技术不限于任何特定计算机语言或者程序。例如，所公开的技术可以通过以C、C++、JAVA或者任何其他适合的编程语言编写的软件来实现。同样地，所公开的技术不限于任何特定计算机或者硬件类型。适合的计算机和硬件的某些细节是众所周知的并且不需要在本公开中被详细阐述。

[0027] 此外，基于软件的实施例(包括例如用于使计算机执行所公开的方法中的任一方法的计算机可执行指令)中的任一实施例可以通过适合的通信手段被上传、被下载或者被远程访问。这样的适合的通信手段包括例如因特网、万维网、内联网、软件应用、电缆(包括光纤电缆)、磁通信、电磁通信(包括RF、微波和红外通信)、电子通信、或者其他这样的通信手段。

[0028] II.对所公开的技术的介绍

[0029] 超标量乱序微架构采用大量的电路资源来重命名寄存器、以数据流顺序调度指令、在误推测之后清理、并且针对精确异常而按序引退结果。这包括昂贵的电路，诸如深度多端口的寄存器文件、用于数据流指令调度唤醒的多端口的内容可访问存储器(CAM)、以及多宽度(many-wide)总线复用器和旁路网络，所有的这些都是资源密集的。例如，多读取、多写入RAM的基于FPGA的实现通常要求复制、多循环操作、时钟加倍、组交错、实况值表和其他昂贵技术的混合。

[0030] 所公开的技术可以通过应用包括高指令集并行性(ILP)、乱序(out-of-order，OoO)、超标量执行的技术来实现性能增强，同时避免处理器硬件和相关联的软件二者中的大量的复杂性和开销。在所公开的技术的一些示例中，基于块的处理器使用针对区域和能量高效的高ILP执行所设计的EDGE ISA。在一些示例中，EDGE架构和相关联的编译器的使用

解决了大部分的寄存器重命名、CAM和复杂性。

[0031] 在所公开的技术的某些示例中，EDGE ISA可以消除对于一个或多个复杂架构特征(包括寄存器重命名、数据流分析、误推测恢复以及按序引退)的需要，同时支持主流编程语言(诸如C和C++)。在所公开的技术的某些示例中，基于块的处理器执行多个(两个或更多个)指令作为原子块。基于块的指令可以被用于以更显式的方式表达程序数据流和/或指令流的语义，这允许改进的编译器和处理器性能。在所公开的技术的某些示例中，显式数据图形执行指令集架构(EDGEISA)包括关于可以用于改进对不正确的控制流指令的检测的程序控制流的信息，从而增加性能、节省存储器资源和/或以及节省能量。

[0032] 在所公开的技术的一些示例中，在指令块内组织的指令被原子地取指、执行和提交。块内部的指令以数据流顺序执行，其减少或消除使用寄存器的重命名并且提供功率高效的OoO执行。编译器可以被用于通过ISA显式地编码数据依赖性，从而减少或者消除负担的处理器核控制逻辑在运行时重新发现依赖性。使用所断言的执行，块内分支可以被转换为数据流指令，并且除了存储器依赖性之外的依赖性可以限于直接数据依赖性。所公开的目标形式编码技术允许块内的指令经由操作数缓冲器直接地传递其操作数，从而减少对耗电的(power-hungry)多端口物理寄存器文件的访问。

[0033] 在指令块之间，指令可以使用存储器和寄存器来通信。因此，通过利用混合数据流执行模型，EDGE架构可以仍然支持命令式编程语言和顺序的存储器语义，但是期望地还享有具有近按序的功率效率和复杂性的乱序执行的益处。

[0034] III.示例基于块的处理器

[0035] 图1是如可以被实现在所公开的技术的一些示例中的基于块的处理器100的块图10。处理器100被配置为根据指令集架构(ISA)来执行原子指令块，ISA描述了处理器操作的若干方面，包括寄存器模型、由基于块的指令执行的若干定义操作、存储器模型、中断、以及其他架构特征。基于块的处理器包括多个一个或多个处理器核110，其包括处理器核111。基于块的处理器可以被实现为定制或专用集成电路(例如，包括片上系统(SoC)集成电路)、被实现为现场可编程门阵列(FPGA)或其他可重新配置逻辑、或者被实现为由物理通用处理器托管的虚拟机。

[0036] 如在图1中所示，处理器核经由核互连120而彼此连接。核互连120携带数据并且控制核110中的单独的核、存储器接口140以及输入/输出(I/O)接口150之间的信号。核互连120可以使用电的、光学的、磁的或者其他适合的通信技术来发射和接收信号，并且可以取决于特定期望的配置而提供根据若干不同的拓扑来布置的通信连接。例如，核互连120可以具有交叉开关、总线、点对点总线、或者其他适合的拓扑。在一些示例中，核110中的任一核可以被连接到其他核中的任一核，而在其他示例中，一些核仅被连接到其他核的子集。例如，每个核可以仅被连接到最近的4、8或20个邻近核。核互连120可以用于将输入/输出数据发射至核以及从核发射输入/输出数据，以及将控制信号和其他信息信号发射至核以及从核发射控制信号和其他信息信号。例如，核110中的每个核110可以接收并且发射指示当前正由相应核中的每个核所执行的指令的执行状态的信号量(semaphore)。在一些示例中，核互连120被实现为将核110和存储器系统连接的接线，而在其他示例中，核互连可以包括用于多路复用(一条或多条)互连线上的数据信号的电路、开关和/或路由部件(包括活跃的信号驱动器和中继器)、或者其他适合的电路。在所公开的技术的一些示例中，在处理器100内

发射的信号和/或向/从处理器100发射的信号不限于全摆幅电数字信号，而是处理器可以被配置为包括差分信号、脉冲信号、或者用于发射数据和控制信号的其他适合的信号。

[0037] 在图1的示例中，处理器的存储器接口140包括被用于连接到存储器145(例如，被定位在除了处理器100之外的另一集成电路上的存储器(例如，存储器可以是静态RAM(SRAM)或动态RAM(DRAM)、或者嵌入在与处理器相同的集成电路上的存储器(例如，嵌入式SRAM或DRAM(eDRAM)))的接口逻辑。存储器接口140和/或主存储器可以包括高速缓存(例如，n路或关联高速缓存)以改进存储器存取性能。在一些示例中，高速缓存使用静态RAM(SRAM)被实现，并且主存储器145使用动态RAM(DRAM)被实现。在一些示例中，存储器接口140被包括在与处理器100的其他部件相同的集成电路上。在一些示例中，存储器接口140包括允许在不使用(一个或多个)寄存器文件和/或处理器100的情况下传送存储器中的数据块的直接存储器访问(DMA)控制器。在一些示例中，存储器接口140管理虚拟存储器的分配，从而扩展可用的主存储器145。在一些示例中，用于旁路高速缓存结构、或者用于当执行存储器同步操作(例如，处理争用问题或者在多个不同线程、进程或处理器之间共享的存储器)时确保高速缓存相干性的支持，由存储器接口140和/或相应的高速缓存结构提供。

[0038] I/O接口150包括用于将输入信号和输出信号接收并且发送到其他部件155的电路，其他部件155诸如硬件中断、系统控制信号、外围接口、协处理器控制和/或数据信号(例如，用于图形处理单元、浮点协处理器、物理处理单元、数字信号处理器或者其他协处理部件的信号)、时钟信号、信号量、或者其他适合的I/O信号。I/O信号可以是同步的或者异步的。在一些示例中，I/O接口的全部或部分结合存储器接口140使用存储器映射的I/O技术被实现。在一些示例中，I/O信号实现不限于全摆幅电数字信号，但是I/O接口150可以被配置为提供差分信号、脉冲信号、或者用于发射数据和控制信号的其他适合的信号。

[0039] 基于块的处理器100还可以包括控制单元160。控制单元160监督处理器100的操作。能够由控制单元160执行的操作可以包括对核的分配和去分配以用于执行指令处理；对在核、寄存器文件、存储器接口140和/或I/O接口150中的任何项之间的输入数据和输出数据的控制；对执行流的修改；以及验证控制流中的分支指令、指令头和其他改变的(一个或多个)目标位置。控制单元160可以根据表示针对指令块的退出点和控制流概率的控制流和元数据信息来产生并且控制处理器。

[0040] 控制单元160还可以处理硬件中断，并且控制特殊系统寄存器(例如，被存储在一个或多个寄存器文件中的程序计数器)的读取和写入。在所公开的技术的一些示例中，控制单元160至少部分地使用处理器核110中的一个或多个核被实现，而在其他示例中，控制单元160使用非基于块的处理器核(例如，耦合到存储器的通用RISC处理核、在FPGA中提供的硬宏处理器块、或者通用软处理器)被实现。在一些示例中，控制单元160至少部分地使用以下各项中的一项或多项被实现：硬连线有限状态机、可编程微代码、可编程门阵列、或者其他适合的控制电路。在备选示例中，可以由核110中的一个或多个核来执行控制单元功能。

[0041] 控制单元160包括用于控制处理器核110的指令流水线的多个调度器165-168。在其他示例中，调度器可以被布置为使得它们关于每个单独处理器核被包含。如在此所使用的，调度器块分配涉及引导指令块的操作，包括发起指令块映射、取指、译码、执行、提交、中止、空闲以及刷新指令块。另外，指令调度涉及调度指令块内的指令的发出和执行。例如，基于指示针对存储器存取指令的相对顺序的指令依赖性和数据，控制单元160可以确定指令

块中的哪个(哪些)指令准备好发出,并且启动对指令的发出和执行。处理器核110在指令块映射期间被指派到指令块。所叙述的指令操作的阶段出于说明性目的,并且在所公开的技术的一些示例中,某些操作可以被组合、被省略、被分开为多个操作、或者被添加附加操作。调度器165-168中的每个调度器调度指令的流,包括:对用于执行指令处理的核的分配和解除分配;对在核、寄存器文件、存储器接口140和/或I/O接口150中的任何项之间的输入数据和输出数据的控制。

[0042] 基于块的处理器100还包括时钟发生器170,其将一个或多个时钟信号分配到处理器内的各种部件(例如,核110、互连120、存储器接口140和I/O接口150)。在所公开的技术的一些示例中,所有部件共享共同的时钟,而在其他示例中,不同的部件使用不同的时钟(例如,具有不同的时钟频率的时钟信号)。在一些示例中,时钟的一部分被选通,以在处理器部件中的一些部件未被使用时允许功率节省。在一些示例中,时钟信号使用锁相环(PLL)被生成以生成具有固定的恒定频率和占空比的信号。接收时钟信号的电路可以在单个沿(例如,上升沿)上被触发,而在其他示例中,接收电路中的至少一些电路由上升和下降时钟沿而被触发。在一些示例中,时钟信号可以光学地或无线地被传输。

[0043] IV.示例基于块的处理器微架构

[0044] 图2是进一步详述如可以在所公开的技术的某些示例中使用的、用于实现基于块的处理器100(并且特别地,基于块的处理器核之一的实例)的示例微架构200的块图。为了便于解释,示例性微架构具有五个流水线阶段,其包括:指令取指(IF)、译码(DC)、包括操作数取指的发出(IS)、执行(EX)以及存储器/数据访问(LS)。然而,本领域的普通技术人员将容易地理解到,对所图示的微架构的修改(诸如添加/移除阶段、添加/移除执行操作的单元、以及其他实现细节)可以被修改为适合用于基于块的处理器的特定应用。

[0045] 如图2所示,处理器核包括耦合到指令译码器220的指令高速缓存210。指令高速缓存210被配置为从存储器接收基于块的处理器指令。在一些FPGA实现中,指令高速缓存可以由双读端口、双写端口、18或36Kb(千比特)、32位宽块RAM来实现。在一些示例中,物理块RAM被配置为用作两个或更多个更小块的RAM。

[0046] 处理器核还包括指令窗口230,其包括指令调度器235、译码的指令存储236和多个操作数缓冲器239。在FPGA实现中,这些指令窗口部件230中的每个可以被实现,包括使用LUT RAM(例如,具有被配置为查找表的SRAM)或BRAM(块RAM)。指令调度器235可以将针对指令的指令标识符(指令ID或IID)作为控制信号发送到译码的指令存储236和操作数缓冲器239。如下面进一步讨论的,指令块中的每个指令具有唯一地识别指令块内的指令的相关联的指令标识符。在一些示例中,用于发送执行指令的结果的指令目标被编码在指令中。以这种方式,代替监视寄存器依赖性,指令之间的依赖性可以使用指令标识符来跟踪。在一些示例中,处理器核可以包括两个或更多个指令窗口。在一些示例中,处理器核可以包括具有多个块上下文的一个指令窗口。

[0047] 如下面将进一步讨论的,微架构200包括存储针对基于块的处理器架构中定义的寄存器的数据的寄存器文件290,并且可以具有一个或多个读端口和一个或多个写端口。因为指令块在事务基础上执行,所以由指令块的实例做出的对寄存器值的改变对相同实例不可见;寄存器写将在完成了对指令块的执行后被提交。

[0048] 译码的指令存储236存储用于控制处理器流水线中的硬件部件的操作的译码的信

号。例如,32位指令可以被译码成128位的译码的指令数据。译码的指令数据由译码器220在指令被取指之后被生成。操作数缓冲器239存储操作数(例如,从寄存器文件接收到的寄存器值,从存储器接收到的数据,编码于指令内的立即操作数,由早前发出的指令计算的操作数,或者其他操作数的值)直到它们相应的译码的指令准备好执行。用于流水线的执行阶段的指令操作数和断言相应地从操作数缓冲器239、而不是(至少直接地)从寄存器文件290读取。指令窗口230可以包括用于指向指令的断言的缓冲器,包括用于组合通过多个指令发送到指令的断言的线或逻辑(wired-OR logic)。

[0049] 在一些示例中,除了寄存器读操作的所有指令操作数从操作数缓冲器239而不是从寄存器文件被读取。在一些示例中,这些值被维持直到指令发出并且操作数被传递到执行流水线。在一些FPGA示例中,译码的指令存储236和操作数缓冲器239利用多个LUT RAM被实现。

[0050] 指令调度器235维持每个译码的指令的依赖性(例如,指令的断言和数据操作数)的就绪状态的记录。当全部指令的依赖性(如果有的话)都被满足时,指令唤醒并且准备好发出。在一些示例中,最低编号的就绪指令ID在每个流水线时钟周期被选择并且其译码的指令数据和输入操作数被读取。在图示的示例中,除了数据复用和功能单元控制信号,译码的指令数据可以编码至多两个就绪事件。指令调度器235接受这些事件和/或从其他源(被选择用于输入到分别具有多路复用器237和238的输入T0和T1上的调度器)接收事件,并且更新窗口中的其他指令的就绪状态。因此,数据流执行继续,以指令块的就绪零输入指令开始,接着是这些指令瞄准的指令,等等。一些指令准备好在它们不具有依赖性时立即发出(例如,移动立即指令)。取决于ISA、控制结构、以及其他因素,在一些示例中,译码的指令存储236大约为100位宽并且包括关于指令依赖性的信息,包括指示哪个(哪些)目标指令的活动就绪状态由于发出指令而将被设置的数据。如本文中所使用的,就绪状态是指针对给定指令指示其操作数(如果有的话)是否就绪以及其操作数(如果有的话)中的哪些就绪、以及指令本身现在是否准备好发出的处理器状态。在一些示例中,就绪状态包括译码的就绪状态和活动就绪状态。译码的就绪状态数据通过译码(一个或多个)指令来初始化。活动就绪状态表示在指令块的当前实例的执行期间、到目前为止已经评估的指令的输入操作数的集合。相应指令的活动就绪状态通过执行瞄准例如相应指令的左操作数、右操作数和/或断言操作数的(一个或多个)指令来设置。

[0051] 指令窗口230和指令调度器235的属性(诸如面积、时钟周期和功能)可以对EDGE核的实现的性能和EDGE多处理器的吞吐量具有重大影响。在一些示例中,微架构的前端(IF、DC)部分可以与微架构的后端部分(IS、EX、LS)解耦运行。在一些FPGA实现中,指令窗口230被配置为每个时钟将两个指令取指和译码到指令窗口中。

[0052] 指令调度器235具有各种各样的功能和要求。其可以是高度并发的。在每个时钟周期,指令译码器220将针对一个或多个指令的译码的就绪状态和译码的指令数据写入到指令窗口230中。在每个时钟周期,指令调度器235选择接下来的(一个或多个)指令来发出,并且作为响应,后端发送就绪事件,例如,瞄准特定指令的输入槽(例如,断言槽、右操作数(OP0)或左操作数(OP1))的目标就绪事件、或者瞄准在广播ID上等待的所有指令的广播就绪事件。这些事件使得每指令的活动就绪状态被设置为,其与译码的就绪状态一起可以用于信号通知对应指令准备好发出。指令调度器235有时接受针对尚未被译码的目标指令的

事件，并且调度器还可以抑制发出的就绪指令的重新发出。

[0053] 指令窗口230中的控制电路(例如，使用译码的指令存储236生成的信号)用于生成控制信号，以调节核操作(包括例如对数据路径和多路复用器选择信号的控制)并且调度核内的指令的流。这可以包括生成并且使用存储器存取指令编码，对用于执行指令处理的核的分配和解除分配，在核110、寄存器文件、存储器接口140和/或I/O接口150中的任何一项之间的输入数据和输出数据的控制。

[0054] 在一些示例中，指令调度器235被实现为耦合到其他指令窗口逻辑的有限状态机。在一些示例中，指令调度器被映射到FPGA中的RAM的一个或多个库，并且可以利用块RAM、LUT RAM或其他可重新配置RAM来实现。如相关领域的普通技术人员将显而易见的，以集成电路、可编程逻辑或其他适合的逻辑实现的其他电路结构可以用于实现针对指令调度器235的硬件。在所公开的技术的一些示例中，前端流水线阶段IF和DC可以与后端流水线阶段(IS、EX、LS)解耦运行。

[0055] 在图2的示例中，操作数缓冲器239经由一个或多个开关(例如，多路复用器241和242)将数据操作数(为了方便，其可以被指定为左操作数(LOP)和右操作数(ROP))发送到执行状态流水线寄存器245的集合。这些操作数还可以分别被称为OP1和OP0。第一路由器240用于将数据从操作数缓冲器239发送到功能单元250中的一个或多个，功能单元250可以包括但不限于整数ALU(算术逻辑单元)(例如，整数ALU 255)、浮点单元(例如，浮点ALU 256)、移位/旋转逻辑(例如，桶式移位器257)、或者其他适合的执行单元，其可以包括图形功能、物理功能以及其他数学运算。在一些示例中，可编程执行单元258可以被重新配置为实现多个不同的任意功能(例如，先验或在运行时)。

[0056] 来自功能单元250的数据可以随后通过第二路由器(未示出)被路由到加载/存储流水线寄存器260的集合、加载/存储队列270(例如，用于执行存储器加载和存储器存储操作)，或者被反馈回到执行流水线寄存器、由此旁路操作数缓冲器239。加载/存储队列270被耦合到对用于存储器操作的数据进行高速缓存的数据高速缓存275。数据高速缓存275和加载/存储流水线寄存器260的输出可以被发送到第三路由器280，其转而根据正在流水线阶段中执行的指令而将数据发送到寄存器文件290、操作数缓冲器239和/或执行流水线寄存器245。

[0057] 当指令块的执行完成时，指令块被指定为“已提交”，并且来自控制输出的信号可以转而由基于块的处理器100内的其他核和/或由控制单元160使用，以启动对其他指令块的调度、取指和执行。如相关领域的普通技术人员将容易理解的，单独的核内的部件不限于图2中示出的那些，而是可以根据具体应用的要求而变化。例如，核可以具有更少或更多的指令窗口，单个指令译码器可以由两个或更多个指令窗口共享，并且使用的功能单元的数量和类型可以取决于针对基于块的处理器的具体目标应用而变化。在选择和分配具有指令核的资源中应用的其他考虑包括性能要求、能量使用要求、集成电路管芯、处理工艺、和/或成本。

[0058] 对相关领域的普通技术人员将显而易见的是，可以通过设计和分配处理器核110的指令窗口和控制单元内的资源，而在处理器性能上做出权衡。面积、时钟周期、功能和限制基本上确定单独的核110的实现的性能和基于块的处理器100的吞吐量。

[0059] 对受执行的指令影响的处理器(诸如寄存器文件290和存储器)的可见架构状态的

更新可以被本地缓冲在核内，直到指令被提交。控制电路可以确定何时指令准备好被提交，对提交逻辑排序，并且发出提交信号。例如，针对指令块的提交阶段可以在所有寄存器写入被缓冲、所有对存储器的写入(包括无条件和有条件的存储)被缓冲以及分支目标被计算时开始。指令块可以在对可见架构状态的更新完成时被提交。例如，指令块可以在寄存器写被写入作为寄存器文件、存储被发送到加载/存储单元或存储器控制器以及提交信号被生成时而被提交。控制电路还至少部分地控制功能单元到指令窗口的分配。

[0060] 因为指令块作为原子事务单元被提交(或中止)，所以应当指出的是，某些操作的结果对指令块内的指令不可用。这与提供在单独的、逐个指令的基础上可见的结果的RISC和CISC架构形成对比。因此，公开了用于在基于块的处理器环境中支持存储器同步和其他存储器操作的附加技术。

[0061] 在一些示例中，基于块的执行可以是非断言的、或者被断言为真或假。断言的指令直到其被另一指令的断言结果瞄准、并且该结果与断言条件匹配时才变成就绪。如果指令的断言不匹配，则指令决不发出。

[0062] 在一些示例中，在分支到新指令块后，(存储于指令调度器235中的)所有指令窗口就绪状态被闪速清除(块重置)。然而，当块分支回到其本身(块刷新)时，仅仅活动就绪状态被清除；译码的就绪状态被保留，以使得不必重新取指和译码块的指令。因此，代替执行块重置，刷新可以用于节省循环中的时间和能量。

[0063] 由于一些软件关键路径包括相关指令的单个链(例如，指令A瞄准指令B，指令B转而瞄准指令C)，所以通常期望的是，数据流调度器不添加用于连续的背靠背指令唤醒的流水线气泡(bubble)。在这样的情况下，IS-阶段的就绪-发出-目标-就绪流水线的再次发生应当在一个周期中完成，假设这不会严重影响时钟频率。

[0064] 诸如ADD的指令具有一个周期的延时。在EX阶段结果转发的情况下，调度器可以在IS阶段中、甚至在指令完成之前唤醒它们目标的指令。其他指令结果可以等待ALU比较、采用多个周期、或者具有未知的延时。这些指令等待直到稍后唤醒它们的目标。

[0065] 最终，调度器设计可以在一定范围的EDGE ISA上可扩展。在一些示例中，每个流水线周期可以接受从一个至四个的译码的指令和从两个至四个的目标就绪事件，并且在每个周期发出一个至两个指令。

[0066] 多种不同的技术可以用于实现指令调度器235。例如，调度器235可以被实现为并行调度器，其中指令的就绪状态被显式地表示在FPGA D型触发器(FF)中，并且其中每一个指令的就绪状态在每个周期都被重新评估。在其他示例中，指令调度器235可以被实现为将就绪状态保持在LUT RAM中的更紧凑的增量调度器，并且指令调度器235在每个周期更新大约两个至四个目标指令的就绪状态。

[0067] 寄存器文件290可以包括用于将数据存储在寄存器文件中的两个或更多个写端口，以及具有用于从寄存器文件内的单独的寄存器读取数据的多个读端口。在一些示例中，单个指令窗口(例如，指令窗口230)可以一次访问寄存器文件的仅一个端口，而在其他示例中，指令窗口230可以访问一个读端口和一个写端口、或者可以同时访问两个或更多个读端口和/或写端口。在一些示例中，微架构被配置为使得并非寄存器290的所有读端口都可以使用旁路机构。对于图2中所示的示例微架构200，寄存器文件可以将旁路路径上的寄存器数据发送到针对操作数OP0、而不是操作数OP1的多路复用器242中的一个。因此，对于一个

周期内的多次寄存器读取，仅仅一个操作数可以使用旁路，而另一寄存器读取结果被发送到操作数缓冲器239，其将额外时钟周期插入指令流水线中。

[0068] 在一些示例中，寄存器文件290可以包括64个寄存器，寄存器中的每个寄存器保持32位的数据的字。(为便于解释，除非另外指定，否则本申请将把32位的数据称为字。根据所公开的技术的适合的处理器可以利用8位、16位、64位、128位或其他数量的位的字进行操作)在一些示例中，寄存器文件290内的寄存器中的一些寄存器可以被分配至特殊目的。例如，寄存器中的一些寄存器可以被专用作系统寄存器，其示例包括以下的寄存器：存储恒定值(例如，所有零字)、(一个或多个)程序计数器(PC)(其指示正被执行的程序线程的当前地址)、物理核数目、逻辑核数目、核分配拓扑、核控制标志、执行标志、处理器拓扑或者其他适合的专用目的。在一些示例中，寄存器文件290被实现为触发器阵列，而在其他示例中，寄存器文件可以使用锁存器、SRAM或者其他形式的存储器存储装置而被实现。针对给定处理器的ISA规格指定寄存器文件290内的寄存器如何被定义并且被使用。

[0069] V.示例现场可编码门阵列架构

[0070] 图3是描绘被配置为实现所公开的技术的某些示例的示例现场可编程门阵列(FPGA)架构的块图300。例如，以上关于图1讨论的基于块的处理器100(包括使用图2中描绘的微架构200的那些示例)可以被映射到图3的FPGA架构。

[0071] FPGA包括以阵列布置的可重新配置逻辑块的阵列。例如，FPGA包括：第一行逻辑块，其包括逻辑块310、311和319；以及第二行逻辑块，其包括逻辑块320、321和329。逻辑块中的每个包括可以被重新配置为实现任意逻辑功能的逻辑，以及还可以包括诸如锁存器、触发器和存储器的顺序逻辑元件的逻辑。逻辑块使用包括多个互连开关(其也可以是可编程的)的路由结构而被彼此互连。例如，存在被定位在第一行可重新配置逻辑块与第二行可重新配置逻辑块之间的第一行开关块330、331、332等等。开关可以被配置以便改变在可重新配置逻辑块之间运送信号的有线连接。例如，指令调度器、功能单元、流水线缓冲器以及操作数缓冲器可以使用图3的开关块而被映射到逻辑块。

[0072] FPGA还包括多个更复杂的部件。例如，逻辑块包括多个块RAM，例如块RAM 340和块RAM 349。块RAM通常包含更大数量的存储器位，例如，通过将地址应用到存储器并且从一个或多个读端口读取来访问的几千个存储器位。在一些示例中，块RAM可以包括两个或更多个写端口和两个或更多个读端口。在其他示例中，块RAM可以仅仅具有单个读端口和/或单个写端口。在块RAM通常通过应用地址并且读取对应数据来访问的同时，在一些示例中，块RAM可以被配置有附加电路，其允许实现包括移位寄存器和先进先出(FIFO)缓冲器的更复杂的功能。

[0073] 图示的FPGA还包括多个硬宏块，其包括硬宏块350和硬宏块359。这些宏块可以包括更复杂的功能，诸如处理器功能、数字信号处理功能、加速器或认为是期望的其他功能。FPGA还由可以被耦合到逻辑块、块ram和/或硬宏块的I/O环360环绕，以便向远离FPGA的部件接收和发送信号。在一些示例中，I/O信号是全轨电压信号，而其他示例使用差分信号。在一些示例中，I/O端口可以被多路复用(例如，时分多路复用)以便支持比在FPGA上可用的管脚的数量更多的信号的输入和输出。

[0074] 尽管FPGA的许多示例通常是通过使用电可擦存储器而可重新配置任意次数的，但是在其他示例中，可以使用一次性可编程逻辑元件。例如，逻辑块和开关可以利用熔丝、反

熔丝的使用而被编程，或者利用ROM掩码而被编程，以不容易可逆的方式一次性编程逻辑功能。

[0075] 在可重新配置情况中，FPGA通常具有根据被称为比特流或配置比特流的文件来接收数据的配置端口。比特流被读取到设备中并且用于编程和配置逻辑块、开关、块ram和/或硬宏。当期望新设计时，配置可以被擦除并且新的设计被配置到设备中。在一些示例中，FPGA可以被部分地重新配置以便节省编程时间。例如，逻辑块、开关或块ram的子集可以在现场被动态地重新配置而无需对整个设备重新编程。

[0076] 针对被映射到可重新配置逻辑上的基于块的处理器的实现的一个挑战是确定可以使用定制或现成设备的可用块高效实现的微架构结构。然而，使用所公开的技术，可以实现更高的性能和/或更高效的结构。另外，应当容易理解，尽管FPGA的一些示例是独立集成电路，但是在其他示例中，FPGA可以被不同地封装，例如在多芯片模块(MCM)中、或者在与定制或基础片上系统(SoC)相同的电路管芯上被封装。

[0077] 图4是图示可以被配置为形成示例FPGA集成电路的逻辑结构的部分的四个可重新配置逻辑块410、411、412和413的块图400。示出的可重新配置逻辑块内部的部件是相同的或均质的，但是应当容易理解，在其他示例中，多于一种类型的可重新配置逻辑块可以存在于单个FPGA上。

[0078] 第一可重新配置逻辑块410包括被耦合到进位逻辑430、多个多路复用器440和445以及存储元件(这里为D触发器)450的六输入查找表(LUT)420。LUT 420可以使用小的存储器(例如，如所示的具有六个地址位和两个输出位的存储器)来实现。因此，任何六输入布尔函数可以通过使用单个LUT来实现。在一些示例中，LUT的输出可以被组合，或者可重新配置逻辑块可以具有能够被连接在一起以便执行更复杂的逻辑功能的多个LUT。在一些示例中，除了LUT，可以提供常见逻辑功能。例如，进位逻辑430可以被配置为执行针对加法器的进位传播逻辑。多路复用器用于选择来自其他部件的各种输出。例如，多路复用器440可以用于选择LUT 420或进位逻辑430的输出，而多路复用器445可以用于选择LUT 420或多路复用器440的另一输出。在一些示例中，多路复用器用于选择状态元件(例如触发器450)的顺序输出或者查找表的组合输出。对本领域的普通技术人员而言应当容易理解，不同的逻辑功能、LUT大小和顺序要素可以被采用于可重新配置逻辑元件中。因此，用于将基于块的处理器映射到这样的可重新配置逻辑的技术可以取决于特定目标FPGA架构而变化。可重新配置逻辑块内部的逻辑的配置可以使用FPGA的配置端口来编程。在一些示例中，LUT不被一次编程，而是可以被配置为用作存储在基于块的处理器中使用的特定数据的小的存储器。

[0079] 在所公开的技术的一些示例中，逻辑合成工具(逻辑编译器)用于将针对块处理器的规格转换成可以被应用到FPGA的配置端口，以将逻辑配置为实现基于块的处理器的配置比特流。在一些示例中，设计者可以使用RPM(关系放置宏)技术来改进面积和互连延迟，并且实现用于在模块组成和大规模复制下的轻松路由和定时关闭的可重复布局。例如，通过包括实例化模块并且将它们拼接到调度器中的结构RTL，针对指令调度器的逻辑可以被锁定到单个LUT的集合，从而允许对FPGA内的逻辑的紧凑聚类和放置。

[0080] VI.示例指令块流

[0081] 现在转到图5的图500，图示了基于块的指令流的一部分510，包括若干可变长度指令块511-514。指令流可以用于实现用户应用、系统服务或者任何其他适合的用途。指令流

可以被存储在存储器中，从存储器中的另一过程接收，通过网络连接接收，或者以任何其他适合的方式被存储或接收。在图5中所示的示例中，每个指令块从指令头开始，其跟随有不同的数目的指令。例如，指令块511包括头520和二十条指令521。所图示的特定指令头520包括部分地控制指令块内的指令的执行的若干数据字段，并且还允许改进的性能增强技术，包括例如分支预测、推测执行、惰性评估和/或其他技术。指令头520还包括指令块大小的指示。指令块大小可以处于比例如被包含在指令块内的4指令数据块(chunk)的数目更大的指令的数据块中。换句话说，块的大小被移位4位以便压缩被分配到指定指令块大小的头空间。因此，0的大小值指示最小大小的指令块，其是跟随有四条指令的块头。在一些示例中，指令块大小被表达为字节数、字数、n字数据块数、地址、地址偏移或者使用用于描述指令块的大小的其他适合的表达。在一些示例中，指令块大小由指令块头和/或脚中的终止位模式来指示。

[0082] 指令块头520还可以包括一个或多个执行标志，其指示用于执行指令块的一种或多种操作模式。例如，操作模式可以包括核融合操作、向量模式操作、存储器依赖性预测和/或按序或确定性指令执行。另外，执行标志可以包括抑制指令块的推测执行的块同步标志。

[0083] 在所公开的技术的一些示例中，指令头520包括指示编码数据是指令头的一个或多个标识位。例如，在一些基于块的处理器ISA中，最低有效位空间中的单个ID位总是被设定为二进制值1，以指示有效指令块的开始。在其他示例中，不同的位编码可以用于(一个或多个)标识位。在一些示例中，指令头520包括指示相关联的指令块被编码所针对的ISA的特定版本的信息。

[0084] 指令块头还可以包括用于在例如分支预测、控制流确定和/或分支处理中使用的若干块退出类型。退出类型可以指示分支指令的类型是什么，例如：顺序分支指令，其指向存储器中的下一相连的指令块；偏移指令，其是相对于偏移而计算的存储器地址处的另一指令块的分支；子例程调用、或者子例程返回。通过编码指令头中的分支退出类型，分支预测器可以至少部分地在相同指令块内的分支指令已经被取指和/或被译码之前开始操作。

[0085] 图示的指令块头520还包括存储掩码，其指示编码于块指令中的加载存储队列标识符中的哪些被指派到存储操作。指令块头还可以包括写入掩码，其标识相关联的指令块将写入的(一个或多个)全局寄存器。在一些示例中，存储掩码由例如指令译码器(例如，译码器220)存储在存储向量寄存器中。在其他示例中，指令块头520不包括存储掩码，而是在指令块被译码时由指令译码器通过分析指令依赖性来动态地生成存储掩码。例如，译码器可以生成针对指令块的加载存储标识符以确定存储掩码并且将存储掩码数据存储在存储向量寄存器中。类似地，在其他示例中，写入掩码未被编码于指令块头中，而是被动态地生成(例如，由指令译码器通过分析指令块中的指令所涉及的寄存器)并且被存储在写入掩码寄存器中。写入掩码可以用于确定指令的执行何时已经完成并且因此启动指令块的提交。相关联的寄存器文件必须在指令块可以完成之前接收对每个条目的写入。在一些示例中，基于块的处理器架构可以不仅包括标量指令，而且还包括单指令多数据(SIMD)指令，其允许具有单个指令内的更大数目的数据操作数的操作。

[0086] 可以用于指令521的合适的基于块的指令的示例可以包括用于执行整数和浮点运算、逻辑操作、类型转换、寄存器读和写、存储器加载和存储、分支和跳的执行的指令，以及其他适合的处理器指令。在一些示例中，指令包括用于将处理器配置为根据通过例如推测

的一个或多个操作来进行操作的指令。因为指令的依赖性被编码于指令块中(例如,指令块头、瞄准指令的其他指令中、和/或指令本身中),所以指令可以在指令的依赖性被满足时发出并且不按程序顺序执行。

[0087] VII.示例块指令目标编码

[0088] 图6是描绘C语言源代码的两个部分610和615及其相应的指令块620和625的示例的图600,其图示了基于块的指令可以如何显式地编码其目标。在该示例中,前两个READ(读取)指令630和631分别瞄准ADD(加法)指令632的右(T[2R])和左(T[2L])操作数(2R指示瞄准指令号2的右操作数;2L指示指令号2的左操作数)。在所图示的ISA中,读指令是从全局寄存器文件(例如,寄存器文件290)读取的唯一指令;然而,任何指令可以瞄准全局寄存器文件。当ADD指令632接收到这两个寄存器读取的结果时,其将变为就绪并且执行。要指出,本公开内容有时将右操作数称为OP0并且将左操作数称为OP1。

[0089] 当TLEI(测试小于等于立即(test-less-than-equal-immediate))指令633从ADD接收其单个输入操作数时,其将变为准备好发出并且执行。测试然后产生在信道一(B[1P])上广播到在用于断言的广播信道上监听的所有指令的断言操作数,其在该示例中是两个断言的分支指令(BRO_T 634和BRO_F 635)。接收匹配断言的分支指令将发出,但是利用互补断言编码的其他指令将不发出。

[0090] 指令块620的依赖性图640还被图示为指令节点阵列650和其对应的操作数目标655和656。这图示了块指令620、对应的指令窗口条目以及由指令所表示的底层数据流程图之间的对应性。此处,译码的指令READ 630和READ 631准备好发出,因为其不具有输入依赖性。当其发出并且执行时,从寄存器R6和R7读取的值被写入到ADD 632的右操作数缓冲器和左操作数缓冲器中,这使得ADD 632的左操作数和右操作数"就绪"。因此,ADD 632指令变为就绪、发出到ALU、执行、并且和被写入到TLEI指令633的左操作数。

[0091] VIII.示例基于块的指令格式

[0092] 图7是图示用于指令头710、通用指令720、分支指令730和存储器访问指令740(例如,存储器加载或存储指令)的指令格式的一般化示例的图。指令格式可以用于根据在指定操作模式的指令头中指定的多个执行标志而执行的指令块。指令头或者指令中的每一个根据位数而被标记。例如,指令头710包括四个32位的字并且从其最低有效位(lsb)(位0)被标记直到其最高有效位(msb)(位127)。如所示出的,指令头包括写入掩码字段、多个执行标志字段、指令块大小字段和指令头ID位(指令头的最低有效位)。在一些示例中,指令头710包括附加元数据715和/或716,其可以用于控制指令块执行和性能的附加方面。

[0093] 图7中描绘的执行标志字段占据指令块头710的第6至第13位并且指示用于执行指令块的一种或多种操作模式。例如,操作模式可以包括核融合操作、向量模式操作、分支预测器抑制、存储器依赖性预测器抑制、块同步、块之后的中断、块之前的中断、块下降、和/或按序或确定性的指令执行。块同步标志占据指令块的第9位并且当被设置为1时抑制指令块的推测执行。抑制推测执行例如当诸如存储条件指令的共享存储器操作或其他共享存储器操作由指令块执行、以防止违反ISA规格的存储器危害时是高度期望的。

[0094] 退出类型字段包括可以用于指示被编码在指令块内的控制流指令的类型的数据。例如,退出类型字段可以指示指令块包括以下各项中的一项或多项:顺序分支指令、偏移分支指令、间接分支指令、调用指令、和/或返回指令。在一些示例中,分支指令可以是用于在

指令块之间传递控制流的任何控制流指令，其包括相对地址和/或绝对地址，并且使用有条件的断言或者无条件的断言。除了确定隐式控制流指令之外，退出类型字段可以用于分支预测和推测执行。

[0095] 所图示的通用块指令720被存储为一个32位的字，并且包括操作码字段、断言字段、广播ID字段(BID)、向量操作字段(V)、单指令多数据(SIMD)字段、第一目标字段(T1)、以及第二目标字段(T2)。对于具有比目标字段更多的消费者的指令而言，编译器可以使用移动指令来构建扇出树，或者其可以将高扇出指令指派到广播。广播支持通过轻量网络而将操作数发送到核中的任何数目的消费者指令。

[0096] 虽然由通用指令720概述的通用指令格式可以表示由基于块的处理器处理的一些或全部指令，但是本领域的技术人员将容易理解到，即使对于ISA的特定示例而言，指令字段中的一个或多个指令字段也可以偏离用于特定指令的通用格式。操作码字段指定由指令720执行的(一个或多个)操作，诸如存储器读/写、寄存器加载/存储、加法、减法、乘法、除法、移位、旋转、系统操作或者其他适合的指令。断言字段指定指令将执行的条件。例如，断言字段可以指定值“真”，并且指令将仅在对应的条件标志匹配指定的断言值的情况下执行。在一些示例中，断言字段至少部分地指定哪一个被用于比较断言，而在其他示例中，执行在由先前指令(例如，指令块中的先前指令)设定的标志上被断定。在一些示例中，断言字段可以指定指令将总是被执行或者绝不被执行。因此，断言字段的使用可以通过减少分支指令的数目来允许更密集的目标代码、改进的能量效率以及改进的处理器性能。

[0097] 目标字段T1和T2指定基于块的指令的结果被发送到的指令。例如，在指令槽7处的ADD指令可以指定其计算结果将被发送到槽3和10处的指令，包括操作数槽(例如，左操作数、右操作数、或断言操作数)的指定。取决于特定指令和ISA，所图示的目标字段之一或二者可以由其他信息替换，例如，第一目标字段T1可以由中间操作数、附加操作码、指定两个目标等来替换。

[0098] 分支指令730包括操作码字段、断言字段、广播ID字段(BID)以及偏移字段。操作码和断言字段在如关于通用指令所描述的格式和功能方面是类似的。偏移可以以四个指令的组为单位被表达，因此扩展在其上可以执行分支的存储器地址范围。利用通用指令720和分支指令730示出的断言可以用于避免指令块内的附加分支。例如，特定指令的执行可以根据先前指令的结果(例如，两个操作数的比较)被断定。如果断言是假的，则指令将不提交由特定指令计算出的值。如果断言值未匹配所要求的断言，则指令不发出。例如，BRO_F(断言假)指令在其被发送假断言值的情况下将发出。

[0099] 应当容易理解到，如在此所使用的，术语“分支指令”不限于将程序执行改变到相对存储器位置，而且还包括跳转到绝对或者符号存储器位置、子例程调用和返回，以及可以修改执行流的其他指令。在一些示例中，通过改变系统寄存器(例如，程序计数器PC或者指令指针)的值来修改执行流，而在其他示例中，可以通过修改被存储在存储器中的指定位置处的值来改变执行流。在一些示例中，跳转寄存器分支指令用于跳转到被存储在寄存器中的存储器位置。在一些示例中，分别使用跳转和链接和跳转寄存器指令来实现子例程调用和返回。

[0100] 存储器访问指令740格式包括操作码字段、断言字段、广播ID字段(BID)、立即字段(IMM)和目标字段(T1)。操作码字段、广播字段、断言字段在格式上类似并且如关于通用指

令所描述的起作用。例如，对特定指令的执行可以在先前指令的结果(例如，两个操作数的比较)上断言。如果断言为假，则指令将不提交由该特定指令计算的值。如果断言值与所要求的断言不匹配，则指令不发出。立即字段可以被用作针对被发送到加载或存储指令的操作数的偏移。操作数加上(移位的)立即偏移被用作针对加载/存储指令的存储器地址(例如，用于从存储器中读取数据或者将数据存储到存储器中的地址)。对于一些指令，诸如存储条件指令，目标字段T1 745用于指定通过执行生成的状态指示符将被存储在哪里。例如，目标字段T1 745可以指定寄存器来(例如，基于加载链接地址和链接值)存储指示存储条件指令是否成功执行的状态指示符。后续指令块可以检查状态指示符值并且采取合适的动作(例如，通过冲洗指令块，使指令块重新执行，引发异常，等等)。

[0101] IX.示例处理器状态图

[0102] 图8是图示在指令块被映射、执行和引退时被分配给指令块的多个状态的状态图800。例如，状态中的一个或多个可以在执行指令期间根据一个或多个执行标志来分配。应当容易理解，图8中示出的状态是所公开的技术的一个示例，但是在其他示例中，指令块可以具有附加的或更少的状态、以及具有与状态图8中描绘的状态不同的状态。在状态805处，指令块未被映射。指令块可以驻留在耦合到基于块的处理器的存储器中、被存储在诸如硬盘驱动器或闪存驱动器的计算机可读存储设备上，并且可以在处理器本地或者被定位在远程服务器处并且可使用计算机网络访问。未映射的指令还可以至少部分地驻留在耦合到基于块的处理器的高速缓存存储器中。

[0103] 在指令块映射状态810处，针对基于块的处理器(例如指令调度器)的控制逻辑可以用于监视基于块的处理器的处理核资源，并且将指令块映射到处理核中的一个或多个。

[0104] 控制单元可以将指令块中的一个或多个映射到处理器核和/或特定处理器核的指令窗口。在一些示例中，控制单元监视先前已经执行了特定指令块的处理器核并且可以重新使用针对仍然驻留在“预热”的处理器核上的指令块的译码的指令。一旦一个或多个指令块已经被映射到处理器核，指令块就可以前进到取指状态820。

[0105] 当指令块处于取指状态820(例如，指令取指)时，映射的处理器核从基于块的处理器的存储器系统中取指计算机可读块指令，并且将它们加载到与特定处理器核相关联的存储器中。例如，针对指令块的取指的指令可以被取指并且被存储在处理器核内的指令高速缓存中。指令可以使用核互连而被传送到处理器核。一旦已经取指了指令块的至少一个指令，该指令块就可以进入指令译码状态830。

[0106] 在指令译码状态830期间，取指的指令的各个位被译码成能够由处理器核用于控制特定指令的执行的信号，包括对指示存储器访问指令的相对顺序的标识符的生成。例如，译码的指令可以被存储在以上在图2中示出的存储器存储中的一个中。译码包括生成针对译码的指令的依赖性、针对译码的指令的操作数信息和针对译码的指令的目标。一旦指令块的至少一个指令已经被译码，指令块就可以前进到发出状态840。

[0107] 在发出状态840期间，指令依赖性被评估以确定指令是否准备好执行。例如，指令调度器可以监视指令的源操作数并且(针对断言的指令)断言操作数必须在指令准备好发出之前可用。对于一些编码，某些指令还必须根据指定顺序发出。例如，存储器加载存储操作根据编码于每个指令中的LSID值来排序。在一些示例中，多于一个指令准备好同时发出，并且指令调度器选择准备好发出的指令之一来发出。指令可以使用它们的IID来识别以方

便评估指令依赖性。一旦已经发出了指令块的至少一个指令，针对(一个或多个)发出的指令的源操作数可以被取指(或者在旁路路径上维持)，并且指令块可以前进到执行状态850。

[0108] 在执行状态850期间，与指令相关联的操作使用例如如以上关于图2所讨论的功能单元260来执行。如以上所讨论的，执行的功能可以包括算术功能、逻辑功能、分支指令、存储器操作和寄存器操作。与处理器核相关联的控制逻辑监视指令块的执行，并且一旦确定了指令块可以被提交或者指令块要被中止，指令块状态就被设置为提交/中止状态860。在一些示例中，控制逻辑使用针对指令块的写掩码和/或存储掩码来确定执行是否已经足够地前进到提交指令块。

[0109] 在提交/中止状态860处，处理器核控制单元确定由指令块执行的操作可以被完成。例如存储器加载存储操作、寄存器读/写、分支指令和其他指令将必定根据指令块的控制流来执行。对于条件存储器指令，数据将被写入到存储器，以及在提交/中止状态860期间生成的指示成功的状态指示符值。备选地，如果指令块要被中止，例如，因为指令的依赖性中的一个或多个未被满足、或者指令在未被满足的针对指令块的断言上被推测地执行，则指令块被中止使得其将不会影响存储器或寄存器文件中的指令的顺序的状态。不管指令块是已经提交了还是中止了，指令块都进行到状态870以确定指令块是否应当被刷新。如果指令块被刷新，则处理器核通常使用新数据值(特别是通过刚刚提交的块的执行而更新的寄存器和存储器)重新执行指令块，并且直接前进到执行状态850。因此，可以避免在映射、取指和译码指令块中花费的时间和能量。备选地，如果指令块未被刷新，则指令块进入空闲状态880。

[0110] 在空闲状态880中，执行指令块的处理器核可以通过例如对处理器核内的硬件掉电、同时维持针对指令块的译码的指令的至少一部分而被空闲。在某个时刻，控制单元确定890处理器核上的空闲指令块是否要被刷新。如果空闲指令块要被刷新，则指令块可以在执行状态850处重新开始执行。备选地，如果空闲指令块不要被刷新，则指令块不被映射并且处理器核可以被冲洗并且随后指令块可以被映射到冲洗的处理器核。

[0111] 尽管为便于解释，状态图800将指令块的状态图示为在单个处理器核上执行，但是对相关领域的普通技术人员而言应当容易理解，在某些示例中，多个处理器核可以用于并发执行给定指令块的多个实例。

[0112] X.示例基于块的处理器和存储器配置

[0113] 图9是图示包括基于块的处理器910的装置的图900，基于块的处理器910包括被配置为执行包括用于包括存储器同步和存储器锁定的存储器操作的指令的指令块的控制单元920。控制单元包括核调度器925，其控制：对用于执行指令处理的核的分配和解除分配，对在核、寄存器文件、存储器接口和/或I/O接口中的任何项之间的输入数据和输出数据的控制。控制单元920还可以包括用于执行某些存储器操作的专用寄存器。

[0114] 基于块的处理器910还包括被配置为取指并且执行指令块的一个或多个处理器核930-937。核中的每个核包括控制指令块中的指令被取指、译码、发出和执行的顺序的指令调度器(例如，指令调度器941)。图示的基于块的处理器910具有至多八个核，但是在其他示例中可以存在1个、2个、4个、64个、512个、1024个或其他数量的基于块的处理器的核。基于块的处理器910被耦合到存储器950并且被耦合到计算机可读存储介质盘960，存储器950包括指令块A和B，其包括实现所公开的存储器操作的指令(分别为955和956)，计算机可读存

储介质盘960存储用于执行所公开的存储器操作的指令965。

[0115] XI.操作基于块的处理器的示例方法

[0116] 图10是概述如可以在所公开的技术的某些示例中执行的操作基于块的处理器的示例方法的流程图1000。例如，基于块的处理器的定制或ASIC实现、或者诸如以上讨论的那些的基于块的处理器的FPGA实现可以用于执行图示的方法。

[0117] 在处理块1010处，生成针对指令的事务块的结构依赖性。指令中的每个指令具有分配的指令标识符，例如，指令在指令块内的相对位置的索引可以被用作指令标识符。指令可以利用使用指令标识符的其相应的目标指令而被编码。处理器将由指令生成的结果作为针对目标指令的输入操作数发送到(一个或多个)指定目标指令。在其他示例中，指令标识符可以被编码在指令块的头部分或脚部分中。指令依赖性可以包括但不限于针对指令的输入操作数和针对到存储器的加载和存储的排序信息。

[0118] 在一些示例中，仅仅针对指令的事务块的部分生成依赖性。这包括在所有指令块已经被取指之前取指、译码并且执行指令的情况。基于块的处理器不需要等待所有指令块被译码以开始操作。在一些示例中，指令利用唯一地识别事务指令块中的指令的指令标识符来编码。例如，编码指令标识符的一种方式是通过指令块内的指令的相对位置。其他指令可以随后瞄准用于通过指定指令所处的指令ID发送数据操作数和断言的特定指令。例如，如果指令被定位在针对指令块内的指令的第四可用存储器地址处，则另一指令可以通过指定为四的指令ID值来瞄准该指令。如本领域的普通技术而言将容易理解的，用于编码指令标识符的其他技术可以被使用。然而，重要的是要指出，指令标识符以静态方式被编码，其允许对指令块内的指令依赖性的更容易的跟踪。这与指令不会彼此识别而是代替地使用寄存器或主存储器来在指令之间传递值的传统CISC和RISC方法形成对比。可以针对指令生成的指令依赖性的特定示例包括：指令是否在广播信道上接收到操作数，指令断言是否就绪，指令的输入操作数是否就绪，指令是否应当被抑制(例如，因为指令已经发出)。在一些示例中，就绪状态可以被进一步分成：译码的就绪状态，其可以由指令译码器初始化并且针对指令块的后续实例保持静态；或者活动就绪状态，其可以针对执行指令块的每个实例而被刷新。因此，译码的就绪状态指示指令是否需要编码的特定依赖性中的每个依赖性，而活动就绪状态指示针对执行讨论中的指令块的特定实例已经满足了指定依赖性。在已经生成了指令依赖性之后，方法前进到处理块1020。

[0119] 在处理块1020处，基于在处理块1010处生成的指令依赖性，一旦已经满足了对应指令的依赖性，就信号通知一个或多个指令已准备好。对于一些指令，例如移动立即指令，指令的依赖性被立即满足，因为该指令不需要任何依赖性。因此，这样的指令准备好甚至在任何寄存器值或其他依赖性已经到达指令块之前立即发出。对于其他指令，指令必须等待其合适的依赖性被生成和满足。就绪指令之后等待被选择以由指令调度器发出。在一些示例中，其对应的“就绪”位被设置。对于给定指令窗口，若干指令可以就绪或者变为就绪，同时仅仅一个或少量这样的指令可以在每个周期发出。发出的指令立即开始执行。在一些示例中，就绪指令被选择用于以它们被布置在指令块中的顺序来发出和后续执行。在其他示例中，其他方案可以用于确定在存在比处理器核的资源允许的可用于发出的更多指令的情况下，执行哪个就绪指令。例如，在可以并发执行两个指令的核中，当三个指令准备好发出时，两个指令可以被选择以开始执行并且第三个指令被延迟直到执行槽可用。

[0120] 在处理块1030处，执行下一发出的指令。例如，在图2的微架构的检测到的流水线的功能单元250和265可以用于执行指令。指令结果可以使用加载存储队列270被存储在存储器中，写入到寄存器文件290，或者被发回到流水线中以用于由使用操作数缓冲器239或旁路机构的其他指令执行以直接前进到执行流水线寄存器245。以这种方式，操作数可以被直接发送以用于由取决于当前执行指令的指令来使用。

[0121] XII.利用刷新来操作基于块的处理器的示例方法

[0122] 图11是概述如可以在所公开的技术的某些示例中执行的执行指令块的示例的流程图1100。例如，以上关于图2所描述的基于块的处理器(包括使用以上在图2中讨论的微架构的实现)可以用于实现方法的某些示例。在一些示例中，方法可以使用定制或专用集成电路(ASIC)来执行，而在其他示例中，可以使用诸如FPGA的可重新配置逻辑电路。在处理块1110处，针对指令的事务块的指令头和一个或多个指令可以被并发取指。在一些示例中，在特定时钟周期中取指指令头的仅仅一部分。在某些FPGA示例中，多读端口块RAM允许对指令头和块指令的并发取指。这在某些应用中可以是有利的，因为其允许并发指令头的译码、同时没有依赖性的指令块指令可以被立即发出并且开始执行。在方法的其他示例中，一次仅仅取指存储器的单个字。

[0123] 在处理块1120处，指令头部分被译码。在处理块1110处取指的指令也被译码。指令译码器生成针对指令的就绪状态数据并且将针对指令的就绪状态数据存储在存储器中，其可以使用与指令相关联的指令标识符来索引。在一些示例中，针对给定指令的指令标识符基于指令块内的指令的相对位置，并且指令块中的其他指令基于指令的标识符来引用该指令。译码的就绪状态数据可以被存储在由被映射到FPGA的LUT上的分布式RAM组成的RAM中。在其他示例中，其他形式的RAM、或锁存器、或触发器用于存储译码的指令状态，其包括由指令调度器使用的译码的就绪状态数据。例如，就绪状态数据可以被存储在被耦合到图2的指令调度器235的存储器中，而用于控制微架构流水线的执行的控制信息可以被存储在译码的指令存储236中。指令调度器235还可以用于控制来自操作数缓冲器239的操作数读取。

[0124] 在处理块1130处，处理器在如由就绪状态数据指示的其依赖性可用时发出并且执行指令中的至少一个指令。例如，多个位可以被存储在FPGA分布式LUT RAM或块RAM中，并且当所有位与指定模式匹配时，指令被发出。因为仅仅有限数量的指令可以使用执行流水线在给定时钟周期中执行，所以如果多于该数量的支持的指令变为准备好发出，则某些指令必须等待。诸如使用“准备好发出”位掩码加优先级编码器的技术可以用于确定哪些指令应当在下次被执行。

[0125] 在处理块1140处，处理器使用旁路机构，以便旁路针对用于执行指令的操作数的至少一个源的其操作数缓冲器。通过旁路操作数缓冲器(例如，操作数缓存器239)，流水线可以通过将下一指令的数据直接发送到流水线寄存器245来避免一个时钟周期的延迟。在其他示例中，基于块的处理器不使用这样的旁路机构。使用旁路机构还可以避免关于在操作数缓冲器239上可用的读端口的数量的限制。例如，在一些微架构中，读端口的数量或者可以在相同时钟周期中被写入到操作数缓冲器的操作数的类型可能受限制。例如，操作数缓冲器可能仅仅能够在每个时钟周期接受一个左操作数和一个右操作数。通过还允许使用旁路，当存在多于一个指令执行时，可以通过将针对下一指令的操作数直接发送到流水线寄存器245、同时由另一指令并发生成的操作数被安全地存储在操作数缓冲器239中，来避

免流水线暂停。

[0126] 在处理块1150处，指令块到达执行的提交阶段并且执行被延迟直到指令块到达该提交阶段的任何存储器和/或寄存器操作。例如，加载存储队列270或者寄存器文件290内的影子寄存器可以存储数据直到实际上确定其将通过执行指令块而被写入。这在例如多个指令写入到相同存储器地址、或者存储器或寄存器操作不是由于指令块的推测执行而发生的情况下维持机器状态的一致性。方法之后前进以处理程序中的下一指令块。如果相同指令块要例如在程序循环中被再次执行，则方法前进到处理块1160。在一些示例中，指令块可以全部或部分地被译码并且可用，在这种情况下方法还可以前进到处理块1160以便刷新指令块的指令窗口。在其他示例中，例如在指令块尚未被完全译码的情况下，或者在指令块被新执行的情况下，方法前进到处理块1170。

[0127] 在处理块1160处，指令窗口就绪状态例如通过重置针对指令块的活动就绪状态数据、但是重新使用针对指令块的先前译码的就绪状态数据而被刷新。因此，某些依赖性可以被确定，并且允许执行前进，而不对来自指令块的先前执行的工作中的一些工作重新译码和重新使用。在处理块1170处，整个指令窗口就绪状态被重置。不管是刷新还是重置，方法前进到处理块1130以便发出并且执行附加的指令。在附加的指令信息必须被取指和译码的情况下，方法可以备选地前进到处理块1110或处理块1120(当适合于执行指令块时)。

[0128] XIII.配置可重新配置逻辑器件的示例方法

[0129] 图12是概述如可以在所公开的技术的某些示例中执行的配置可重新配置逻辑器件的示例方法的流程图1200。例如，以上关于图3所讨论的FPGA可以被配置为使用以上关于图2讨论的示例微架构来实现图1的基于块的处理器。

[0130] 在处理块1210处，将基于块的处理器部件的描述映射到FPGA的可重新配置逻辑器件部件。例如，处理器设计者可以指定以硬件描述语言的对基于块的处理器的描述，硬件描述语言诸如为SystemVerilog、SystemC、Verilog或硬件描述语言的任何其他适合的组合。在一些示例中，以诸如C或C++的传统编程语言编写的描述用于描述基于块的处理器的至少部分。基于块的处理器的描述可以包括以上讨论的新颖部件中的任何部件。在一些示例中，设计者可以指定要由处理器微架构的元件瞄准的特定FPGA单元。例如，设计者可以指定指令高速缓存和/或数据高速缓存使用FPGA的块RAM资源来实现。在一些示例中，程序员可以使用由FPGA供应商提供的可用宏来实现FIFO缓冲器、移位寄存器和使用针对该FPGA的经济映射的其他部件。

[0131] 在处理块1220处，产生配置比特流以用于实现针对基于块的处理器的电路，其包括使用由指令标识符索引的就绪状态数据的乱序指令调度器。例如，以硬件描述语言表示的基于块的处理器的描述可以被编译以生成网表，并且该网表转而用于生成比特流文件。在比特流文件中指示的信号可以被施加到FPGA的配置接口以便将FPGA配置为执行用于实现根据所公开的技术的基于块的处理器的功能。

[0132] 在处理块1230处，可重新配置逻辑器件使用在处理块1220处生成的比特流来配置。例如，一些FPGA具有用于将数据连续地流传输到FPGA的配置存储器中的回读端口，由此配置FPGA。在一些示例中，FPGA的配置存储器通过并行或其他可寻址端口而被寻址。在一些示例中，具有与FPGA类似的结构的可重新配置逻辑器件可以被配置一次，而不可被重新配置。在其他示例中，FPGA可以被电擦除并且被重新写入以便提供新配置。在一些示例中，

FPGA在无论何时集成电路被重新上电时都被重新配置，而在其他示例中，FGPA配置跨重复的电力周期维持状态。

[0133] XIV.示例计算环境

[0134] 图13图示了在其中可以实现所描述的实施例、技术和工艺(包括配置基于块的处理器)的适合的计算环境1300的一般示例。例如，计算环境1300可以实现所公开的技术，以用于配置处理器以实现所公开的基于块的处理器架构和微架构、和/或将代码编译成计算机可执行指令和/或用于执行这样的操作的配置比特流，如在此所描述的。

[0135] 计算环境1300不旨在提出关于技术的使用或者功能的范围的任何限制，因为技术可以被实现在不同的通用或者专用计算环境中。例如，所公开的技术可以利用其他计算机系统配置被实现，包括手持式设备、多处理器系统、可编程消费者电子产品、网络PC、微型计算机、大型计算机，等等。所公开的技术还可以被实践在分布式计算环境中，其中任务由通过通信网络链接的远程处理设备来执行。在分布式计算环境中，程序模块(包括用于基于块的指令块的可执行指令)可以被定位在本地存储器存储设备和远程存储器存储设备二者中。

[0136] 参考图13，计算环境1300包括至少一个基于块的处理单元1310、以及存储器1320。在图13中，该最基本配置1330被包括在虚线内。基于块的处理单元1310执行计算机可执行指令，并且可以是真实处理器或者虚拟处理器。在多处理系统中，多个处理单元执行计算机可执行指令以增加处理能力，并且由此多个处理器可以同时运行。存储器1320可以是易失性存储器(例如，寄存器、高速缓存、RAM)、非易失性存储器(例如，ROM、EEPROM、闪存存储器等)、或者以上两者的组合。存储器1320存储可以例如实现在此所描述的技术的软件1380、图像和视频。计算环境可以具有附加的特征。例如，计算环境1300包括存储装置1340、一个或多个输入设备1350、一个或多个输出设备1360以及一个或多个通信连接1370。诸如总线、控制器或者网络的互连机制(未示出)将计算环境1300的部件互连。通常，操作系统软件(未示出)提供用于在计算环境1300中执行的其他软件的操作环境，并且协调计算环境1300的部件的活动。

[0137] 存储装置1340可以是可移除或者不可移除的，并且包括磁盘、磁带或者磁带盒、CD-ROM、CD-RW、DVD或者可以用于存储信息并且可以在计算环境1300内访问的任何其他介质。存储装置1340存储用于软件1380的指令、插件数据和消息，其可以用于实现在此所描述的技术。

[0138] (一个或多个)输入设备1350可以是触摸输入设备，诸如键盘、小键盘、鼠标、触屏显示器、笔、或轨迹球、语音输入设备、扫描设备、或者向计算环境1300提供输入的另一设备。对于音频而言，(一个或多个)输入设备1350可以是接受以模拟或者数字形式的音频输入的声卡或者类似设备，或者可以是向计算环境1300提供音频样本的CD-ROM读取器。(一个或多个)输出设备1360可以是显示器、打印机、扬声器、CD刻录机、或者提供来自计算环境1300的输出的另一设备。

[0139] (一个或多个)通信连接1370实现通过通信介质(例如，连接网络)与另一计算实体的通信。通信介质传达诸如计算机可执行指令、压缩图形信息、视频、或者调制数据信号中的其他数据的信息。(一个或多个)通信连接1370不限于有线连接(例如，兆比特或吉比特以太网、Infiniband、通过电连接或通过光纤连接的光纤信道)，而且还包括无线技术(例如，

经由蓝牙、WiFi(IEEE 802.11a/b/n)、WiMax、蜂窝、卫星、激光、红外的RF连接)以及用于提供用于所公开的方法的网络连接的其他适合的通信连接。在虚拟主机环境中,(一个或多个)通信连接可以是由虚拟主机所提供的虚拟化网络连接。

[0140] 可以使用实现计算云1390中的所公开的技术的全部或部分的计算机可执行指令来执行所公开的方法的一些实施例。例如,所公开的编译器和/或基于块的处理器的服务器被定位在计算环境中,或者所公开的编译器可以在被定位在计算云1390中的服务器上执行。在一些示例中,所公开的编译器在传统的中央处理单元(例如,RISC或者CISC处理器)上执行。

[0141] 计算机可读介质是可以在计算环境1300内访问的任何可用介质。以示例而非限制的方式,利用计算环境1300,计算机可读介质包括存储器1320和/或存储装置1340。如应当容易理解的,术语计算机可读存储介质包括用于数据存储的介质(诸如存储器1320和存储装置1340)而非传输介质(诸如调制数据信号)。

[0142] XV.所公开的技术的附加示例

[0143] 在这里根据以上讨论的示例来讨论所公开的主题的附加示例。例如,以上关于图1、图2和图9讨论的基于块的处理器的方面可以用于实现这些附加示例,包括FPGA、诸如以上关于图3和图4讨论的那些FPGA。

[0144] 在所公开的技术的某些示例中,基于块的处理器中的全部或一部分通过将FPGA配置为包括用于执行在基于块的处理器ISA中表达的程序的结构来实现。在一些示例中,处理器被实现在嵌入式设备中,诸如用于部署在物联网(IoT)网络中。在一些示例中,诸如高速缓存的结构,以及在指令调度器、加载存储队列和/或寄存器文件中使用的存储被实现在具有单个写端口或单个读端口的存储器中。在其他示例中,这些结构中的一个或多个被实现在具有多个读端口和/或写端口的存储器中。在一些示例中,指令块的指令块头以及一个或多个指令可以从存储器和/或指令高速缓存并发地取指。在一些示例中,旁路机构允许从微架构流水线的执行部分生成的操作来旁路操作数,由此允许具有共享的或链式的依赖性的指令的背靠背发出。在一些示例中,旁路机构允许当在执行时钟周期期间与指令窗口操作数缓冲器上的写入端口相比存在更多生成的操作数时,避免流水线暂停。

[0145] 在一些示例中,调度器可以使用译码的或先前译码的指令依赖性,来在它们已经被取指之前唤醒并且发出指令。在一些示例中,针对指令调度器的存储可以被分成两个或更多个部分,以便将存储映射到FPGA的两个或更多个物理存储单元。在一些示例中,指令调度器包括并行调度器。在其他示例中,指令调度器包括一个或多个事件队列和一个或多个指令队列。在一些示例中,调度器被配置为在重新执行指令块后刷新指令中的一些但不是全部。

[0146] 在所公开的技术的一些示例中,一种装置包括基于块的处理器,并且基于块的处理器包括:指令译码器,其被配置为针对指令的事务块的至少部分生成译码的就绪依赖性,该指令中的每个指令与编码在该事务块中的不同指令标识符相关联;以及指令调度器,其被配置为从指令的该事务块不按程序顺序地发出指令。指令调度器可以基于确定针对该指令的该译码的就绪依赖性被满足来发出指令。确定可以包括访问存储使用被编码在指令的该事务块中的相应指令标识符的该译码的就绪依赖性的存储装置。

[0147] 在一些示例中,该确定包括使用由执行的指令信号通知的指令标识符来生成用于

访问该存储装置的索引。在一些示例中，该处理器还包括指令取指单元，该指令取指单元被配置为取指针对指令的该事务块的头的至少部分，并且并发取指指令的该事务块的指令的至少部分。在一些示例中，该指令取指单元包括存储所取指的头的第一块存储器和存储所取指的指令的第二块存储器。在一些示例中，第一块存储器可以存储所取指的头、所取指的指令、或者头和指令的混合。在一些示例中，基于块的处理器是利用可重新配置逻辑器件实现的软核处理器。在一些示例中，基于块的处理器被配置为利用优先级编码器并且基于针对下一指令编码的指令标识符来选择要执行的指令块的下一指令。在一些示例中，这包括使用准备好发出位掩码。

[0148] 在一些示例中，该指令调度器被耦合到数据操作数缓冲器，该数据操作数缓冲器存储用于由该指令在后续时钟周期中执行的生成的数据。在一些示例中，处理器包括旁路逻辑，该旁路逻辑允许数据操作数被转发以用于由指令在针对指令的当前执行时钟周期之后紧跟着的后续时钟周期中执行，从而允许该数据操作数被转发而不将该数据操作数存储在该数据操作数缓冲器中。在一些示例中，该数据操作数缓冲器被配置为每个时钟周期为不超过一个的指令来存储操作数数据，并且处理器还包括旁路逻辑，该旁路逻辑允许在不同数据操作数被存储在该数据操作数缓冲器中时、针对不同指令的数据操作数在相同时钟周期中被转发到该处理器的执行单元。

[0149] 在所公开的技术的一些示例中，一种可重新配置逻辑器件(诸如可重新编程或一次性编程FPGA)被配置为执行基于块的处理器指令集。该可重新配置逻辑器件被配置为使用以下各项来执行指令：多个查找表(LUT)；指令高速缓存，其被配置为从由该处理器从耦合到该可重新配置逻辑器件的存储器所取指的指令块接收指令；指令调度器，其被配置为存储由编码在指令块中的指令标识符索引的就绪状态数据，并且还被配置为在该就绪状态数据指示针对指令的所有依赖性被满足时发出指令；一个或多个执行单元，其被配置为执行由发出的指令指定的操作；数据高速缓存，其被配置为存储通过使用该执行单元执行指令而从该存储器读取的和/或写入到该存储器的数据；以及寄存器文件，其被配置为存储由该指令的寄存器操作数指定的数据。该数据高速缓存可以被配置为延迟完成存储器操作直到该指令块到达执行的提交阶段。

[0150] 在一些示例中，指令高速缓存和/或数据高速缓存利用可重新配置逻辑器件的块随机存取存储器(RAM)资源来实现。在一些示例中，分布式RAM或LUT RAM用于实现指令高速缓存和/或数据高速缓存。

[0151] 在一些示例中，该指令调度器和/或该寄存器文件利用使用该多个LUT的一部分形成的随机存取存储器(RAM)来实现。在一些示例中，指令调度器和/或该寄存器文件利用块RAM来实现。在一些示例中，LUT由被耦合到一个或多个多路复用器的静态随机存取存储器(RAM)单元形成。

[0152] 在所公开的处理器的一些示例中，该指令调度器被耦合到：译码的指令字存储器，其被配置为存储针对所接收的指令的至少部分的译码的指令控制数据；以及多个一个或多个操作数缓冲器，其被配置为存储用于执行所接收的指令的操作数数据。在一些示例中，该可重新配置逻辑器件还被配置为通过刷新并且重新执行该指令块来执行指令块的后续实例。在一些示例中，该就绪状态数据包括在该刷新后不被清除的译码的就绪状态信息、以及在该刷新后被清除的活动就绪状态数据。

[0153] 在一些示例中，调度器被配置为重新使用针对用于执行该指令块的后续实例的指令的所存储的就绪状态数据的至少部分，并且处理器被配置为不重新取指并且不重新译码用于执行该后续实例的该指令。在一些示例中，该指令调度器被配置为通过将所存储的就绪状态数据与通过执行另外的指令块生成的一个或多个信号相比较，来确定所有指令的依赖性被满足。

[0154] 在所公开的技术的一些示例中，一种形成具有一个或多个可配置逻辑器件的基于块的处理器的方法包括：产生配置比特流，该配置比特流包括用于利用该可配置逻辑器件来实现针对该基于块的处理器的电路的配置信息，并且一旦可配置逻辑器件已经使用配置比特流来编程，针对该基于块的处理器的该电路就包括乱序指令调度器，该乱序指令调度器被配置为基于存储在存储器中的、通过唯一地识别事务指令块的每个相应指令的指令标识符索引的就绪状态数据来发出指令。

[0155] 在一些示例中，形成基于块的处理器的方法包括将指令高速缓存和/或数据高速缓存的描述映射到利用可配置逻辑器件实现的块RAM硬件的方法。在一些示例中，方法包括将部件中的至少一个部件的描述映射到使用多个LUT实现的RAM。在一些示例中，方法包括将配置比特流施加到包括对设备编程的可配置逻辑器件的集成电路的配置端口。在一些示例中，方法包括将基于块的处理器的硬件描述语言规格映射到网表、该网表包括要利用可配置逻辑器件实现的硬件元件的描述，并且得到的网表用于产生配置比特流。在一些示例中，方法包括通过将硬件描述规格与基于块的处理器的硬或软宏描述进行组合，来将计算系统的硬件描述语言规格映射到配置比特流。在一些示例中，方法包括将配置比特流存储在计算机可读存储设备或存储器中。

[0156] 鉴于所公开的主题的原理可以应用的许多可能实施例，应当认识到所图示的实施例仅是优选的示例，并且不应该当作将权利要求的范围限于那些优选的示例。相反，要求保护的主题的范围由所附的权利要求进行限定。我们因此对我们的发明要求保护落在这些权利要求的范围内的全部内容。

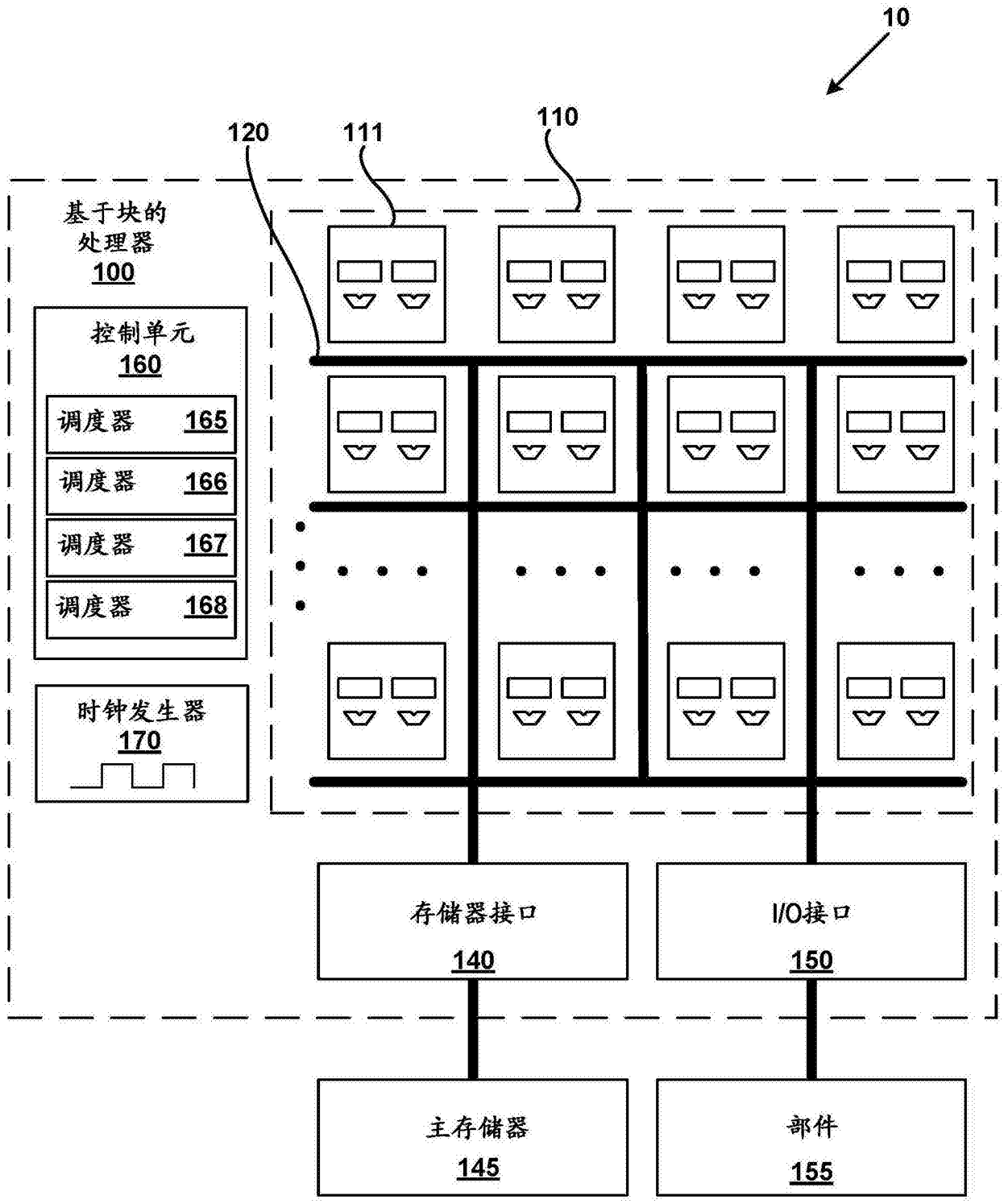

10
120
111
110
基于块的
处理器
100
控制单元
160
调度器 165
调度器 166
调度器 167
调度器 168
时钟发生器
170
存储器接口
140
I/O接口
150
主存储器
145
部件
155

图1

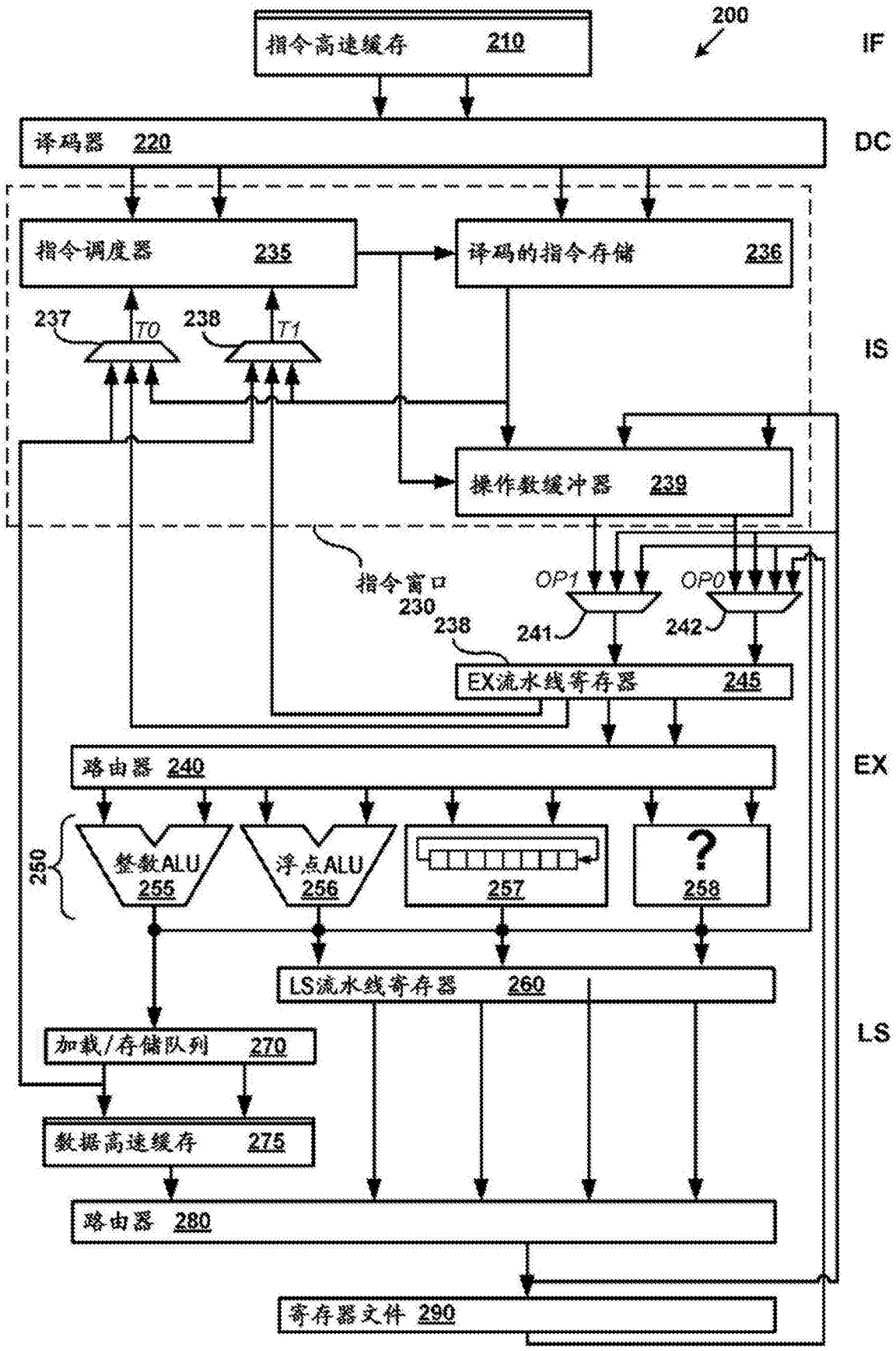

200
指令高速缓存 210
IF
译码器 220
DC
指令调度器 235
译码的指令存储 236
237
T0
238
T1
IS
操作数缓冲器 239
指令窗口
230
OP1
OP0
238
241
242
EX流水线寄存器 245
路由器 240
EX
250
整数ALU
255
浮点ALU
256
257
?
258
LS流水线寄存器 260
LS
加载/存储队列 270
数据高速缓存 275
路由器 280
寄存器文件 290

图2

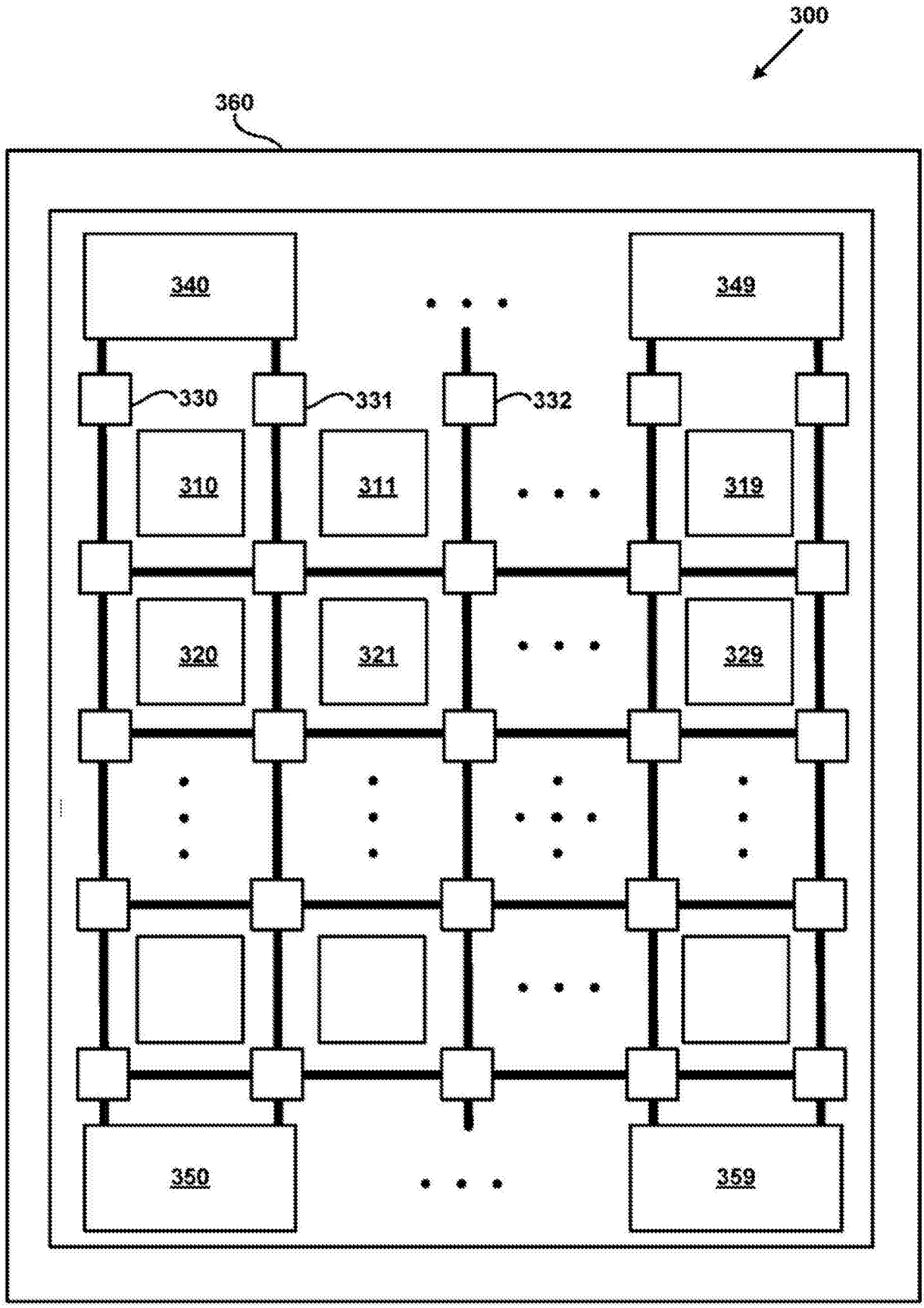


图3

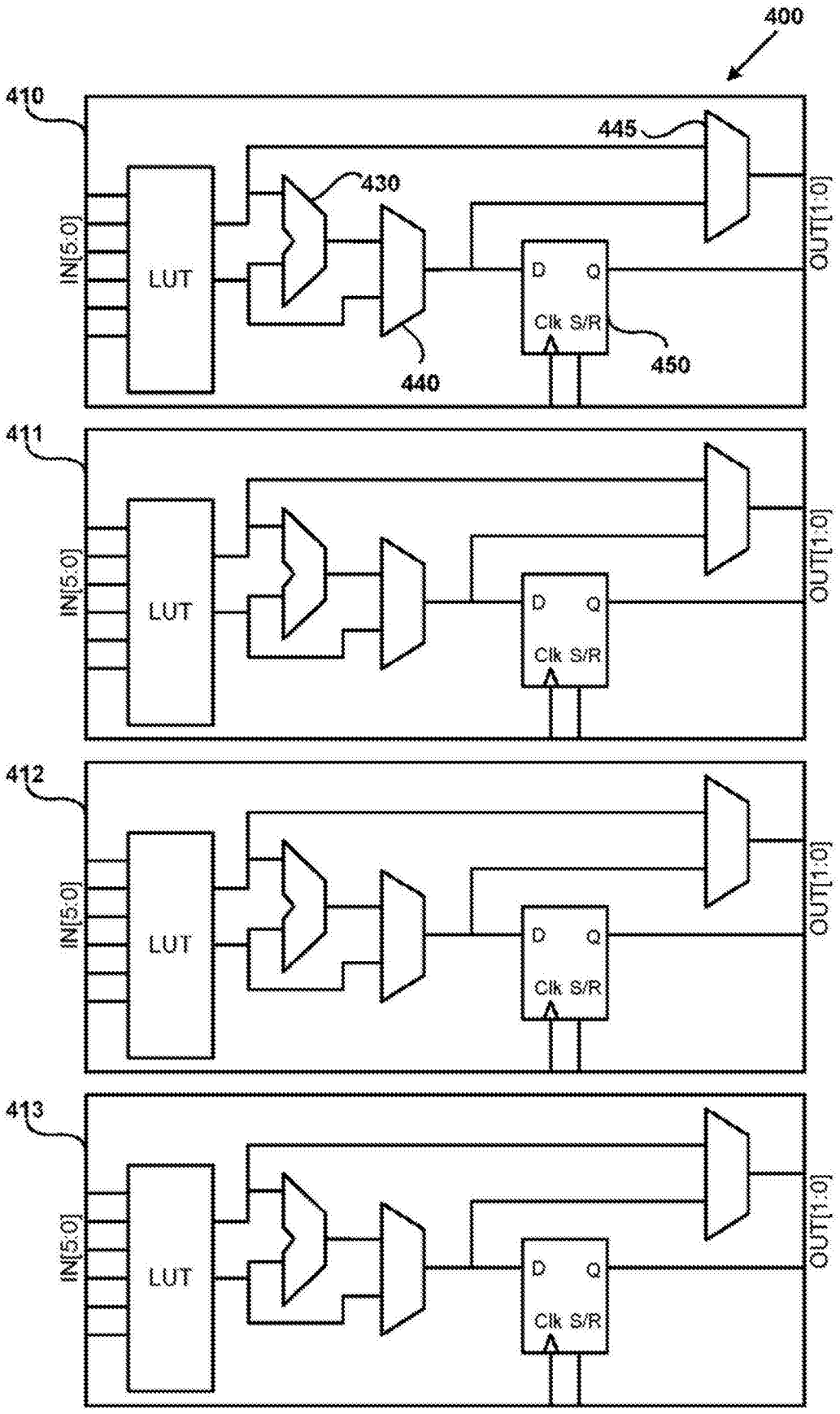


图4

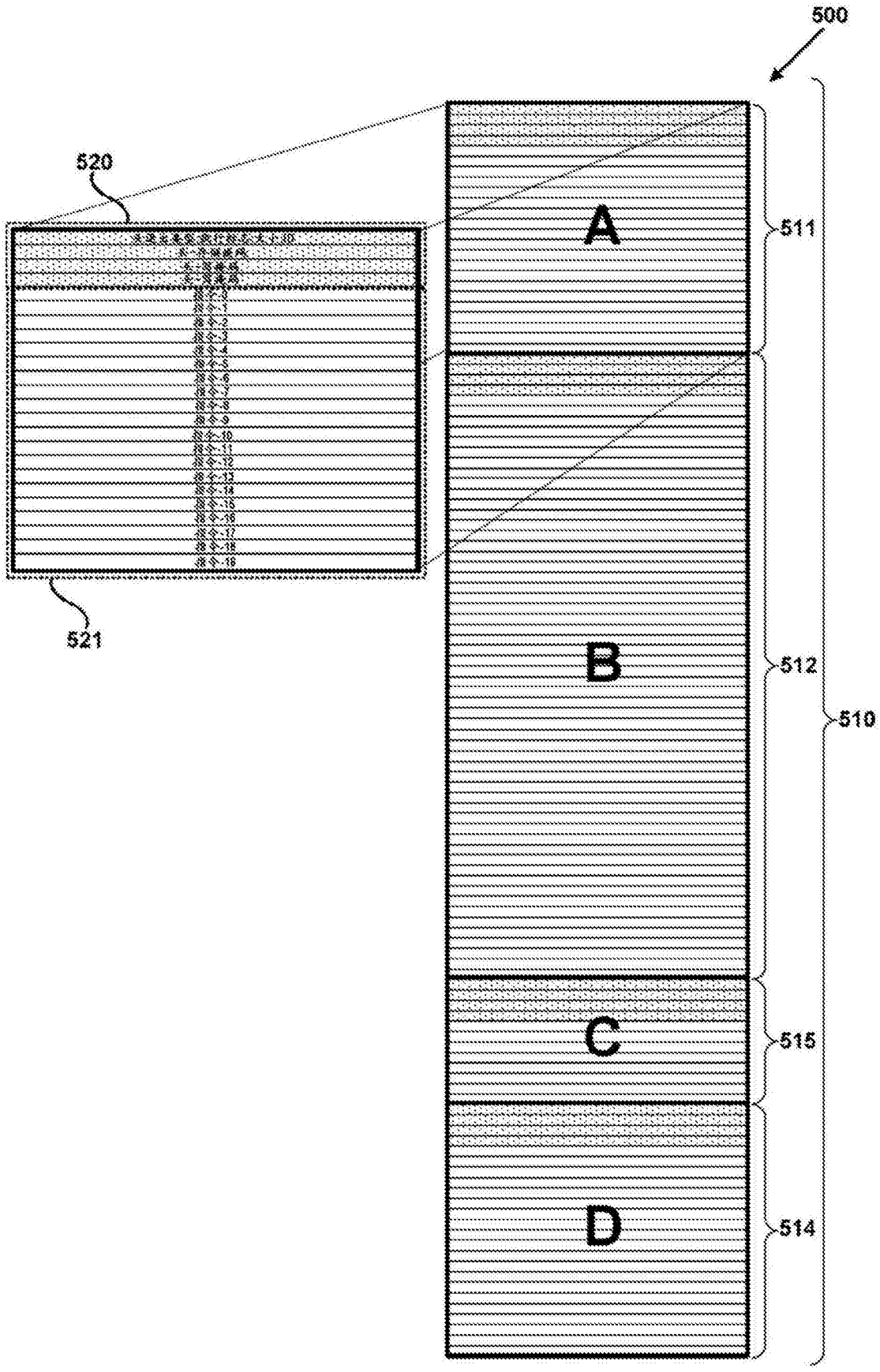


图5

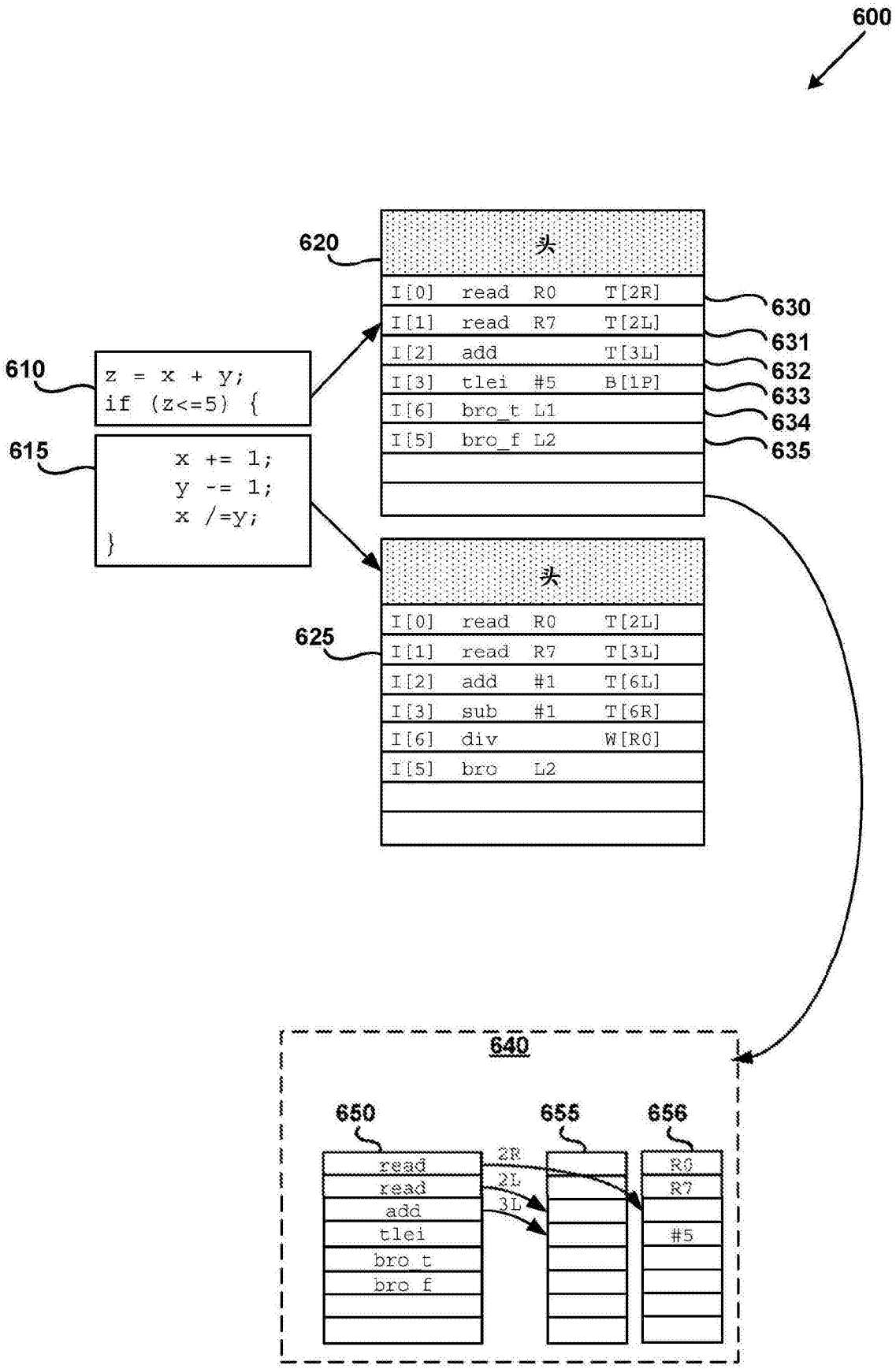

600
620
头
I[0] read R0 T[2R]
I[1] read R7 T[2L]
I[2] add T[3L]
I[3] tlei #5 B[1P]
I[6] bro_t L1
I[5] bro_f L2
630
631
632
633
634
635
610
z = x + y;
if (z<=5) {
615
x += 1;
y -= 1;
x /=y;
}
头
625
I[0] read R0 T[2L]
I[1] read R7 T[3L]
I[2] add #1 T[6L]
I[3] sub #1 T[6R]
I[6] div W[R0]
I[5] bro L2
640
650
655
656
read
read
add
tlei
bro_t
bro_f
2R
2L
3L
R0
R7
#5

图6

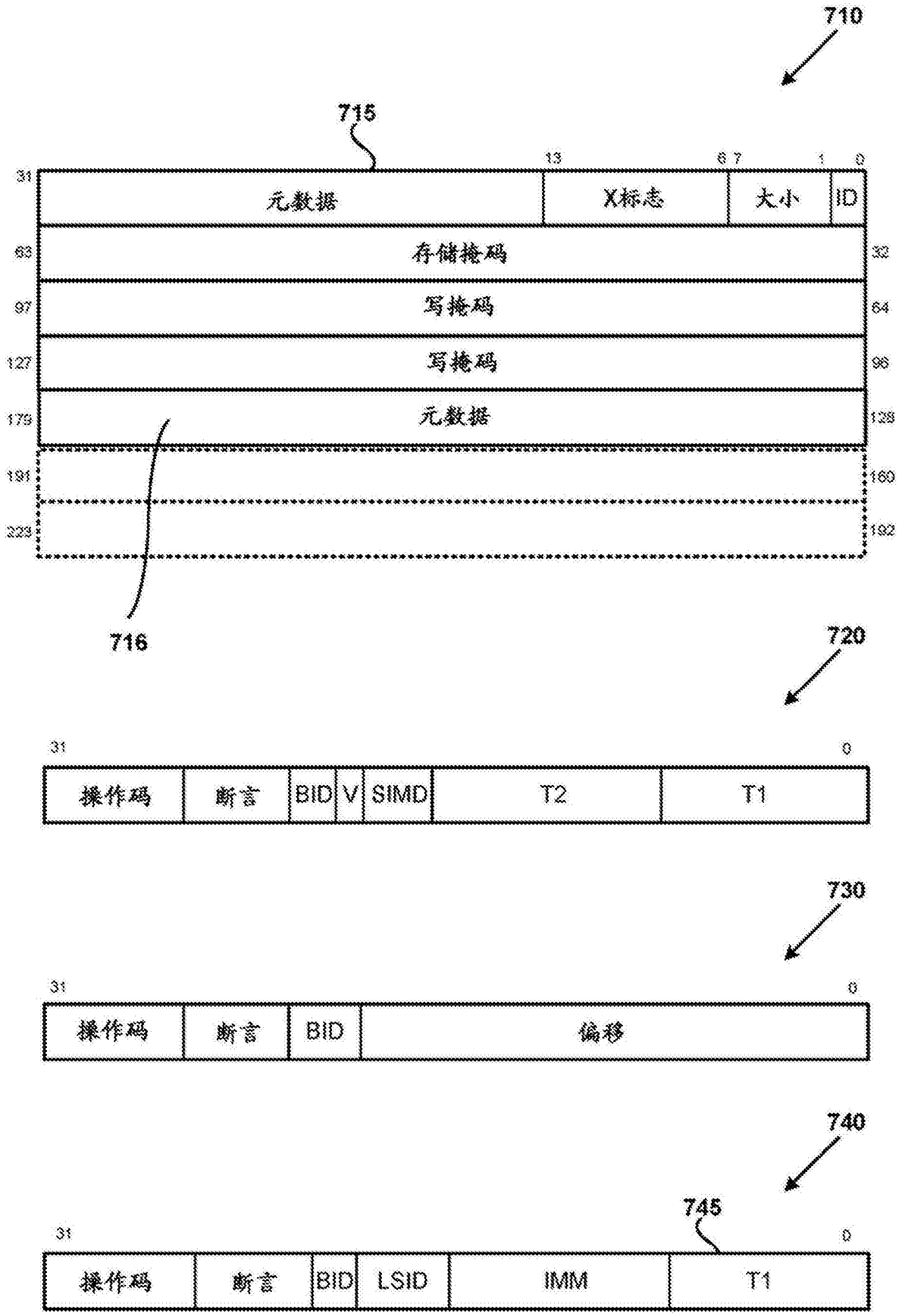

710
715
31
13
6 7
1
0
元数据
X标志
大小
ID
63
32
存储掩码
97
64
写掩码
127
96
写掩码
179
128
元数据
191
160
223
192
716
720
31
0
操作码
断言
BID
V
SIMD
T2
T1
730
31
0
操作码
断言
BID
偏移
740
745
31
0
操作码
断言
BID
LSID
IMM
T1

图7

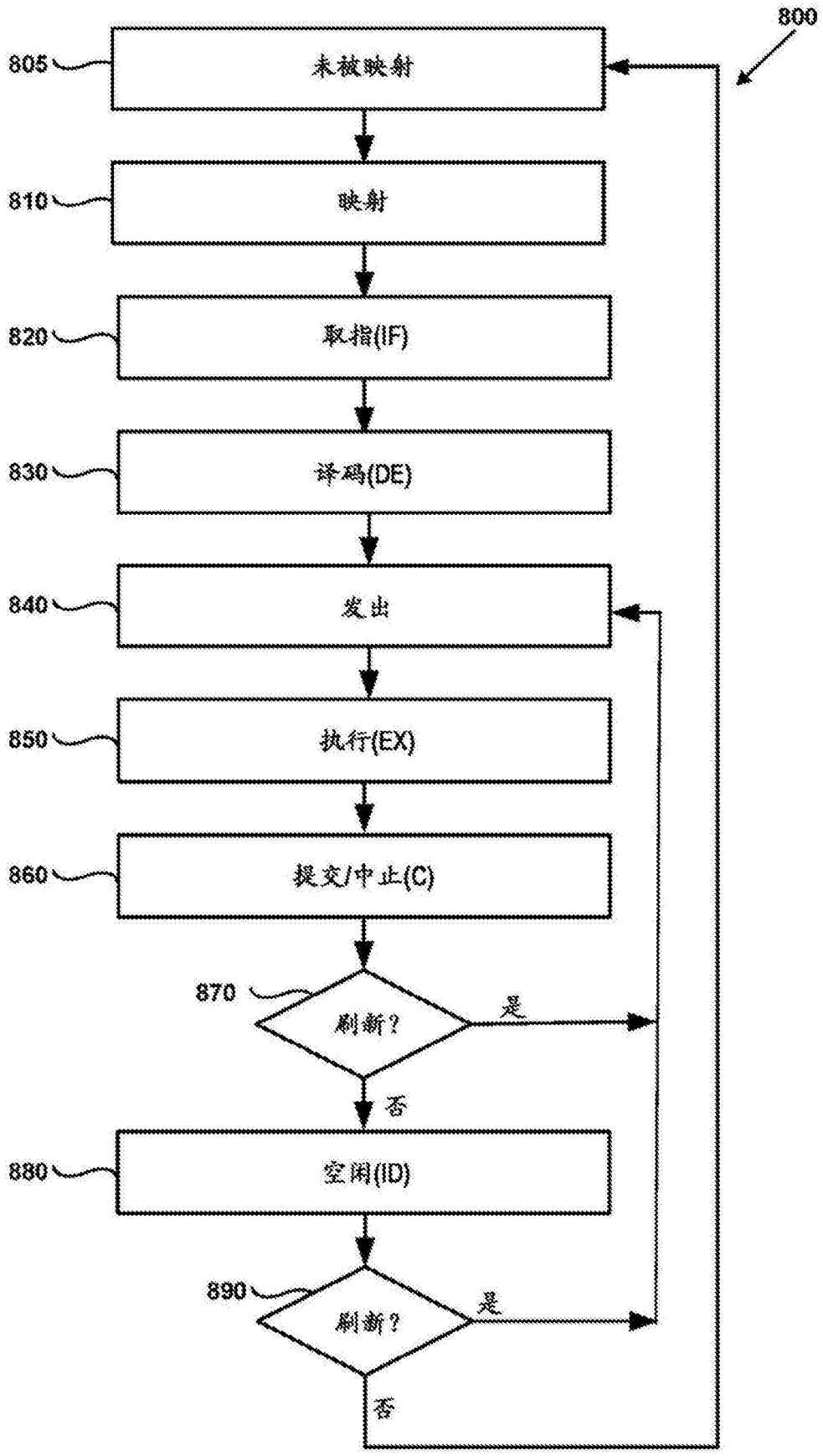

800
805
未被映射
810
映射
820
取指(IF)
830
译码(DE)
840
发出
850
执行(EX)
860
提交/中止(C)
870
刷新?
是
否
880
空闲(ID)
890
刷新?
是
否

图8

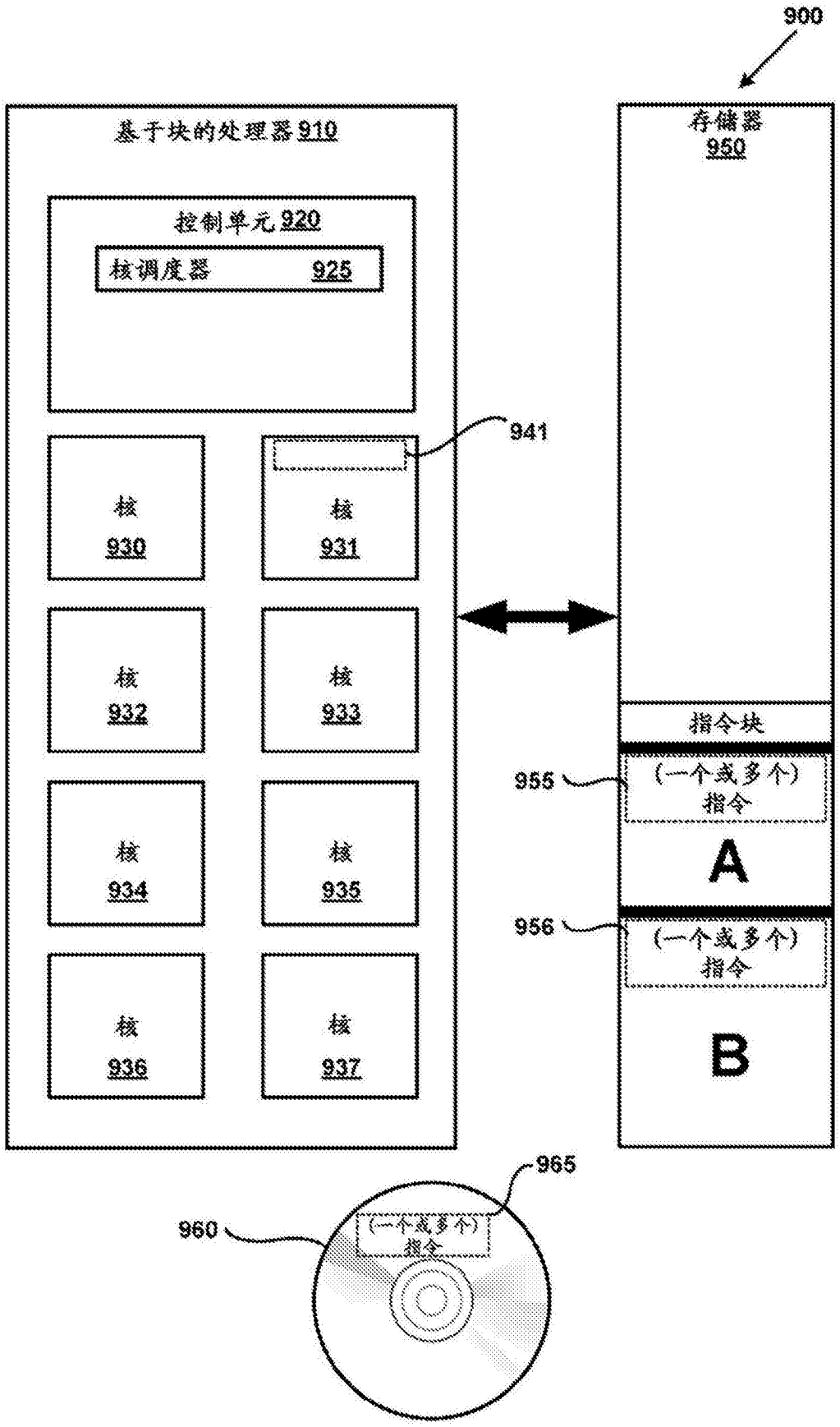

900
基于块的处理器910
控制单元920
核调度器 925
941
核 930
核 931
核 932
核 933
核 934
核 935
核 936
核 937
存储器 950
指令块
955
(一个或多个)指令
A
956
(一个或多个)指令
B
965
960
(一个或多个)指令

图9

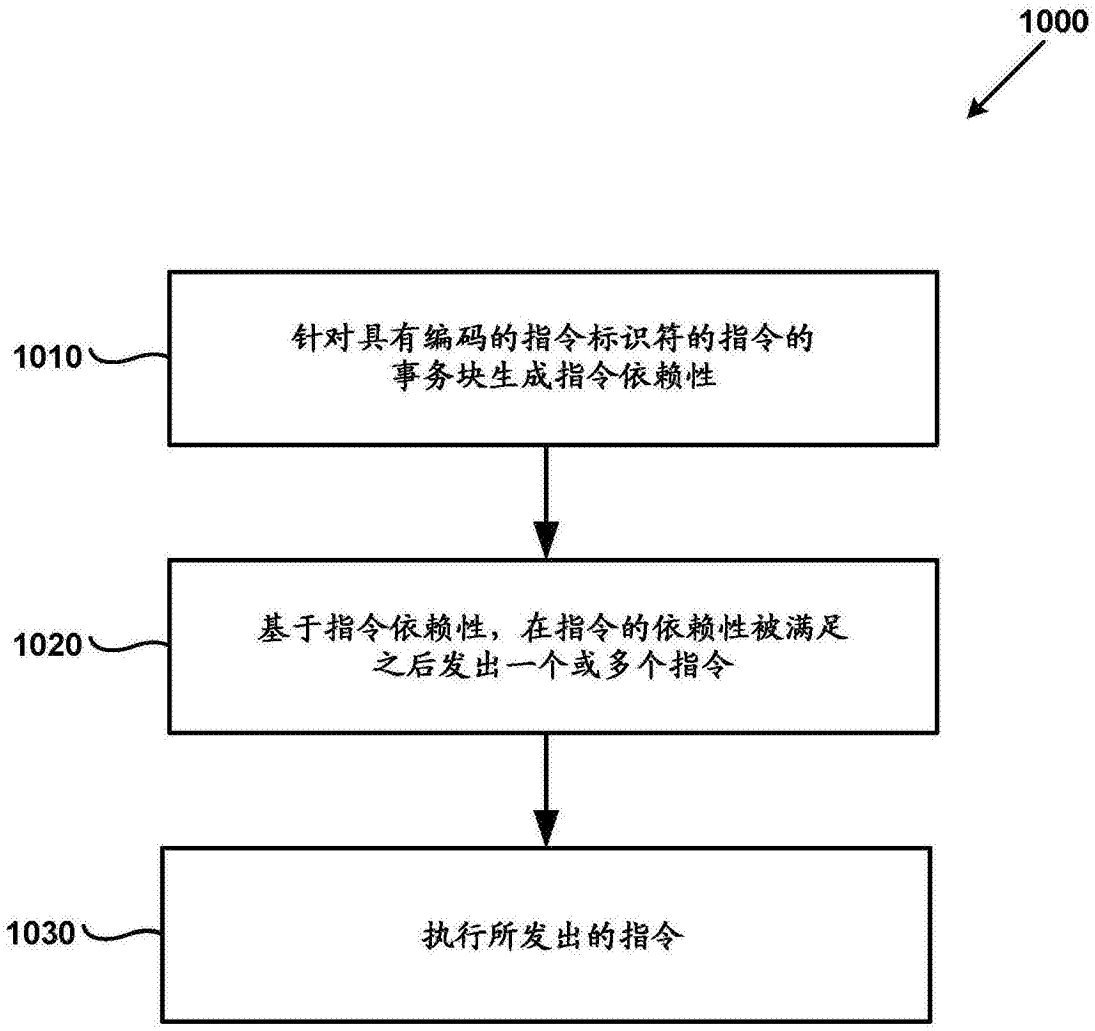

1000
1010
针对具有编码的指令标识符的指令的事务块生成指令依赖性
1020
基于指令依赖性，在指令的依赖性被满足之后发出一个或多个指令
1030
执行所发出的指令

图10

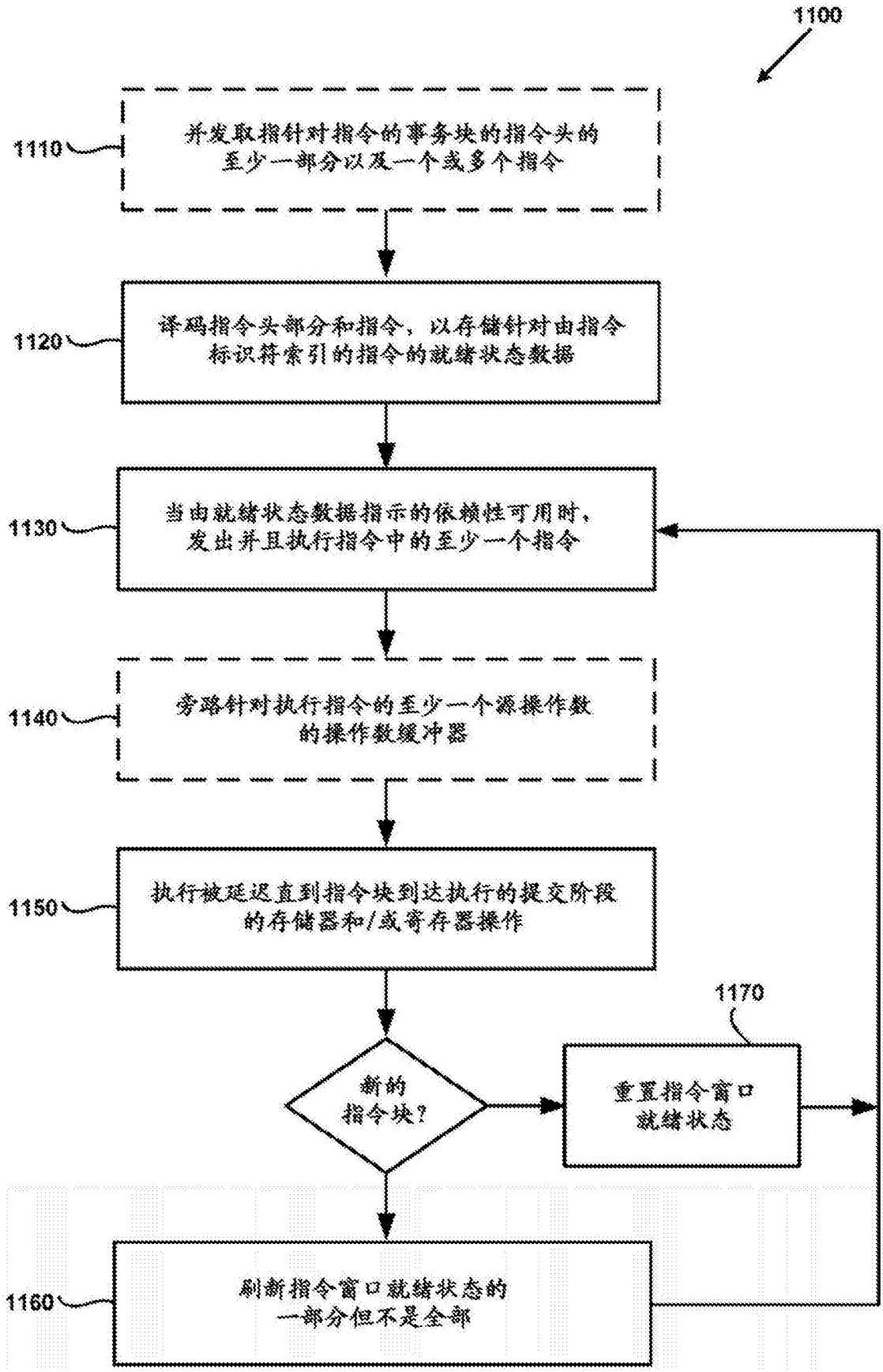

1100
1110
并发取指针对指令的事务块的指令头的至少一部分以及一个或多个指令
1120
译码指令头部分和指令，以存储针对由指令标识符索引的指令的就绪状态数据
1130
当由就绪状态数据指示的依赖性可用时，发出并且执行指令中的至少一个指令
1140
旁路针对执行指令的至少一个源操作数的操作数缓冲器
1150
执行被延迟直到指令块到达执行的提交阶段的存储器和/或寄存器操作
1170
新的指令块？
重置指令窗口就绪状态
1160
刷新指令窗口就绪状态的一部分但不是全部

图11

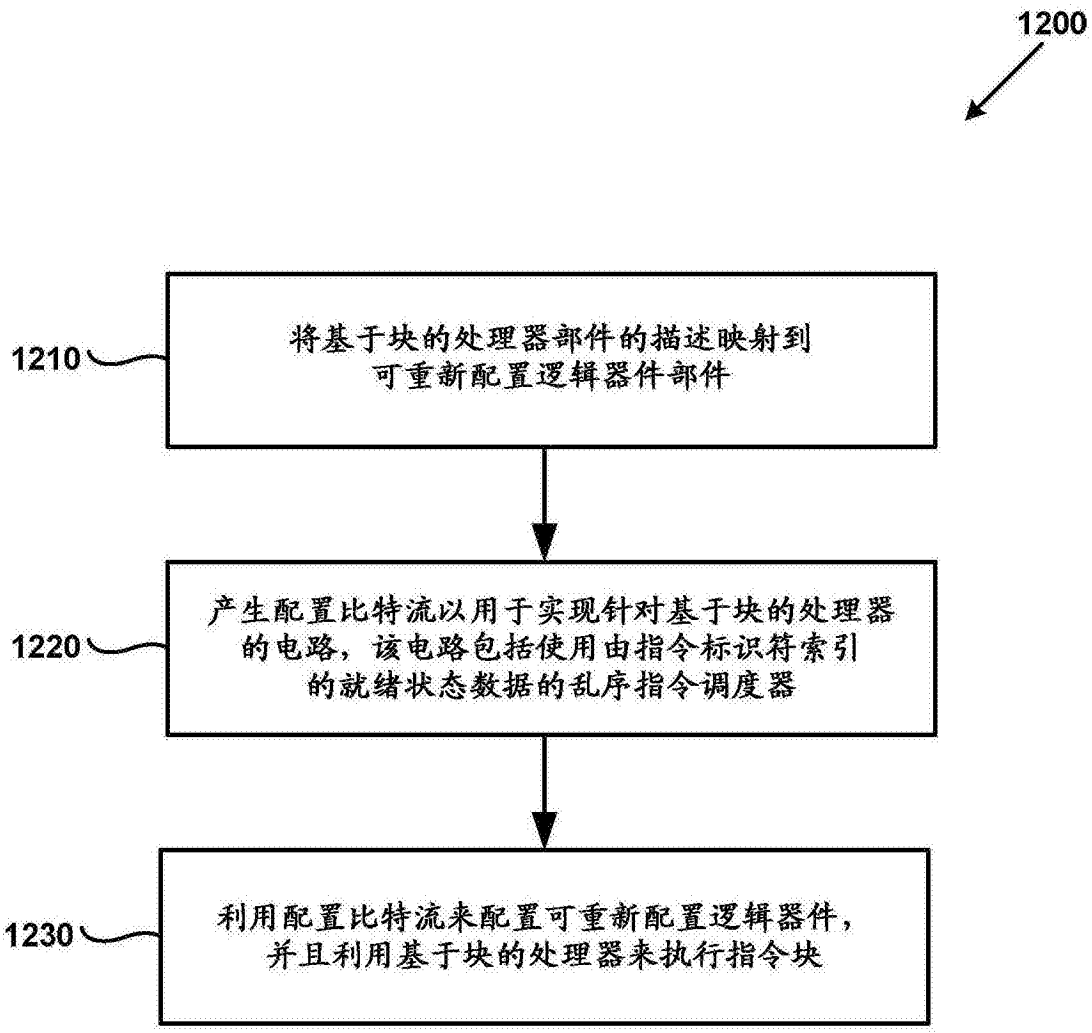

1200
1210
将基于块的处理器部件的描述映射到可重新配置逻辑器件部件
1220
产生配置比特流以用于实现针对基于块的处理器的电路，该电路包括使用由指令标识符索引的就绪状态数据的乱序指令调度器
1230
利用配置比特流来配置可重新配置逻辑器件，并且利用基于块的处理器来执行指令块

图12

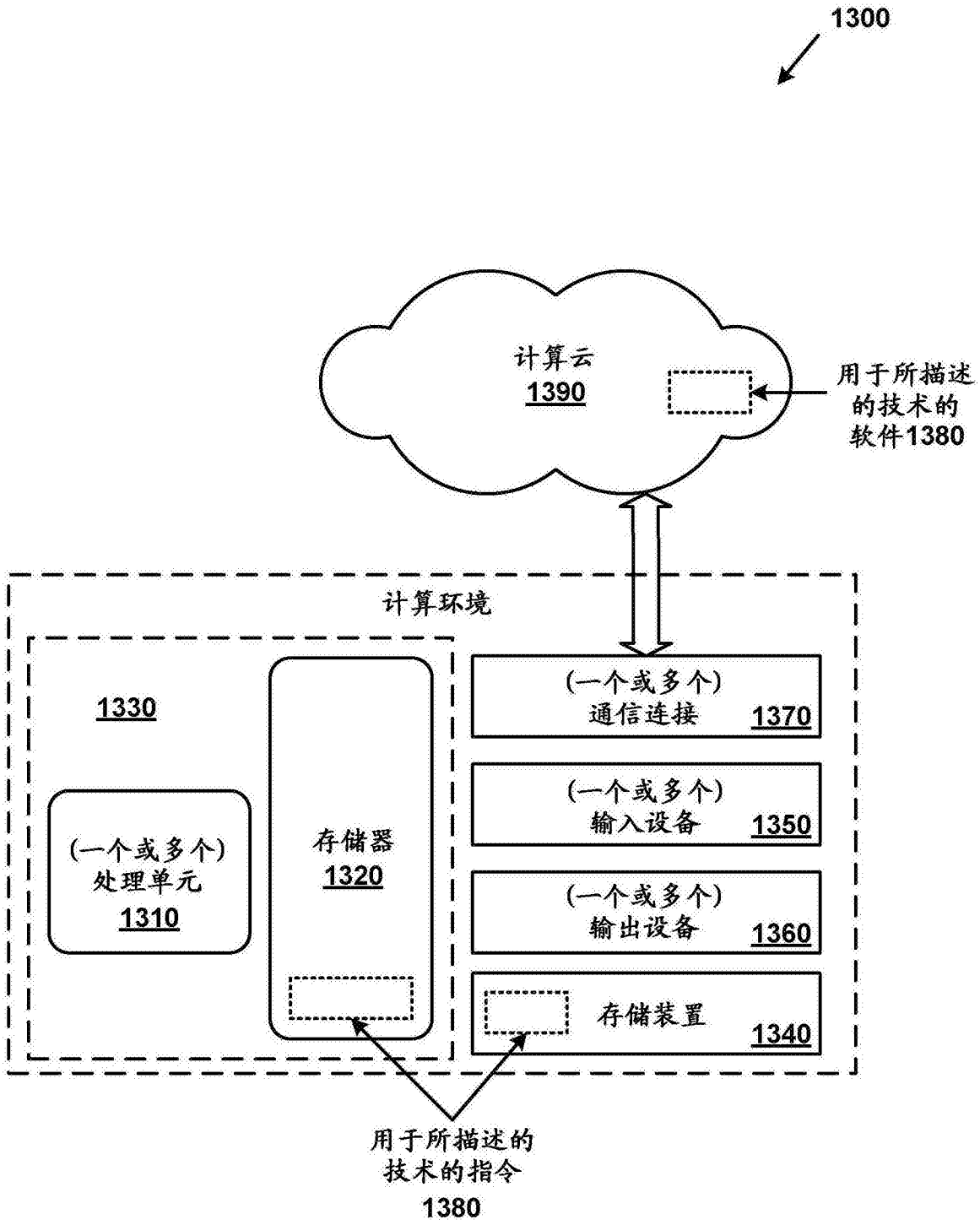

1300
计算云
1390
用于所描述的技术的软件1380
计算环境
1330
(一个或多个)处理单元
1310
存储器
1320
(一个或多个)通信连接
1370
(一个或多个)输入设备
1350
(一个或多个)输出设备
1360
存储装置
1340
用于所描述的技术的指令
1380

图13