(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 110968348 A
(43)申请公布日 2020.04.07

(21)申请号 201910795336.4

(22)申请日 2019.08.27

(30)优先权数据
16/147,254 2018.09.28 US

(71)申请人 英特尔公司
地址 美国加利福尼亚州

(72)发明人 R•萨德 R•凡伦天 B•托尔
C•J•休斯 A•F•海内克
E•乌尔德-阿迈德-瓦尔
M•J•查尼

(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 陈依心 何焜

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

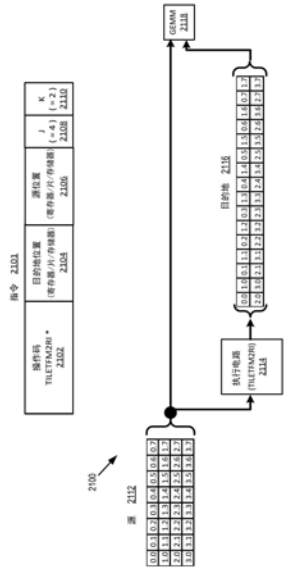
权利要求书2页 说明书38页 附图39页

(54)发明名称

用于执行将矩阵变换为行交错格式的指令的系统和方法

(57)摘要

本申请公开了用于执行将矩阵变换为行交错格式的指令的系统和方法。所公开实施例涉及用于执行用于将矩阵变换为行交错格式的指令的系统和方法。在一个示例中，处理器包括：取出和解码电路，用于取出并解码指令，该指令具有用于指定操作码以及源和目的地矩阵的位置的字段，其中操作码指示处理器用于将所指定的源矩阵变换为具有行交错格式的所指定的目的地矩阵；以及执行电路，用于通过经由以下操作将所指定的源矩阵变换为所指定的RowInt格式化的目的地矩阵来对经解码的指令作出响应：以行为主或列为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵，K宽度子矩阵具有K列和足够的行以保存J个元素。



CN 110968348 A

1.一种处理器,包括:

取出电路,用于取出指令;

解码电路,用于对所述指令进行解码,所述指令具有用于指定操作码、源矩阵的位置和目的地矩阵的位置的字段,所述操作码指示所述处理器用于将所述源矩阵变换为按照行交错(RowInt)格式的所述目的地矩阵;

执行电路,用于按照所述操作码通过以下操作来执行所述指令:使所述源矩阵的每个J元素子列的J个元素交错为所述目的地矩阵的子矩阵,所述子矩阵具有K列和足够的行以供所述子矩阵保存J个元素。

2.如权利要求1所述的处理器,其特征在于,所述源矩阵和所述目的地矩阵中的每一个包括向量寄存器的集合、片寄存器的集合、以及存储器位置的集合中的一个。

3.如权利要求1-2中的任一项所述的处理器,其特征在于,所述指令格式进一步包括用于指定J和K的字段。

4.如权利要求1-3中的任一项所述的处理器,其特征在于,所述指令格式进一步包括用于指定所述源矩阵的元素尺寸和所述目的地矩阵的元素尺寸的字段,所述元素尺寸包括二元数位、半字节、字节、字、双字和四字中的一个。

5.如权利要求1-4中的任一项所述的处理器,其特征在于,所述指令格式进一步包括用于指定所述源矩阵的元素格式和所述目的地矩阵的元素格式的字段,所述元素格式包括紧缩或标量单精度浮点数据、或紧缩或标量双精度浮点数据、以及紧缩或标量整数数据。

6.如权利要求1-5中的任一项所述的处理器,其特征在于,所述源矩阵包括M x N元素数组,并且所述目的地矩阵包括一半的行和两倍的列,其中所述指令的所述格式进一步用于指定M和N中的至少一个。

7.如权利要求1-6中的任一项所述的处理器,其特征在于,所述指令格式进一步包括用于指定掩码的字段,所述掩码是针对每个目的地元素具有一个位的多位的值,所述位用于控制所述目的地元素是否将被更新,或用于控制所述目的地元素将被归零还是合并。

8.一种系统,包括处理器和存储器,所述处理器包括:

取出电路,用于取出指令;

解码电路,用于对所述指令进行解码,所述指令具有用于指定操作码、源矩阵的位置和目的地矩阵的位置的字段,所述操作码指示所述处理器用于将所述源矩阵变换为按照行交错(RowInt)格式的所述目的地矩阵;

执行电路,用于按照所述操作码通过以下操作来执行所述指令:使所述源矩阵的每个J元素子列的J个元素交错为所述目的地矩阵的子矩阵,所述子矩阵具有K列和足够的行以供所述子矩阵保存J个元素。

9.如权利要求8所述的系统,其特征在于,所述源矩阵和所述目的地矩阵中的每一个包括向量寄存器的集合、片寄存器的集合、以及存储器位置的集合中的一个。

10.如权利要求8-9中的任一项所述的系统,其特征在于,所述指令格式进一步包括用于指定J和K的字段。

11.如权利要求8-9中的任一项所述的系统,其特征在于,所述指令格式进一步包括用于指定所述源矩阵的元素尺寸和所述目的地矩阵的元素尺寸的字段,所述元素尺寸包括二元数位、半字节、字节、字、双字和四字中的一个。

12.如权利要求8-9中的任一项所述的系统,其特征在于,所述指令格式进一步包括用于指定所述源矩阵的元素格式和所述目的地矩阵的元素格式的字段,所述元素格式包括紧缩或标量单精度浮点数据、或紧缩或标量双精度浮点数据、以及紧缩或标量整数数据。

13.如权利要求8-9中的任一项所述的系统,其特征在于,所述源矩阵包括M x N元素数组,并且所述目的地矩阵包括一半的行和两倍的列,其中所述指令的所述格式进一步用于指定M和N中的至少一个。

14.如权利要求8-9中的任一项所述的系统,其特征在于,所述指令格式进一步包括用于指定掩码的字段,所述掩码是针对每个目的地元素具有一个位的多位的值,所述位用于控制所述目的地元素是否将被更新,或用于控制所述目的地元素将被归零还是合并。

15.一种由处理器执行的方法,所述方法包括:

使用取出电路取出指令;

使用解码电路对所述指令进行解码,所述指令具有用于指定操作码、源矩阵的位置和目的地矩阵的位置的字段,所述操作码指示所述处理器用于将所述源矩阵变换为按照行交错(RowInt)格式的所述目的地矩阵;

使用解码电路对所取出的指令进行解码;以及

使用执行电路按照所述操作码通过以下操作来执行所述指令:使所述源矩阵的每个J元素子列的J个元素交错为所述目的地矩阵的子矩阵,所述子矩阵具有K列和足够的行以供所述子矩阵保存J个元素。

16.如权利要求15所述的方法,其特征在于,所述源矩阵和所述目的地矩阵中的每一个包括向量寄存器的集合、片寄存器的集合、以及存储器位置的集合中的一个。

17.如权利要求15-16中的任一项所述的方法,其特征在于,J和K被附加的指令操作数中的一个或多个指定,被传递为所述操作码的前缀或后缀,被编码在立即数中,以及被已经由软件编程的控制寄存器指定。

18.如权利要求15-16中的任一项所述的方法,其特征在于,所述指令格式进一步包括用于指定所述源矩阵的元素尺寸和所述目的地矩阵的元素尺寸的字段,所述元素尺寸包括二元数位、半字节、字节、字、双字和四字中的一个。

19.如权利要求15-16中的任一项所述的方法,其特征在于,所述指令格式进一步包括用于指定所述源矩阵的元素格式和所述目的地矩阵的元素格式的字段,所述元素格式包括紧缩或标量单精度浮点数据、或紧缩或标量双精度浮点数据、以及紧缩或标量整数数据。

20.如权利要求15-16中的任一项所述的方法,其特征在于,所述源矩阵包括M x N元素数组,并且所述目的地矩阵包括一半的行和两倍的列,其中所述指令的所述格式进一步用于指定M和N中的至少一个。

21.如权利要求15-16中的任一项所述的方法,其特征在于,所述指令格式进一步包括用于指定掩码的字段,所述掩码是针对每个目的地元素具有一个位的多位的值,所述位用于控制所述目的地元素是否将被更新,或用于控制所述目的地元素将被归零还是合并。

22.一种机器可读介质,包括代码,所述代码当被执行时使机器执行如权利要求15-21中的任一项所述的方法。

用于执行将矩阵变换为行交错格式的指令的系统和方法

技术领域

[0001] 本发明领域总体上涉及计算机处理器架构,并且更具体地涉及用于执行用于将矩阵变换为行交错格式的指令的系统和方法。

背景技术

[0002] 在诸如机器学习和其他批量数据处理之类的许多计算任务中,矩阵正变得日益重要。深度学习是一类机器学习算法。诸如深度神经网络的深度学习架构已经被应用于包括计算机视觉、语音识别、自然语言处理、音频识别、社交网络过滤、机器翻译、生物信息学和药物设计等的领域。

[0003] 用于深度学习的两种工具推理和训练趋向于低精度算术。使深度学习算法和计算的吞吐量最大化可以帮助满足深度学习处理器的需求,深度学习处理器例如在数据中心中执行深度学习的那些处理器。

[0004] 矩阵-矩阵乘法(也称为GEMM或通用矩阵乘法)是对当今处理器的常见的重计算操作。用于矩阵乘法(例如,GEMM)的特殊硬件是用于改善诸如深度学习之类的某些应用的峰值计算(和能效)的好的选择。这些应用中的一些,包括深度学习,可以利用相对少的位对输入数据元素进行操作而不损失准确度,只要输出元素具有足够的位(即,多于输入)。

附图说明

[0005] 在所附附图中以示例方式而非限制方式来图示本发明,在附图中,类似的附图标记指示类似的要素,其中:

[0006] 图1A图示经配置的片的实施例;

[0007] 图1B图示经配置的片的实施例;

[0008] 图2图示矩阵存储的若干示例;

[0009] 图3图示利用矩阵(片)操作加速器的系统的实施例;

[0010] 图4和图5示出如何使用矩阵操作加速器来共享存储器的不同实施例;

[0011] 图6图示使用片的矩阵乘法累加操作(“TMMA”)的实施例;

[0012] 图7图示链式融合乘法累加指令的迭代的执行的子集的实施例;

[0013] 图8图示链式融合乘法累加指令的迭代的执行的子集的实施例;

[0014] 图9图示链式融合乘法累加指令的迭代的执行的子集的实施例;

[0015] 图10图示链式融合乘法累加指令的迭代的执行的子集的实施例;

[0016] 图11图示根据实施例的尺寸为2的幂的SIMD实现方式,其中,累加器使用比至乘法器的输入的尺寸大的输入尺寸;

[0017] 图12图示利用矩阵操作电路的系统的实施例;

[0018] 图13图示处理器核流水线的实施例,该处理器核流水线支持使用片的矩阵操作;

[0019] 图14图示处理器核流水线的实施例,该处理器核流水线支持使用片的矩阵操作;

[0020] 图15图示按行为主格式和列为主格式表达的矩阵的示例;

[0021] 图16图示矩阵(片)的使用的示例；

[0022] 图17图示矩阵(片)的使用的方法的实施例；

[0023] 图18图示根据实施例的对片的使用的配置的支持；

[0024] 图19图示将支持的矩阵(片)的描述的实施例；

[0025] 图20(A)-图20(D)图示(多个)寄存器的示例；

[0026] 图21是图示根据一些实施例的使用TILETFM2RI指令来加速矩阵乘法的框图；

[0027] 图22A图示根据一些实施例的TILETFM2RI指令的示例性执行；

[0028] 图22B图示根据一些实施例的TILETFM2RI指令的示例性执行；

[0029] 图23图示执行用于处理TILETFM2RI指令的流程的处理器的实施例；

[0030] 图24是图示根据一些实施例的TILETFM2RI指令的格式的框图；

[0031] 图25A-图25B是图示根据实施例的通用向量友好指令格式及其指令模板的框图；

[0032] 图25A是图示根据实施例的通用向量友好指令格式及其A类指令模板的框图；

[0033] 图25B是图示根据实施例的通用向量友好指令格式及其B类指令模板的框图；

[0034] 图26A是图示根据实施例的示例性专用向量友好指令格式的框图；

[0035] 图26B是图示根据一个实施例的构成完整操作码字段的具有专用向量友好指令格式的字段的框图；

[0036] 图26C是图示根据一个实施例的构成寄存器索引字段的具有专用向量友好指令格式的字段的框图；

[0037] 图26D是图示根据一个实施例的构成扩充操作字段的具有专用向量友好指令格式的字段的框图；

[0038] 图27是根据一个实施例的寄存器架构的框图；

[0039] 图28A是图示根据实施例的示例性有序流水线以及示例性寄存器重命名的乱序发布/执行流水线两者的框图；

[0040] 图28B是图示根据实施例的要包括在处理器中的有序架构核的示例性实施例和示例性的寄存器重命名的乱序发布/执行架构核两者的框图；

[0041] 图29A-图29B图示更具体的示例性有序核架构的框图，该核将是芯片中的若干逻辑块之一(包括相同类型和/或不同类型的其他核)；

[0042] 图29A是根据实施例的单个处理器核以及它与管芯上互连网络的连接及其第二级(L2)高速缓存的本地子集的框图；

[0043] 图29B是根据实施例的图29A中的处理器核的一部分的展开图；

[0044] 图30是根据实施例的可具有超过一个的核、可具有集成存储器控制器、并且可具有集成图形器件的处理器的框图；

[0045] 图31-图34是示例性计算机架构的框图；

[0046] 图31示出根据本发明的一个实施例的系统的框图；

[0047] 图32是根据本发明的实施例的第一更具体的示例性系统的框图；

[0048] 图33是根据本发明的实施例的第二更具体的示例性系统的框图；

[0049] 图34是根据本发明的实施例的芯片上系统(SoC)的框图；以及

[0050] 图35是根据实施例的对照使用软件指令转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

[0051] 在下列描述中，阐述了众多特定细节。然而，应当理解，实施例可在没有这些特定细节的情况下实施。在其他实例中，未详细示出公知的电路、结构和技术，以免使对本描述的理解模糊。

[0052] 说明书中对“一个实施例”、“实施例”、“示例实施例”等的引用表明所描述的实施例可以包括特定的特征、结构或特性，但是每个实施例可能不一定都包括该特定的特征、结构或特性。此外，此类短语不一定是指同一个实施例。此外，当结合实施例描述特定的特征、结构或特性时，认为结合无论是否被明确描述的其他实施例而影响此类特征、结构或特性是在本领域技术人员的知识范围之内的。

[0053] 在许多主流处理器中，处置矩阵是困难的和/或指令密集性任务。例如，可将矩阵的多行置入多个紧缩数据(例如，SIMD或向量)寄存器中，随后可单独地对矩阵的多行进行操作。例如，取决于数据尺寸，将两个8x2矩阵相加可能要求加载或聚集到四个紧缩数据寄存器中。然后，执行与来自每个矩阵的第一行对应的紧缩数据寄存器的第一加法，并且执行与来自每个矩阵的第二行对应的紧缩数据寄存器的第二加法。随后，将所得到的紧缩数据寄存器往回分散到存储器。尽管对于小矩阵而言，该场景可能是可接受的，但是该场景对于较大矩阵通常是不可接受的。

讨论

[0054] 本文中描述的是用于在诸如中央处理单元(CPU)、图形处理单元(GPU)和加速器之类的计算机硬件中支持矩阵操作的机制。矩阵操作利用表示存储器的一个或多个紧缩区域(诸如，寄存器)的2维(2-D)数据结构。贯穿本说明书，这些2-D数据结构被称为片。注意，矩阵可以比片小(使用少于片的全部)，或可利用多个片(矩阵大于任一片的尺寸)。贯穿本说明书，使用矩阵(片)语言来指示使用影响矩阵的片来执行的操作；矩阵是否大于任一片通常是不相关的。

[0055] 每个片可由不同的操作来作用，这些操作诸如本文中详述的那些操作，包括但不限于：矩阵(片)乘法、片加法、片减法、片对角线、片归零、片变换、片点积、片广播、片行广播、片列广播、片乘法、片乘法和累加、片移动，等等。此外，在未来可以与这些操作一起使用或为了支持非数值应用而使用对诸如使用缩放和/或偏置的操作器的支持，非数值应用例如，OpenCL“本地存储器”、数据压缩/解压缩，等等。本文中还描述了用于执行三元片操作(TILETFM2RI)指令的指令。

[0056] 存储(诸如，(非易失性和易失性的)存储器、寄存器、高速缓存等)的多个部分被布置为具有不同横向尺度和纵向尺度的片。例如，片可具有横向尺度4(例如，矩阵的四行)和纵向尺度8(例如，矩阵的8列)。典型地，横向尺度与元素尺寸(例如，2位、4位、8位、16位、32位、64位、128位等)相关。可支持多种数据类型(单精度浮点、双精度浮点、整数等)。

经配置的片的示例性使用

[0057] 在一些实施例中，可配置片参数。例如，可配置给定的片以提供片选项。示例性片选项包括但不限于：片的行数、片的列数、片是否为有效以及片是否由相等尺寸的片对组成。

[0058] 图1A图示经配置的片的实施例。如图所示，应用存储器102的4kB具有存储于其上的4个1kB的片——片t0 104、片t1 106、片t2 108和片t3 110。在该示例中，这4个片不由对

组成，并且每个片具有以行和列布置的元素。片t0 104和片t1 106具有K行和N列的4字节元素(例如，单精度数据)，其中K＝8，且N＝32。片t2 108和片t3 110具有K行和N/2列的8字节元素(例如，双精度数据)。由于双精度操作数的宽度是单精度操作数的两倍，因此该配置与用于提供片选项的调色板一致，将至少4kB的总存储提供给至少4个名称。在操作中，可使用加载操作和存储操作从存储器加载片以及向存储器存储片。取决于所使用的指令编码方案，可用的应用存储器的量以及可用片的尺寸、数量和配置有所不同。

[0059] 图1B图示经配置的片的实施例。如图所示，应用存储器122的4kB具有存储于其上的2对1kB的片，第一对是片t4L 124和片t4R 126，第二对是片t5L 128和片t5R 130。如图所示，片对被划分为左片和右片。在其他实施例中，片对被划分为偶数片和奇数片。在该示例中，这4个片各自都具有以行和列布置的元素。片t4L 124和片t4R 126具有K行和N列的4字节元素(例如，单精度浮点数据)，其中K＝8，且N＝32。片t5L 128和片t5R 130具有K行和N/2列的8字节元素(例如，双精度浮点数据)。由于双精度操作数的宽度是单精度操作数的两倍，因此该配置与用于提供片选项的调色板一致，将至少4kB的总存储提供给至少2个名称。图1A的四个片使用4个名称，每一个名称对1kB的片命名，而图1B中的2个片对可使用2个名称来指定成对的片。在一些实施例中，片指令接受成对的片的名称作为操作数。在操作中，可使用加载操作和存储操作从存储器加载片以及向存储器存储片。取决于所使用的指令编码方案，可用的应用存储器的量以及可用片的尺寸、数量和配置有所不同。

[0060] 在一些实施例中，片参数是可定义的。例如，“调色板”用于提供片选项。示例性选项包括但不限于：片名称的数量、存储的行中的字节数、片中的行数和列数，等等。例如，片的最大“高度”(行数)可定义为：

[0061] 片最大行＝所构造的存储/(调色板名称的数量*每行的字节数)。

[0062] 由此，可写入应用，使得名称的固定使用将能够利用跨实现方式的不同存储尺寸。

[0063] 使用片配置(“TILECONFIG”)指令完成对片的配置，其中，在所选择的调色板中定义特定的片使用。该声明包括要使用的片名称的数量、每个名称(片)的所请求的行数和列数，并且在一些实施例中包括每个片的所请求的数据类型。在一些实施例中，在TILECONFIG指令的执行期间执行一致性校验，以确定其匹配调色板条目的限制。

示例性片存储类型

[0064] 图2图示矩阵存储的若干示例。在(A)中，片被存储在存储器中。如图所示，每“行”由四个紧缩数据元素组成。为了达到下一“行”，使用跨步值。注意，行可被连续地存储在存储器中。当片存储不映射底层存储器阵列行宽度时，跨步式存储器访问允许对一行以及随后对下一行的访问。

[0065] 从存储器加载片以及向存储器存储片典型地是从应用存储器到紧缩的数据行的跨步式访问。示例性TILELOAD和TILESTORE指令或对于作为加载操作指令中的TILE(片)操作数的应用存储器的其他指令参考在一些实施例中是可重新开始的，以针对每条指令处置(高达)2*行的页错误、未掩码的浮点异常和/或中断。

[0066] 在(B)中，矩阵存储在由多个寄存器组成的片中，这些寄存器诸如，紧缩数据寄存器(单指令多数据(SIMD)或向量寄存器)。在该示例中，片被叠加在三个物理寄存器上。典型地，使用连续的寄存器，然而，情况不必是这样。

[0067] 在(C)中，矩阵被存储在可由在片操作中使用的融合乘法累加(FMA)电路访问的非

寄存器存储中的片中。该存储可在FMA内部，或邻近FMA。此外，在一些实施例中，如下文所讨论，该存储可用于数据元素，而不是用于整个行或整个片。

[0068] 经由CPUID报告TMMA架构的所支持的参数。在一些实施例中，信息列表包括最大高度和最大SIMD尺度。配置TMMA架构要求指定每个片的尺度、每个片的元素尺寸以及调色板标识符。通过执行TILECONFIG指令来完成该配置。

[0069] TILECONFIG指令的成功执行启用后续的TILE操作器。TILERELEASEALL指令清除片配置，并禁用TILE操作(直到下一TILECONFIG指令执行)。在一些实施例中，在使用片的上下文切换中使用XSAVE、XSTORE等。在一些实施例中，在XSAVE中使用2个XCR0位，一个用于TILECONFIG元数据，一个位与实际的片有效载荷数据对应。

[0070] TILECONFIG不仅配置片使用，还设置状态变量，该状态变量指示在片经配置的情况下程序在代码区域中。实现方式可枚举对可与片区域一起使用的其他指令的限制，诸如，没有对现有寄存器组的使用，等等。

[0071] 退出片区域典型地利用TILERELEASEALL指令来完成。该指令不取参数并迅速使所有片无效(指示数据不再需要任何保存或恢复)，并且清除与处于片区域中对应的内部状态。

[0072] 在一些实施例中，片操作将使超出由片配置指定的尺度的任何行和任何列归零。例如，随着每一行被写入，片操作将使超出所配置的列数(将元素的尺寸考虑在内)的数据归零。例如，对于64字节的行以及配置有10行和12列的片，写入FP32元素的操作将以12*4字节向前10行中的每一行写入输出/结果数据，并且使每一行中的其余的4*4字节归零。片操作还对前10个经配置的行之后的任何行完全归零。当使用具有64字节的行的1K的片时，将会有16行，因此，在该示例中，最后6行也将被归零。

[0073] 在一些实施例中，当加载数据时，上下文恢复指令(例如，XRSTOR)强制使超出片的所配置的行的数据将被维持为零。如果没有有效配置，则所有行被归零。对片数据的XRSTOR能够加载超出那些所配置的列的列中的无用信息。XRSTOR对超出所配置的列数进行清除不应当是可能的，因为不存在与片配置相关联的元素宽度。

[0074] 当将整个TILE存储区写入存储器时，上下文保存(例如，XSAVE)暴露整个TILE存储区。如果XRSTOR将无用数据加载到片的最右边部分中，则将由XSAVE保存那个数据。对于超出为每个片指定的数量的行，XSAVE将写入零。

[0075] 在一些实施例中，片指令是可重新开始的。访问存储器的操作允许在页错误之后重新开始。凭借受控制和/或状态寄存器控制的对异常的掩码，处理浮点操作的计算指令也允许未掩码的浮点异常。

[0076] 为了支持在这些事件后重新开始指令，这些指令将信息存储在下文详述的起始寄存器中。

矩阵(片)操作系统

示例性硬件支持

[0077] 图3图示利用矩阵(片)操作加速器的系统的实施例。在该图示中，主机处理器/处理系统301将命令311(例如，矩阵操纵操作，诸如，算术或矩阵操纵操作、或加载和存储操作)传递至矩阵操作加速器307。然而，这以这种方式示出，仅用于讨论的目的。如稍后所详述，该加速器307可以是处理核的部分。典型地，作为片操纵操作器指令的命令311将片称为

寄存器-寄存器(“reg-reg”)或寄存器-存储器(“reg-mem”)格式。诸如TILESTORE、TILELOAD、TILECONFIG等的其他命令不对片执行数据操作。命令可以是供加速器307处置的经解码的指令(例如,微操作)或宏指令。

[0078] 在该示例中,一致性存储器接口303耦合至主机处理器/处理系统301和矩阵操作加速器307,使得它们能够共享存储器。图4和图5示出如何使用矩阵操作加速器来共享存储器的不同实施例。如图4中所示,主机处理器401和矩阵操作加速器电路405共享同一存储器403。图5图示其中主机处理器501和矩阵操作加速器505不共享存储器,但可访问彼此的存储器的实施例。例如,处理器501可访问片存储器507,并照常利用其主机存储器503。类似地,矩阵操作加速器505可访问主机存储器503,但更典型地使用其自身的存储器507。注意,这些存储器可以是不同类型的。

[0079] 在一些实施例中,使用在物理寄存器上的叠加结构来支持片。例如,取决于实现方式,片可以利用16个1024位的寄存器、32个512位的寄存器,等等。在一些实施例中,矩阵操作利用表示存储器的一个或多个紧缩区域(诸如,寄存器)的2维(2-D)数据结构。贯穿本说明书,这些2-D数据结构被称为片或片寄存器。

[0080] 在一些实施例中,矩阵操作加速器307包括耦合至数据缓冲器305的多个FMA 309(在一些实现方式中,这些缓冲器305中的一个或多个被存储在如图所示的网格的FMA中)。数据缓冲器305对从存储器加载的片和/或向存储器存储的片进行缓冲(例如,使用片加载或片存储指令)。数据缓冲器可以是例如多个寄存器。典型地,这些FMA被布置为能够读取和写入片的链式FMA 309的网格。在该示例中,矩阵操作加速器307用于使用片T0、T1和T2来执行矩阵乘法操作。片中的至少一个片被容纳在FMA网格309中。在一些实施例中,操作中的所有片都被存储在FMA网格309中。在其他实施例中,仅子集被存储在FMA网格309中。如图所示,T1被容纳,而T0和T2不被容纳。注意,A、B和C是指这些片的矩阵,这些矩阵可以占据或可以不占据片的整个空间。

[0081] 图6图示使用片的矩阵乘法累加操作(“TMMA”)的实施例。

[0082] 矩阵(片A 601)中的行数与串联的(链式)FMA的数量匹配,这些串联的FMA包括计算的等待时间。实现方式可自由地在更小高度的网格上再循环,但是计算保持相同。

[0083] 源/目的地向量来自N行的片(片C 605),并且FMA的网格611执行N个向量-矩阵操作,从而导致执行片的矩阵乘法的完整指令。片B 603是另一向量源,并将“广播”项提供给每一级中的FMA。

[0084] 在操作中,在一些实施例中,(存储在片B 603中的)矩阵B的元素跨FMA的矩形网格散布。(存储在片A 601中的)矩阵B使其行的元素被变换,以与FMA的矩形网格的列尺度匹配。在网格中的每个FMA处,A和B的元素被相乘,并被加到(来自上方的图中)传入的被加数,并且传出的和被传递至FMA的下一行(或最终输出)。

[0085] 单个步骤的等待时间与K(矩阵B的行高)成比例,并且从属的TMMA典型地(在单片中或跨片)具有足够的源-目的地行以隐藏该等待时间。实现方式还可跨时间步长分割SIMD(紧缩数据元素)尺度M(矩阵A的行高),但是这仅改变K乘以的常数。当程序指定比由TMACC枚举的最大值小的K时,实现方式利用“掩码”或“早出”来自由地实现此。

[0086] 整个TMMA的等待时间与N*K成比例。重复率与N成比例。每条TMMA指令的MAC的数量为N*K*M。

[0087] 图7图示链式融合乘法累加指令的迭代的执行的子集的实施例。具体而言，这图示目的地的一个紧缩数据元素位置的迭代的执行电路。在该实施例中，链式融合乘法累加正对有符号源进行操作，其中，累加器2倍于输入数据的尺寸。

[0088] 第一有符号源(源1 701)和第二有符号源(源2 703)各自都具有四个紧缩数据元素。这些紧缩数据元素中的每一个都存储诸如浮点数据之类的有符号数据。第三有符号源(源3 709)具有两个紧缩数据元素，其中的每一个都存储有符号数据。第一有符号源701的尺寸和第二有符号源703的尺寸是第三有符号源(初始值或先前结果)709的尺寸的一半。例如，第一有符号源701和第二有符号源703可具有32位的紧缩数据元素(例如，单精度浮点)，而第三有符号源709可具有64位的紧缩数据元素(例如，双精度浮点)。

[0089] 在该图示中，仅示出第一有符号源701和第二有符号源703的最高有效的两个紧缩数据元素位置以及第三有符号源709的最高有效的紧缩数据元素位置。当然，还将处理其他紧缩数据元素位置。

[0090] 如图所示，成对地处理紧缩数据元素。例如，使用乘法器电路705将第一有符号源701和第二有符号源703的最高有效的紧缩数据元素位置的数据相乘，并且使用乘法器电路707将来自第一有符号源701和第二有符号源703的次高有效的紧缩数据元素位置的数据相乘。在一些实施例中，这些乘法器电路705和707重新用于其他紧缩数据元素位置。在其他实施例中，使用附加的乘法器电路，使得并行地处理紧缩数据元素。在一些上下文中，使用尺寸为有符号第三源709的尺寸的通道来完成并行执行。使用加法电路711将这些乘法中的每个乘法的结果相加。

[0091] (使用不同的加法器713或同一加法器711)将这些乘法的结果的加法的结果加到来自有符号源3 709的最高有效紧缩数据元素位置的数据。

[0092] 最终，第二加法的结果被存储到有符号目的地715中与来自有符号第三源709的所使用的紧缩数据元素位置对应的紧缩数据元素位置中，或者如果有下一迭代，则该第二加法的结果被继续传递到该下一迭代。在一些实施例中，将写掩码应用于此存储，使得如果对应的写掩码(位)被置位，则存储发生，如果对应的写掩码(位)未被置位，则存储不发生。

[0093] 图8图示链式融合乘法累加指令的迭代的执行的子集的实施例。具体而言，这图示目的地的一个紧缩数据元素位置的迭代的执行电路。在该实施例中，链式融合乘法累加正对有符号源进行操作，其中，累加器2倍于输入数据的尺寸。

[0094] 第一有符号源(源1 801)和第二有符号源(源2 803)各自都具有四个紧缩数据元素。这些紧缩数据元素中的每一个都存储诸如整数数据之类的有符号数据。第三有符号源(源3 809)具有两个紧缩数据元素，其中的每一个都存储有符号数据。第一有符号源801的尺寸和第二有符号源803的尺寸是第三有符号源809的尺寸的一半。例如，第一有符号源801和第二有符号源803可具有32位的紧缩数据元素(例如，单精度浮点)，而第三有符号源809可具有64位的紧缩数据元素(例如，双精度浮点)。

[0095] 在该图示中，仅示出第一有符号源801和第二有符号源803的最高有效的两个紧缩数据元素位置以及第三有符号源809的最高有效的紧缩数据元素位置。当然，还将处理其他紧缩数据元素位置。

[0096] 如图所示，成对地处理紧缩数据元素。例如，使用乘法器电路805将第一有符号源801和第二有符号源803的最高有效的紧缩数据元素位置的数据相乘，并且使用乘法器电路

807将来自第一有符号源801和第二有符号源803的次高有效的紧缩数据元素位置的数据相乘。在一些实施例中，这些乘法器电路805和807重新用于其他紧缩数据元素位置。在其他实施例中，使用附加的乘法器电路，使得并行地处理紧缩数据元素。在一些上下文中，使用尺寸为有符号第三源(初始值或先前迭代结果)809的尺寸的通道来完成并行执行。使用加法/饱和电路813将多个乘法中的每个乘法的结果加到有符号第三源809。

[0097] 当加法导致过大的值时，加法/饱和(累加器)电路813保留操作数的符号。具体而言，对于多路加法与向目的地或下一迭代的写入之间的无限精度结果，饱和评估发生。当累加器813是浮点且输入项是整数时，乘积的和以及浮点累加器输入值被转换为无限精度值(数百位的定点数)，执行乘法结果与第三输入的加法，并执行向实际累加器类型的单次舍入。

[0098] 无符号饱和意味着输出值被限于那个元素宽度的最大无符号数(全1)。有符号饱和意味着值被限于处于那个元素宽度的最小负数与最大正数之间的范围中(例如，对于字节，范围为从-128(=-2^7)到127(=2^7-1))。

[0099] 加法和饱和校验的结果被存储到有符号结果815中与来自有符号第三源809的所使用的紧缩数据元素位置对应的紧缩数据元素位置中，或者如果有下一迭代，则该结果被继续传递到该下一迭代。在一些实施例中，将写掩码应用于此存储，使得如果对应的写掩码(位)被置位，则存储发生，如果对应的写掩码(位)未被置位，则存储不发生。

[0100] 图9图示链式融合乘法累加指令的迭代的执行的子集的实施例。具体而言，这图示目的地的一个紧缩数据元素位置的迭代的执行电路。在该实施例中，链式融合乘法累加正对有符号源和无符号源进行操作，其中，累加器4倍于输入数据的尺寸。

[0101] 第一有符号源(源1 901)和第二无符号源(源2 903)各自都具有四个紧缩数据元素。这些紧缩数据元素中的每一个都具有诸如浮点数据或整数数据之类的数据。第三有符号源(初始值或结果915)具有存储有符号数据的紧缩数据元素。第一源901的尺寸和第二源903的尺寸是第三有符号源915的尺寸的四分之一。例如，第一源901和第二源903可具有16位的紧缩数据元素(例如，字)，而第三有符号源915可具有64位的紧缩数据元素(例如，双精度浮点或64位整数)。

[0102] 在该图示中，仅示出第一源901和第二源903的最高有效的四个紧缩数据元素位置以及第三有符号源915的最高有效的紧缩数据元素位置。当然，如果还有任何其他紧缩数据元素位置，则还将处理这些紧缩数据元素位置。

[0103] 如图所示，按四元组处理紧缩数据元素。例如，使用乘法器电路905将第一源901和第二源903的最高有效的紧缩数据元素位置的数据相乘，使用乘法器电路907将来自第一源901和第二源903的次高有效的紧缩数据元素位置的数据相乘，使用乘法器电路909将来自第一源901和第二源903的第三高有效的紧缩数据元素位置的数据相乘，并且使用乘法器电路911将来自第一源901和第二源903的最低有效的紧缩数据元素位置的数据相乘。在一些实施例中，在乘法之前，对第一源901的有符号紧缩数据元素进行符号扩展，并且对第二源903的无符号紧缩数据元素进行零扩展。

[0104] 在一些实施例中，这些乘法器电路905-911重新用于其他紧缩数据元素位置。在其他实施例中，使用附加的乘法器电路，使得并行地处理紧缩数据元素。在一些上下文中，使用尺寸为有符号第三源915的尺寸的通道来完成并行执行。使用加法电路913将这些乘法中

的每个乘法的结果相加。

[0105] (使用不同的加法器917或同一加法器913)将这些乘法的结果的加法的结果加到来自有符号源3 915的最高有效紧缩数据元素位置的数据。

[0106] 最终,第二加法的结果919被存储到有符号目的地中与来自有符号第三源915的所使用的紧缩数据元素位置对应的紧缩数据元素位置中,或者被传递到下一迭代。在一些实施例中,将写掩码应用于此存储,使得如果对应的写掩码(位)被置位,则存储发生,如果对应的写掩码(位)未被置位,则存储不发生。

[0107] 图10图示链式融合乘法累加指令的迭代的执行的子集的实施例。具体而言,这图示目的地的一个紧缩数据元素位置的迭代的执行电路。在该实施例中,链式融合乘法累加正对有符号源和无符号源进行操作,其中,累加器4倍于输入数据的尺寸。

[0108] 第一有符号源1001和第二无符号源1003各自都具有四个紧缩数据元素。这些紧缩数据元素中的每一个都存储诸如浮点数据或整数数据之类的数据。第三有符号源1015(初始或先前结果)具有存储有符号数据的紧缩数据元素。第一源的尺寸和第二源的尺寸是第三有符号源1015(初始或先前结果)的尺寸的四分之一。例如,第一源和第二源可具有16位的紧缩数据元素(例如,字),而第三有符号源1015(初始或先前结果)可具有64位的紧缩数据元素(例如,双精度浮点或64位整数)。

[0109] 在该图示中,示出第一有符号源1001和第二无符号源1003的最高有效的四个紧缩数据元素位置以及第三有符号源1015的最高有效的紧缩数据元素位置。当然,如果还有任何其他紧缩数据元素位置,则还将处理这些紧缩数据元素位置。

[0110] 如图所示,按四元组处理紧缩数据元素。例如,使用乘法器电路1005将第一有符号源1001和第二无符号源1003的最高有效的紧缩数据元素位置的数据相乘,使用乘法器电路1007将来自第一有符号源1001和第二无符号源1003的次高有效的紧缩数据元素位置的数据相乘,使用乘法器电路1009将来自第一有符号源1001和第二无符号源1003的第三高有效的紧缩数据元素位置的数据相乘,并且使用乘法器电路1011将来自第一有符号源1001和第二无符号源1003的最低有效的紧缩数据元素位置的数据相乘。在一些实施例中,在乘法之前,对第一有符号源1001的有符号紧缩数据元素进行符号扩展,并且对第二无符号源1003的无符号紧缩数据元素进行零扩展。

[0111] 在一些实施例中,这些乘法器电路1005-1011重新用于其他紧缩数据元素位置。在其他实施例中,使用附加的乘法器电路,使得并行地处理紧缩数据元素。在一些上下文中,使用尺寸为第三有符号源1015(初始或先前结果)的尺寸的通道来完成并行执行。使用加法器/饱和1013电路将这些乘法结果的加法的结果加到来自第三有符号源1015(初始或先前结果)的最高有效紧缩数据元素位置的数据。

[0112] 当加法导致对于有符号饱和过大或过小的值时,加法/饱和(累加器)电路1013保留操作数的符号。具体而言,对于多路加法与向目的地的写入之间的无限精度结果,饱和评估发生。当累加器1013是浮点且输入项是整数时,乘积的和以及浮点累加器输入值被转换为无限精度值(数百位的定点数),执行乘法结果与第三输入的加法,并执行向实际累加器类型的单次舍入。

[0113] 加法和饱和校验的结果1019被存储到有符号目的地中与来自第三有符号源1015(初始或先前结果)的所使用的紧缩数据元素位置对应的紧缩数据元素位置中或者被传递

到下一迭代。在一些实施例中，将写掩码应用于此存储，使得如果对应的写掩码(位)被置位，则存储发生，如果对应的写掩码(位)未被置位，则存储不发生。

[0114] 图11图示根据实施例的尺寸为2的幂的SIMD实现方式，其中，累加器使用比至乘法器的输入的尺寸大的输入尺寸。注意，(至乘法器的)源和累加器值可以是有符号值或无符号值。对于具有2X输入尺寸的累加器(换言之，累加器输入值的尺寸是源的紧缩数据元素的尺寸的2倍)，表1101图示不同的配置。对于字节尺寸的源，累加器使用尺寸为16位的字或半精度浮点(HPFP)值。对于字尺寸的源，累加器使用尺寸为32位的32位整数或单精度浮点(SPFP)值。对于SPFP或32位整数尺寸的源，累加器使用尺寸为64位的64位整数或双精度浮点(DPFP)值。

[0115] 对于具有4X输入尺寸的累加器(换言之，累加器输入值的尺寸是源的紧缩数据元素的尺寸的4倍)，表1103图示不同的配置。对于字节尺寸的源，累加器使用尺寸为32位的32位整数或单精度浮点(SPFP)值。在一些实施例中，对于字尺寸的源，累加器使用尺寸为64位的64位整数或双精度浮点(DPFP)值。

[0116] 对于具有8X输入尺寸的累加器(换言之，累加器输入值的尺寸是源的紧缩数据元素的尺寸的8倍)，表1105图示配置。对于字节尺寸的源，累加器使用64位整数。

[0117] 如之前所提示，矩阵操作电路可被包括在核中，或可作为外部加速器。图12图示利用矩阵操作电路的系统的实施例。在该图示中，多个实体与环形互连1245耦合。

[0118] 多个核，核0 1201、核1 1203、核2 1205、以及核N 1207提供非基于片的指令支持。在一些实施例中，矩阵操作电路1251设于核1203中，而在其他实施例中，矩阵操作电路1211和1213是在环形互连1245上可访问的。

[0119] 此外，提供一个或多个存储器控制器1223-1225，以代表核和/或矩阵操作电路来与存储器1233和1231通信。

[0120] 图13图示处理器核流水线的实施例，该处理器核流水线支持使用片的矩阵操作。分支预测和解码电路1303执行对来自存储在指令存储1301中的指令的分支预测、对这些指令的解码和/或分支预测和解码两者。例如，本文中详述的指令可存储在指令存储中。在一些实现方式中，分开的电路用于分支预测，并且在一些实施例中，至少一些指令被解码为一个或多个微操作、微代码进入点、微指令、其他指令或使用微代码1305的其他控制信号。分支预测和解码电路1303可使用各种不同的机制来实现。合适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(PLA)、微代码只读存储器(ROM)等。

[0121] 分支预测和解码电路1303耦合至分配/重命名1307电路，在一些实施例中，该分配/重命名1307电路耦合至调度器电路1309。在一些实施例中，这些电路通过执行以下步骤中的一个或多个来提供寄存器重命名、寄存器分配和/或调度功能：1)将逻辑操作数值重命名为物理操作数值(例如，一些实施例中的寄存器别名表)；2)将状态位和标志分配给经解码的指令；以及3)(例如，在一些实施例中，使用预留站)调度经解码的指令供在指令池外部的执行电路上执行。

[0122] 调度器电路1309表示任意数量的不同调度器，包括预留站、中央指令窗口等。调度器电路1309耦合至(多个)物理寄存器堆1315或包括(多个)物理寄存器堆1315。(多个)物理寄存器堆1315中的每一个表示一个或多个物理寄存器堆，其中不同的物理寄存器堆存储一种或多种不同的数据类型，诸如，标量整数、标量浮点、紧缩整数、紧缩浮点、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)、片,等等。在一个实施例中,(多个)物理寄存器堆1315包括向量寄存器电路、写掩码寄存器电路和标量寄存器电路。这些寄存器电路可提供架构向量寄存器、向量掩码寄存器和通用寄存器。(多个)物理寄存器堆1315被引退电路1317覆盖,以图示可实现寄存器重命名和乱序执行的各种方式(诸如,使用(多个)重排序缓冲器和(多个)引退寄存器堆、使用(多个)未来文件(future file)、(多个)历史缓冲器、(多个)引退寄存器堆、使用寄存器映射和寄存器池,等等)。引退电路1317和(多个)物理寄存器堆1315耦合至执行电路1311。

[0123] 尽管在乱序执行的上下文中描述寄存器重命名,但应当理解,寄存器重命名可在有序架构中被使用。虽然处理器的所图示的实施例也可包括分开的指令和数据高速缓存单元以及共享的L2高速缓存单元,但替代实施例也可具有用于指令和数据两者的单个内部高速缓存,诸如例如,第一级(L1)内部高速缓存、或多个级别的内部高速缓存。在一些实施例中,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。替代地,所有高速缓存都可在核和/或处理器的外部。

[0124] 执行电路1311是一个或多个执行单元的集合,包括标量电路1321、向量/SIMD电路1323和矩阵操作电路1327、以及用于访问高速缓存1313的存储器访问电路1325。执行电路执行各种操作(例如,移位、加法、减法、乘法)并对各种数据类型(例如,标量浮点、紧缩整数、紧缩浮点、向量整数、向量浮点)执行。尽管一些实施例可以包括专用于特定功能或功能集合的数个执行单元,但是其他实施例可仅包括一个执行单元或全都执行所有功能的多个执行单元。标量电路1321执行标量操作,向量/SIMD电路1323执行向量/SIMD操作,并且矩阵操作电路1327执行本文中详述的矩阵(片)操作。

[0125] 作为示例,示例性寄存器重命名的、乱序发布/执行核架构可以如下实现流水线:1)指令取出电路执行取出和长度解码级;2)分支和解码电路1303执行解码级;3)分配/重命名1307电路执行分配级和重命名级;4)调度器电路1309执行调度级;5)(耦合至或被包括在调度器电路1309和分配/重命名1307电路和存储器单元中的)(多个)物理寄存器堆执行寄存器读取/存储器读取级;执行电路1311执行执行级;6)存储器单元和(多个)物理寄存器堆单元执行写回/存储器写入级;7)各个单元可涉及异常处置级;以及8)引退单元和(多个)物理寄存器堆单元执行提交级。

[0126] 核可支持一个或多个指令集(例如,x86指令集(具有与较新版本一起添加的一些扩展);加利福尼亚州桑尼维尔市的MIPS技术公司的MIPS指令集;加利福尼亚州桑尼维尔市的ARM控股公司的ARM指令集(具有诸如NEON等任选附加扩展)),其中包括本文中描述的(多条)指令。在一个实施例中,核1390包括用于支持紧缩数据指令集扩展(例如,AVX1、AVX2)的逻辑,由此允许使用紧缩数据来执行由许多多媒体应用使用的操作。

[0127] 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,各种方式包括时分多线程化、同时多线程化(其中单个物理核为物理核正在同时多线程化的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的诸如英特尔®超线程化技术中的同时多线程化)。

[0128] 图14图示处理器核流水线的实施例,该处理器核流水线支持使用片的矩阵操作。分支预测和解码电路1403执行对来自存储在指令存储1401中的指令的分支预测、对这些指令的解码和/或分支预测和解码两者。例如,本文中详述的指令可存储在指令存储中。在一

些实现方式中，分开的电路用于分支预测，并且在一些实施例中，至少一些指令被解码为一个或多个微操作、微代码进入点、微指令、其他指令或使用微代码1405的其他控制信号。分支预测和解码电路1403可使用各种不同的机制来实现。合适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(PLA)、微代码只读存储器(ROM)等。

[0129] 分支预测和解码电路1403耦合至分配/重命名1407电路，在一些实施例中，该分配/重命名1407电路耦合至调度器电路1409。在一些实施例中，这些电路通过执行以下步骤中的一个或多个来提供寄存器重命名、寄存器分配和/或调度功能：1)将逻辑操作数值重命名为物理操作数值(例如，一些实施例中的寄存器别名表)；2)将状态位和标志分配给经解码的指令；以及3)(例如，在一些实施例中，使用预留站)调度经解码的指令供在指令池外部的执行电路上执行。

[0130] 调度器电路1409表示任意数量的不同调度器，包括预留站、中央指令窗口等。(多个)调度器单元调度器电路1409耦合至(多个)物理寄存器堆1415或包括(多个)物理寄存器堆1415。(多个)物理寄存器堆1415中的每一个表示一个或多个物理寄存器堆，其中不同的物理寄存器堆存储一种或多种不同的数据类型，诸如，标量整数、标量浮点、紧缩整数、紧缩浮点、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)、片，等等。在一个实施例中，(多个)物理寄存器堆1415包括向量寄存器电路、写掩码寄存器电路和标量寄存器电路。这些寄存器电路可提供架构向量寄存器、向量掩码寄存器和通用寄存器。(多个)物理寄存器堆1415被引退电路1417覆盖，以图示可实现寄存器重命名和乱序执行的各种方式(诸如，使用(多个)重排序缓冲器和(多个)引退寄存器堆、使用(多个)未来文件(future file)、(多个)历史缓冲器、(多个)引退寄存器堆、使用寄存器映射和寄存器池，等等)。引退电路1417和(多个)物理寄存器堆1415耦合至执行电路1411。

[0131] 尽管在乱序执行的上下文中描述寄存器重命名，但应当理解，寄存器重命名可在有序架构中被使用。虽然处理器的所图示的实施例也可包括分开的指令和数据高速缓存单元以及共享的L2高速缓存单元，但替代实施例也可具有用于指令和数据两者的单个内部高速缓存，诸如例如，第一级(L1)内部高速缓存、或多个级别的内部高速缓存。在一些实施例中，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。替代地，所有高速缓存都可在核和/或处理器的外部。

[0132] 执行电路1411包括一个或多个执行电路1427的集合以及用于访问高速缓存1413的一个或多个存储器访问电路1425的集合。执行电路1427执行本文中详述的矩阵(片)操作。

[0133] 作为示例，示例性寄存器重命名的、乱序发布/执行核架构可以如下实现流水线：1)指令取出电路执行取出和长度解码级；2)分支和解码电路1403执行解码级；3)分配/重命名1407电路执行分配级和重命名级；4)调度器电路1409执行调度级；5)(耦合至或被包括在调度器电路1409和分配/重命名1407电路和存储器单元中的)(多个)物理寄存器堆执行寄存器读取/存储器读取级；执行电路1411执行执行级；6)存储器单元和(多个)物理寄存器堆单元执行写回/存储器写入级；7)各个单元可涉及异常处置级；以及8)引退单元和(多个)物理寄存器堆单元执行提交级。

[0134] 核可支持一个或多个指令集(例如，x86指令集(具有与较新版本一起添加的一些扩展)；加利福尼亚州桑尼维尔市的MIPS技术公司的MIPS指令集；加利福尼亚州桑尼维尔市

的ARM控股公司的ARM指令集(具有诸如NEON等任选附加扩展)),其中包括本文中描述的(多条)指令。在一个实施例中,核1490包括用于支持紧缩数据指令集扩展(例如,AVX1、AVX2)的逻辑,由此允许使用紧缩数据来执行由许多多媒体应用使用的操作。

[0135] 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,各种方式包括时分多线程化、同时多线程化(其中单个物理核为物理核正在同时多线程化的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的诸如英特尔®超线程化技术中的同时多线程化)。

布局

[0136] 贯穿本说明书,使用行为主的数据布局来表达数据。列为主的用户应当根据项的定向来变换这些项。图15图示按行为主格式和列为主格式表达的矩阵的示例。如图所示,矩阵A是2x3矩阵。当该矩阵按行为主的格式存储时,行的数据元素是连续的。当该矩阵按列为主的格式存储时,列的数据元素是连续的。$A^T*B^T=(BA)^T$是矩阵的公知属性,其中,上标T表示变换。按行为主的数据那样来读取列为主的数据导致看起来像变换矩阵的矩阵。

[0137] 在一些实施例中,在硬件中利用行为主的语义,并且列为主的数据将交换操作数顺序并使结果是矩阵的变换,但是对于从存储器的后续列为主的读取,其是正确的非变换矩阵。

[0138] 例如,如果具有两个要相乘的列为主的矩阵:

a b	g i k	ag+bh ai+bj ak+bl
c d *	h j l =	cg+dh ci+dj ck+dl
e f		eg+fh ei+fj ek+fl
(3x2)	(2x3)	(3x3)

[0139] 输入矩阵将按如下方式被存储在线性存储器中(列为主):

a c e b d f

以及

g h i j k l.

[0140] 以尺度2x3和3x2将那些矩阵读取为行为主的,则它们将表现为:

a c e以及g h

b d f i j

k l

[0141] 交换顺序和矩阵乘法:

g h a c e ag+bh cg+dh eg+fh

i j *b d f= ai+bj ci+dj ei+fj

k l ak+bl ck+dl ek+fl

[0142] 变换矩阵移出,并且随后可按行为主的顺序被存储:

ag+bh cg+dh eg+fh ai+bj ci+dj ei+fj ak+bl ck+dl ek+fl

[0143] 并且在后续的列为主的计算中被使用,其是正确的未变换矩阵:

ag+bh ai+bj ak+bl

cg+dh ci+dj ck+dl

eg+fh ei+fj ek+fl

示例性使用

[0144] 图16图示矩阵(片)的使用的示例。在该示例中,矩阵C1601包括两个片,矩阵A1603包括一个片,并且矩阵B1605包括两个片。该图示出用于计算矩阵乘法的算法的内循环的示例。在该示例中,来自矩阵C1601的两个结果片tmm0和tmm1用于将中间结果累加。当来自矩阵A1603的一个片(tmm2)乘以来自矩阵B1605的两个片时,这个片被重复使用2次。指针用于加载来自箭头所指示方向的新A矩阵(片)和两个新B矩阵(片)。未示出的外循环调整用于C片的指针。

[0145] 如图所示的示例性代码包括片配置指令的使用,并且被执行以配置片使用,加载片,用于处理片的循环,将片存储到存储器,并释放片使用。

[0146] 图17图示矩阵(片)的使用的实施例。在1701处,配置片使用。例如,执行TILECONFIG指令以配置片使用,包括设置每个片的行数和列数。典型地,在1703处,从存储器加载至少一个矩阵(片)。在1705处,使用矩阵(片)来执行至少一个矩阵(片)操作。在1707处,将至少一个矩阵(片)向外存储到存储器,并且在1709处,上下文切换可发生。

示例性配置

片配置硬件支持

[0147] 如上文所讨论,片使用通常需要在使用前进行配置。例如,可能不需要完全使用所有的行和列。在一些实施例中不配置这些行和列不仅节省了功率,而且可使用配置来判定操作是否将生成错误。例如,如果M和L不相同,则(N x M)*(L x N)形式的矩阵乘法通常将不起作用。

[0148] 在使用利用片的矩阵之前,在一些实施例中,将配置片支持。例如,配置每个片有多少行和多少列、将使用的片,等等。TILECONFIG指令是对计算机自身的改进,因为它提供对配置计算机以使用(作为处理器核的部分的、或作为外部设备的)矩阵加速器的支持。具体而言,TILECONFIG指令的执行使得配置从存储器被检取,并被应用于矩阵加速器内的矩阵(片)设置。

片使用配置

[0149] 图18图示根据实施例的对片的使用的配置的支持。存储器1801包含将被支持的矩阵(片)的片描述1803。

[0150] 处理器/核1805的指令执行资源1811将片描述1803的多个方面存储到片配置1817中。片配置1817包括用于详述配置了用于调色板的什么片(每个片中的行数和列数)的调色板表1813以及矩阵支持在使用中的标记。具体而言,指令执行资源1811配置成按片配置1817所指定来使用片。指令执行资源1811还可包括用于指示片使用的机器专用寄存器或配置寄存器。还设置附加的值,诸如,使用中值和开始值。片配置1817利用(多个)寄存器1819来存储片使用和配置信息。

[0151] 图19图示将支持的矩阵(片)的描述的实施例。这是将应STTILECFG指令的执行而被存储的描述。在该示例中,每个字段为字节。在字节[0]中,存储调色板ID 1901。调色板ID用于对调色板表1813进行索引,该调色板表1813如由配置所定义来根据调色板ID存储片中的字节数以及与该ID相关联的片的每行的字节。

[0152] 字节1存储将被存储在“startRow”寄存器1903中的值,并且字节2存储将被存储在

寄存器startP 1905中的值。为了支持在这些事件后重新开始指令，这些指令将信息存储在这些寄存器中。为了支持在诸如上文详述的那些事件之类的中断事件之后重新开始指令，这些指令将信息存储在这些寄存器中。startRow值指示应当被用于重新开始的行。startP值指示当对被使用时用于存储操作的行内的位置，并且在一些实施例中，该startP值指示(对的较低片中的)行的下半部分或(对的较高片中的)行的上半部分。一般而言，不需要行(列)中的该位置。

[0153] 成功地执行矩阵(片)指令将会将startRow和startP两者设置为零，TILECONFIG和STTILECFG是例外。

[0154] 在不重新开始被中断的矩阵(片)指令的任何时刻，使startRow和startP值归零是软件的职责。例如，未掩码的浮点异常处置程序可决定在软件中完成操作，并且将程序计数器值改变为另一指令，通常是下一指令。在这种情况下，在恢复程序之前，软件异常处置程序必须使由操作系统呈现给该软件异常处置程序的异常中的startRow和startP值归零。操作系统随后将使用恢复指令来重新加载那些值。

[0155] 字节3存储片的对的指示(每片1b)1907。

[0156] 字节16-17存储片0的行数1913和列数1915，字节18-19存储片1的行数和列数，以此类推。换言之，每一2字节组指定片的行数和列数。如果2字节的组不用于指定片参数，则它们应当具有值零。为比实现限制或调色板限制更多的片指定片参数导致错误。未配置的片用0行0列被设置为初始状态。

[0157] 最终，存储器中的配置通常以诸如用于若干连续字节的全零之类的结尾描述结束。

示例性片和片配置存储

[0158] 图20(A)-图20(D)图示(多个)寄存器1819的示例。图20(A)图示多个寄存器1819。如图所示，每个片(TMM0 2001...TMMN 2003)具有分开的寄存器，其中每个寄存器存储那个特定片的行尺寸和列尺寸。StartP 2011和StartRow 2013被存储在分开的寄存器中。对一个或多个状态寄存器2015置位(例如，TILES_CONFIGURED＝1)以指示片经配置以供使用。

[0159] 图20(B)图示多个寄存器1819。如图所示，每个片具有用于其行和其列的分开的寄存器。例如，TMM0行配置2021、TMM0列配置2023、StartP 2011和StartRow 2013被存储在分开的寄存器中。对一个或多个状态寄存器2015置位(例如，TILES_CONFIGURED＝1)以指示片经配置以供使用。

[0160] 图20(C)图示单个寄存器1819。如图所示，该寄存器将片配置(每片的行和列)2031、StartP 2011和StartRow 2013存储在作为紧缩数据寄存器的单个寄存器中。对一个或多个状态寄存器2015置位(例如，TILES_CONFIGURED＝1)以指示片经配置以供使用。

[0161] 图20(D)图示多个寄存器1819。如图所示，单个寄存器存储片配置(每片的行和列)2031。StartP和StartRow被存储在分开的寄存器2011和2013中。对一个或多个状态寄存器2015置位(例如，TILES_CONFIGURED＝1)以指示片经配置以供使用。

[0162] 构想了其他组合，诸如，将起始寄存器组合到单个寄存器中，在该单个寄存器中，这些起始寄存器被分开显示，等等。

TILETFM2RI

[0163] 如上文所提及的，用于通用矩阵乘法(被称为GEMM)的特殊硬件是用于改善诸如深

度学习之类的某些应用的峰值计算(和能效)的好的选择。这些应用中的一些,包括深度学习,可以利用相对少的位对输入数据元素进行操作而不损失准确度,只要输出元素具有足够的位(即,多于输入)。

[0164] 因此,所公开的方法和系统执行变换指令TILETFM2RI,其对矩阵(片)——即,二维(2D)数据块——进行变换,并且将其结果写入行交错(RowInt)格式化的矩阵/片/2D寄存器。具体而言,当执行融合乘加(FMA)操作(或其他算术)以执行FMA并且然后将相邻操作的结果组合(加到一起)时,可以使用RowInt格式化的矩阵。例如,可以执行由向量或片FMA指定的所有按元素乘法,并且然后将所有这些按元素乘法与相邻元素对一起加到累加器中,该累加器与输入值的两倍一样大,这在输出中产生与在输入中相同的总位数。TILETFM2RI指令在GEMM上下文中实现此类优化。

[0165] 深度学习训练(以及可能地其他应用)使用一些片作为RowInt格式化和常规格式化两者。因此,如果将片从传统的2D块格式(即,具有正常的行和列)变换为RowInt格式,则应用将被加速。RowInt格式取出按照“正常”格式的矩阵的一组连续行,并且使那些行合并/交错。在GEMM操作的上下文中,最终需要的是将矩阵A的每个行与矩阵B的每个列相乘。如果B按照该RowInt格式,则实际上矩阵B的每个列“更短”。

[0166] 图21是图示根据一些实施例的使用TILETFM2RI指令来加速矩阵乘法的框图。如所示,指令2101包括操作码2102(例如,TILETFM2RI),该操作码2102指示处理器用于将所指定的源矩阵变换为具有行交错(RowInt)格式的所指定的目的地矩阵。具体而言,响应于操作码,处理器用于以行为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵,该K宽度子矩阵具有K列和足够的行以保存J个元素。

[0167] 在此处等于4和2的J 2108和K 2110将以若干方式中的一个或多个被指定:作为对于TILETFM2RI指令的操作数(如此处),作为所指定的操作码的后缀或前缀,作为由指令提供的立即数的一部分(例如,J将由16位立即数的低8位指定,而将K由16位立即数的高8位指定),作为在发布指令之前由软件编程的控制寄存器的一部分(例如,XTILECONFIG),或甚至作为架构默认值。J和K可以各自表示从不受限制的整数值范围中选择的整数。

[0168] 指令2101进一步指定目的地矩阵(片)位置2104和源矩阵(片)位置2106。每个所指定的矩阵位置可以在存储器位置、向量寄存器的集合、以及片寄存器的集合中的任一个中。在此,所指定的源矩阵2112和目的地矩阵2116各自包括三十二(32)个字尺寸的元素。所指定的源矩阵2112包括四行和八列,而所指定的目的地矩阵2116包括2行和16列。如所示,所指定的目的地矩阵2116是对所指定的源矩阵2112的行交错(RowInt)格式的变换。

[0169] 还示出用于执行TILETFM2RI指令的系统2100。系统包括所指定的源矩阵(片)2112、执行电路2114和所指定的目的地矩阵(片)2116。所指定的目的地矩阵(片)2116和按照“正常”形式的所指定的源矩阵2112两者均被路由至矩阵乘法电路(GEMM)2118,从而改善GEMM性能和效率。

[0170] 替代地,用于变换矩阵数据的逊色的方法可能存在,但是不实现执行TILETFM2RI指令的所公开实施例的功率和性能增益。在一些其他方法中,软件可以将数据加载到向量/SIMD寄存器中,使用向量指令来执行变换,将重格式化的数据写入存储器,并且然后将重格式化的数据加载到2D/向量/片寄存器中。但是在向量指令中进行格式转换是缓慢的,需要复杂的软件调整,并且可能需要高速缓存中的更多空间。

[0171] 所公开实施例通过允许软件执行TILETFM2RI指令以对二维(2D)数据矩阵(片)进行变换并且将它们放置到RowInt格式化的2D矩阵(片)寄存器中来对替代方法作出改进。有利地,与受限于使用向量寄存器的方法相比,使用该指令对于软件预期是简单的。

[0172] 在一个实施例中,如下文参考图22B进一步示出和描述的,处理器用于执行TILETFM2RI指令,以对来自一对源矩阵(片)的字进行变换并且将结果存储到一对目的地矩阵(片)中。TILETFM2RI将2D数据片变换为所谓的RowInt格式,并且将它们放置到一个或多个2D(即,片)寄存器中。在一个实施例中,源寄存器对的较低编号片表示片对的前16行,而源寄存器对的较高编号片表示片对的接下来的16行。第一目的地片将包含来自源寄存器对的前16列的数据,而第二目的地片将包含来自源寄存器对的第二组16列的数据。在该实施例中,源片(寄存器块)对表现为上/下(这意味着第一片保存行1-16,而第二片保存其余的行17-32)。目的地片(寄存器块)对表现为左/右(这意味着第一片保存列1-16,而第二片保存其余的列17-32)。

[0173] 有利地,所公开实施例可以处置任何尺寸(例如,1字节、2字节、4字节)的数据元素。这包括子字节尺寸的元素,诸如2位或4位的数据元素。

[0174] 作为进一步的优势,所公开实施例还可以处置不同数量的输入片和输出片,包括单个片、一对(如图21和图22A-B所示出的)、或多于一对。

[0175] 用于该指令的输入数据可以来自一个或多个片寄存器,来自向量寄存器的集合,或来自存储器。指令可以将多个片或向量寄存器指定为源和目的地,并且可以显式地对每一个进行编码。在一些实施例中,另一方面,指令指定第一寄存器(例如,X),而剩余的寄存器是第一寄存器的固定函数(例如,X+1,X+2,等等,或X+2,X+4,等等)。在一些实施例中,TILETFM2RI指令将存储器位置指定为一个或多个源矩阵(片)或目的地矩阵(片)的位置。

[0176] 当对(行比列更多、或反过来列比行更多的)矩形片进行变换时,片架构可具有对行和列的数量的不对称限制。因此,一些实施例需要比输出更多或更少的寄存器来保存输入。

[0177] 在一些实施例中,软件用于将填充(padding)添加到输出片(例如,以使片为正方形),或者从输入片移除填充。所公开实施例允许这样做。在一些实施例中,TILETFM2RI指令使用(例如,在寄存器中、作为立即数、或作为操作码的一部分的)参数,该参数指示要作为填充添加到每个输出行(或列)的元素的数量。所填充的元素的值可以是零、或某个其他预选择的值,或者这可以是(作为寄存器、立即数、或操作码的一部分的)对指令的输入。替代地或附加地,指令可以取得参数,该参数指示要从每个输入行或列移除的元素并指示这些元素处于每个行/列的开头、结尾、还是每个行/列中的一些。

[0178] 参考图21-23进一步示出和描述用于执行TILETFM2RI指令的系统和方法。参考图24-26进一步示出和描述TILETFM2RI指令的格式。

示例性执行

[0179] 图22A图示根据一些实施例的TILETFM2RI指令的示例性执行。如所示,指令2201包括操作码2202(例如,TILETFM2RI),该操作码2202指示处理器用于将所指定的源矩阵变换为具有行交错(RowInt)格式的所指定的目的地矩阵。具体而言,响应于操作码,处理器用于以行为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵,该K宽度子矩阵具有K列和足够的行以保存J个元素。

[0180] 在此处等于4和2的J 2208和K 2210将以若干方式中的一个或多个被指定:作为对于TILETFM2RI指令的操作数(如此处),作为所指定的操作码的后缀或前缀,作为由指令提供的立即数的一部分(例如,J将由16位立即数的低8位指定,而K将由16位立即数的高8位指定),作为在发布指令之前由软件编程的控制寄存器的一部分(例如,XTILECONFIG),或甚至作为架构默认值。J和K可以各自表示从不受限制的整数值范围中选择的整数。

[0181] 指令2201进一步指定目的地矩阵(片)位置2204和源矩阵(片)位置2206。每个所指定的矩阵位置可以在存储器位置、向量寄存器的集合、以及片寄存器的集合中的任一个中。在此,所指定的源矩阵2212和目的地矩阵2216各自包括三十二(32)个字尺寸的元素。所指定的源矩阵2212包括四行和八列,而所指定的目的地矩阵2216包括2行和16列。如所示,所指定的目的地矩阵2216是所指定的源矩阵2212的行交错(RowInt)格式的变换。

[0182] 还示出用于执行TILETFM2RI指令的系统2200。系统包括所指定的源矩阵(片)2212、执行电路2214和所指定的目的地矩阵(片)2216。

[0183] 在操作中,取出和解码电路(未示出)用于取出并解码TILETFM2RI指令2201。执行电路2216用于通过经由以下操作将所指定的源矩阵变换为所指定的RowInt格式化的目的地矩阵来对经解码的变换指令作出响应:以行为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵,该K宽度子矩阵具有K列和足够的行以保存J个元素。参考图22B-23、图28A-B和图29A-B进一步示出和描述TILETFM2RI指令的执行。参考图24-26进一步示出和描述TILETFM2RI指令的格式。

[0184] 图22B图示根据一些实施例的TILETFM2RI指令的示例性执行。如所示,指令2252的格式包括用于指定操作码2254(例如,TILETFM2RI)、以及目的地矩阵(片)位置2256、第二目的地矩阵(片)位置2258、源矩阵(片)位置2260和第二源矩阵(片)位置2262的字段。

[0185] 指令2252进一步指定J 2264和K 2266,它们在此处分别等于4和2。然而,应当注意,可以以若干其他方式中的一个或多个来指定J和K:作为对TILETFM2RI指令的操作数,作为所指定的操作码的后缀或前缀,作为由指令提供的立即数的一部分(例如,J将由16位立即数的低8位指定,而将K由16位立即数的高8位指定),作为在发布指令之前由软件编程的控制寄存器的一部分(例如,XTILECONFIG),或甚至作为架构默认值。J和K可以各自表示从不受限制的整数值范围中选择的整数。

[0186] 还示出用于执行TILETFM2RI指令2252的系统2250。系统包括所指定的第一源矩阵(片)2268和所指定的第二源矩阵(片)2275。系统2250进一步包括执行电路2272和所指定的第一和第二目的地矩阵(片)2274和2276。

[0187] 在操作中,处理器用于执行TILETFM2RI指令2252以:对来自一对源矩阵(片)的字进行变换,并且将结果存储到一对目的地矩阵(片)中。TILETFM2RI将所指定的源1 2268矩阵(片)和所指定的源2 2275矩阵(片)变换为行交错(RowInt)格式,并且将它们放置到所指定的目的地1 2274和目的地2 2276中。在一个实施例中,如所示,源片对的较低编号片表示片对的前16行,而源片对的较高编号片表示片对的接下来的16行。第一目的地片用于包含来自源片对的前16列的数据,而第二目的地片用于包含来自源片对的第二组16列的数据。在该实施例中,源片(寄存器块)对表现为上/下(这意味着第一片保存行1-16,而第二片保存其余的行17-32)。目的地片(寄存器块)对表现为左/右(这意味着第一片保存列1-16,而第二片保存其余的列17-32)。在其他实施例中,目的地片(寄存器块)对表现为上/下。

[0188] 在操作中,取出和解码电路(未示出)用于取出并解码TILETFM2RI指令2252。执行电路2272用于通过经由以下操作将所指定的源矩阵变换为所指定的RowInt格式化的目的地矩阵来对经解码的矩阵变换指令作出响应:以行为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵,该K宽度子矩阵具有K列和足够的行以保存J个元素。在其他实施例中,J个元素将以列为主的顺序被交错到K宽度子矩阵中。

[0189] 参考图21、图23、图28A-B和图29A-B进一步示出和描述TILETFM2RI指令的执行。参考图24-26进一步示出和描述TILETFM2RI指令的格式。

执行的(多个)示例性方法

[0190] 图23图示执行用于处理TILETFM2RI指令的流程2300的处理器的实施例。如所示,在2301处,处理器用于使用取出电路来取出具有格式的指令,该格式具有用于指定操作码以及源矩阵和目的地矩阵的位置的字段,其中操作码指示处理器用于将所指定的源矩阵变换为具有行交错(RowInt)格式的所指定的目的地矩阵。具体而言,响应于操作码,处理器用于以行为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵,该K宽度子矩阵具有K列和足够的行以保存J个元素。

[0191] J和K将以若干方式中的一个或多个被指定:作为对TILETFM2RI指令的操作数,作为所指定的操作码的后缀或前缀,作为由指令提供的立即数的一部分(例如,J将由16位立即数的低8位指定,而K将由16位立即数的高8位指定),作为在发布指令之前由软件编程的控制寄存器的一部分(例如,XTILECONFIG),或甚至作为架构默认值。J和K可以各自表示从不受限制的整数值范围中选择的整数。

[0192] 在2303处,处理器用于使用解码电路对所取出的指令进行解码。例如,由诸如本文中详述的解码电路对所取出的TILETFM2RI指令进行解码。在所示出的系统的上下文中,解码电路类似于至少参考图13、图14和图28A-B所示出和描述的解码电路。

[0193] 在2305处,(根据需要)调度经解码的指令的执行,这是可选的(如其虚线边框所指示),这体现在其可以在不同的时间发生,或者根本不发生。在2307处,处理器用于使用执行电路,通过经由以下操作将所指定的源矩阵变换为所指定的RowInt格式化的目的地矩阵来对经解码的指令作出响应:以行为主为主的顺序或列为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵,该K宽度子矩阵具有K列和足够的行以保存J个元素。在其他实施例中,J个元素将以列为主的顺序被交错到K宽度子矩阵中。

[0194] 参考图3-14进一步示出和描述执行电路。在一些实施例中,执行电路是矩阵操作加速器,诸如示出和描述为加速器307(图3)的加速器。在一些实施例中,执行电路是矩阵操作电路,诸如,矩阵操作电路405(图4)、505(图5)、1213(图12)和1327(图13)。

[0195] 在一些实施例中,在2309处,提交或引退该指令,这是可选的(如其虚线边框所指示),这体现在其可以在不同的时间发生,或者根本不发生。

(多个)示例性指令格式

[0196] 图24是图示根据一些实施例的TILETFM2RI指令的格式的框图。如所示,TILETFM2RI指令2400包括用于指定操作码2402的字段,该操作码2402指示处理器用于将所指定的源矩阵变换为具有行交错(RowInt)格式的所指定的目的地矩阵。操作码2402的示例

为TILETFM2RI*。

[0197] 指令2400进一步包括目的地矩阵(片)位置2404和源矩阵(片)位置2406、以及可选的第二目的地矩阵(片)位置2408、以及可选的第二源矩阵(片)位置2410。所指定的源和目的地矩阵位置中的每一个可以在存储器位置、向量寄存器的集合、以及片寄存器的集合中的任一个中。第二源矩阵(片)位置2410和第二目的地矩阵(片)位置2408是可选的,如其虚线边框所指示,这体现在TILETFM2RI指令可以仅指定单个源和目的地,如在图21中,或者可以指定两个源和两个目的地,如在图22B中。在其他实施例(未示出)中,TILETFM2RI可以指定三个或更多个源和三个或更多个目的地。

[0198] TILETFM2RI指令2400进一步包括若干可选参数以控制处理器的行为,这些可选参数包括J 2412、K 2414、元素尺寸2416(二元数位、半字节、字节、字、双字或四字)、元素格式2418(紧缩或标量单精度或双精度浮点数据和紧缩或标量整数数据)、M 2420(源行)、N 2422(源列)、和掩码2424(针对每个目的地元素具有一个位的多位的值,该位用于控制目的地元素是否将被更新,或用于控制该目的地元素将被归零还是合并)。

[0199] 操作码2402被示出为包括星号,其用于传达附加的前缀和/或后缀可以被添加以指定指令行为。指令修饰符2412、2414、2416、2418、2420、2422和2424中的一个或多个可以通过使用操作码2402的前缀或后缀来指定。

[0200] 在一些实施例中,可选的指令修饰符2412、2414、2416、2418、2420、2422和2424中的一个或多个被编码在任选地被包括在指令2400中的立即数字段(未示出)中。在一些实施例中,可选的指令修饰符2412、2414、2416、2418、2420、2422和2424中的一个或多个经由配置/状态寄存器(例如,XTILECONFIG)来指定。换言之,当可选的修饰符2412、2414、2416、2418、2420、2422和2424中的任何一个或多个未被指令指定时,有时它们使用从片架构的其他部分继承的隐式参数。

详细的示例性系统、处理器和仿真

[0201] 本文详述了用于执行上文描述的指令的硬件、软件等的示例。例如,下文所描述的内容详述了指令执行的多个方面,包括诸如取出、解码、调度、执行、引退等之类的各种流水线级。

指令集

[0202] 指令集可包括一种或多种指令格式。给定的指令格式可定义各种字段(例如,位的数量、位的位置)以指定将要执行的操作(例如,操作码)以及将对其执行该操作的(多个)操作数和/或(多个)其他数据字段(例如,掩码),等等。通过指令模板(或子格式)的定义来进一步分解一些指令格式。例如,可将给定指令格式的指令模板定义为具有该指令格式的字段(所包括的字段通常按照相同顺序,但是至少一些字段具有不同的位的位置,因为较少的字段被包括)的不同子集,和/或定义为具有以不同方式进行解释的给定字段。由此,ISA的每一条指令使用给定的指令格式(并且如果经定义,则按照该指令格式的指令模板中的给定的一个指令模板)来表达,并包括用于指定操作和操作数的字段。例如,示例性ADD(加法)指令具有特定的操作码和指令格式,该特定的指令格式包括用于指定该操作码的操作码字段和用于选择操作数(源1/目的地以及源2)的操作数字段;并且该ADD指令在指令流中出现将使得在操作数字段中具有选择特定操作数的特定的内容。已经推出和/或发布了被称为高级向量扩展(AVX)(AVX1和AVX2)和利用向量扩展(VEX)编码方案的SIMD扩展集(参见例如

2014年9月的英特尔®64和IA-32架构软件开发者手册;并且参见2014年10月的英特尔®高级向量扩展编程参考)。

示例性指令格式

[0203] 本文中所描述的(多条)指令的实施例能以不同的格式体现。另外,在下文中详述示例性系统、架构和流水线。(多条)指令的实施例可在此类系统、架构和流水线上执行,但是不限于详述的那些系统、架构和流水线。

[0204] 通用向量友好指令格式

[0205] 向量友好指令格式是适于向量指令(例如,存在专用于向量操作的特定字段)的指令格式。尽管描述了其中通过向量友好指令格式支持向量和标量操作两者的实施例,但是替代实施例仅使用通过向量友好指令格式的向量操作。

[0206] 图25A-图25B是图示根据实施例的通用向量友好指令格式及其指令模板的框图。图25A是图示根据实施例的通用向量友好指令格式及其A类指令模板的框图;而图25B是图示根据实施例的通用向量友好指令格式及其B类指令模板的框图。具体地,针对通用向量友好指令格式2500定义A类和B类指令模板,这两者都包括无存储器访问2505的指令模板和存储器访问2520的指令模板。在向量友好指令格式的上下文中的术语"通用"是指不束缚于任何特定指令集的指令格式。

[0207] 尽管将描述其中向量友好指令格式支持以下情况的实施例:64字节向量操作数长度(或尺寸)与32位(4字节)或64位(8字节)数据元素宽度(或尺寸)(并且由此,64字节向量由16个双字尺寸的元素组成,或者替代地由8个四字尺寸的元素组成);64字节向量操作数长度(或尺寸)与16位(2字节)或8位(1字节)数据元素宽度(或尺寸);32字节向量操作数长度(或尺寸)与32位(4字节)、64位(8字节)、16位(2字节)或8位(1字节)数据元素宽度(或尺寸);以及16字节向量操作数长度(或尺寸)与32位(4字节)、64位(8字节)、16位(2字节)、或8位(1字节)数据元素宽度(或尺寸);但是替代实施例可支持更大、更小和/或不同的向量操作数尺寸(例如,256字节向量操作数)与更大、更小或不同的数据元素宽度(例如,128位(16字节)数据元素宽度)。

[0208] 图25A中的A类指令模板包括:1)在无存储器访问2505的指令模板内,示出无存储器访问的完全舍入控制型操作2510的指令模板、以及无存储器访问的数据变换型操作2515的指令模板;以及2)在存储器访问2520的指令模板内,示出存储器访问的时效性2525的指令模板和存储器访问的非时效性2530的指令模板。图25B中的B类指令模板包括:1)在无存储器访问2505的指令模板内,示出无存储器访问的写掩码控制的部分舍入控制型操作2512的指令模板以及无存储器访问的写掩码控制的vsize型操作2517的指令模板;以及2)在存储器访问2520的指令模板内,示出存储器访问的写掩码控制2527的指令模板。

[0209] 通用向量友好指令格式2500包括以下列出的按照在图25A-25B中图示的顺序的如下字段。

[0210] 格式字段2540——该字段中的特定值(指令格式标识符值)唯一地标识向量友好指令格式,并且由此标识指令在指令流中以向量友好指令格式出现。由此,该字段对于仅具有通用向量友好指令格式的指令集是不需要的,在这个意义上该字段是任选的。

[0211] 基础操作字段2542——其内容区分不同的基础操作。

[0212] 寄存器索引字段2544——其内容直接或者通过地址生成来指定源或目的地操作

数在寄存器中或者在存储器中的位置。这些字段包括足够数量的位以从PxQ(例如,32x512、16x128、32x1024、64x1024)寄存器堆中选择N个寄存器。尽管在一个实施例中N可多达三个源寄存器和一个目的地寄存器,但是替代实施例可支持更多或更少的源和目的地寄存器(例如,可支持多达两个源,其中这些源中的一个源还用作目的地;可支持多达三个源,其中这些源中的一个源还用作目的地;可支持多达两个源和一个目的地)。

[0213] 修饰符(modifier)字段2546——其内容将指定存储器访问的以通用向量指令格式出现的指令与不指定存储器访问的以通用向量指令格式出现的指令区分开;即在无存储器访问2505的指令模板与存储器访问2520的指令模板之间进行区分。存储器访问操作读取和/或写入到存储器层次(在一些情况下,使用寄存器中的值来指定源和/或目的地地址),而非存储器访问操作不这样(例如,源和目的地是寄存器)。尽管在一个实施例中,该字段还在三种不同的方式之间选择以执行存储器地址计算,但是替代实施例可支持更多、更少或不同的方式来执行存储器地址计算。

[0214] 扩充操作字段2550——其内容区分除基础操作以外还要执行各种不同操作中的哪一个操作。该字段是针对上下文的。在一个实施例中,该字段被分成类字段2568、α字段2552和β字段2554。扩充操作字段2550允许在单条指令而非2条、3条或4条指令中执行多组共同的操作。

[0215] 比例字段2560——其内容允许用于存储器地址生成(例如,用于使用($2^{比例}$*索引+基址)的地址生成)的索引字段的内容的按比例缩放。

[0216] 位移字段2562A——其内容用作存储器地址生成的一部分(例如,用于使用($2^{比例}$*索引+基址+位移)的地址生成)。

[0217] 位移因数字段2562B(注意,位移字段2562A直接在位移因数字段2562B上的并置指示使用一个或另一个)——其内容用作地址生成的一部分;它指定将按比例缩放存储器访问的尺寸(N)的位移因数——其中N是存储器访问中的字节数量(例如,用于使用($2^{比例}$*索引+基址+按比例缩放的位移)的地址生成)。忽略冗余的低阶位,并且因此将位移因数字段的内容乘以存储器操作数总尺寸(N)以生成将在计算有效地址中使用的最终位移。N的值由处理器硬件在运行时基于完整操作码字段2574(稍后在本文中描述)和数据操纵字段2554C确定。位移字段2562A和位移因数字段2562B不用于无存储器访问2505的指令模板和/或不同的实施例可实现这两者中的仅一个或不实现这两者中的任一个,在这个意义上,位移字段2562A和位移因数字段2562B是任选的。

[0218] 数据元素宽度字段2564——其内容区分将使用多个数据元素宽度中的哪一个(在一些实施例中用于所有指令;在其他实施例中只用于指令中的一些指令)。如果支持仅一个数据元素宽度和/或使用操作码的某一方面来支持数据元素宽度,则该字段是不需要的,在这个意义上,该字段是任选的。

[0219] 写掩码字段2570——其内容逐数据元素位置地控制目的地向量操作数中的数据元素位置是否反映基础操作和扩充操作的结果。A类指令模板支持合并-写掩码,而B类指令模板支持合并-写掩码和归零-写掩码两者。当合并时,向量掩码允许在执行(由基础操作和扩充操作指定的)任何操作期间保护目的地中的任何元素集免于更新;在另一实施例中,保持其中对应掩码位具有0的目的地的每一元素的旧值。相反,当归零时,向量掩码允许在执行(由基础操作和扩充操作指定的)任何操作期间使目的地中的任何元素集归零;在一个实

施例中，目的地的元素在对应掩码位具有0值时被设为0。该功能的子集是控制正被执行的操作的向量长度的能力(即，从第一个到最后一个正被修改的元素的跨度)，然而，被修改的元素不一定要是连续的。由此，写掩码字段2570允许部分向量操作，这包括加载、存储、算术、逻辑等。尽管描述了其中写掩码字段2570的内容选择了多个写掩码寄存器中的包含要使用的写掩码的一个写掩码寄存器(并且由此，写掩码字段2570的内容间接地标识要执行的掩码)的实施例，但是替代实施例替代地或附加地允许掩码写字段2570的内容直接指定要执行的掩码。

[0220] 立即数字段2572——其内容允许对立即数的指定。该字段在实现不支持立即数的通用向量友好格式中不存在且在不使用立即数的指令中不存在，在这个意义上，该字段是任选的。

[0221] 类字段2568——其内容在不同类的指令之间进行区分。参考图25A-图25B，该字段的内容在A类和B类指令之间进行选择。在图25A-图25B中，圆角方形用于指示特定的值存在于字段中(例如，在图25A-图25B中分别用于类字段2568的A类2568A和B类2568B)。

A类指令模板

[0222] 在A类非存储器访问2505的指令模板的情况下，α字段2552被解释为其内容区分要执行不同扩充操作类型中的哪一种(例如，针对无存储器访问的舍入型操作2510和无存储器访问的数据变换型操作2515的指令模板分别指定舍入2552A.1和数据变换2552A.2)的RS字段2552A，而β字段2554区分要执行所指定类型的操作中的哪一种。在无存储器访问2505的指令模板中，比例字段2560、位移字段2562A和位移比例字段2562B不存在。

无存储器访问的指令模板——完全舍入控制型操作

[0223] 在无存储器访问的完全舍入控制型操作2510的指令模板中，β字段2554被解释为其(多个)内容提供静态舍入的舍入控制字段2554A。尽管在所述实施例中舍入控制字段2554A包括抑制所有浮点异常(SAE)字段2556和舍入操作控制字段2558，但是替代实施例可支持这两个概念，可将这两个概念编码为同一字段，或仅具有这些概念/字段中的一个或另一个(例如，可仅具有舍入操作控制字段2558)。

[0224] SAE字段2556——其内容区分是否禁用异常事件报告；当SAE字段2556的内容指示启用抑制时，给定的指令不报告任何种类的浮点异常标志，并且不唤起任何浮点异常处置程序。

[0225] 舍入操作控制字段2558——其内容区分要执行一组舍入操作中的哪一个(例如，向上舍入、向下舍入、向零舍入以及就近舍入)。由此，舍入操作控制字段2558允许逐指令地改变舍入模式。在其中处理器包括用于指定舍入模式的控制寄存器的一个实施例中，舍入操作控制字段2550的内容覆盖(override)该寄存器值。

无存储器访问的指令模板－数据变换型操作

[0226] 在无存储器访问的数据变换型操作2515的指令模板中，β字段2554被解释为数据变换字段2554B，其内容区分要执行多个数据变换中的哪一个(例如，无数据变换、混合、广播)。

[0227] 在A类存储器访问2520的指令模板的情况下，α字段2552被解释为驱逐提示字段2552B，其内容区分要使用驱逐提示中的哪一个(在图25A中，对于存储器访问时效性2525的指令模板和存储器访问非时效性2530的指令模板分别指定时效性的2552B.1和非时效性的

2552B.2)，而β字段2554被解释为数据操纵字段2554C，其内容区分要执行多个数据操纵操作(也称为基元(primitive))中的哪一个(例如，无操纵、广播、源的向上转换以及目的地的向下转换)。存储器访问2520的指令模板包括比例字段2560，并任选地包括位移字段2562A或位移比例字段2562B。

[0228] 向量存储器指令使用转换支持来执行来自存储器的向量加载以及向存储器的向量存储。如同寻常的向量指令，向量存储器指令以数据元素式的方式从/向存储器传输数据，其中实际被传输的元素由被选为写掩码的向量掩码的内容规定。

存储器访问的指令模板——时效性的

[0229] 时效性的数据是可能足够快地被重新使用以从高速缓存操作受益的数据。然而，这是提示，并且不同的处理器能以不同的方式实现它，包括完全忽略该提示。

存储器访问的指令模板——非时效性的

[0263] 非时效性的数据是不太可能足够快地被重新使用以从第一级高速缓存中的高速缓存操作受益且应当被给予驱逐优先级的数据。然而，这是提示，并且不同的处理器能以不同的方式实现它，包括完全忽略该提示。

B类指令模板

[0231] 在B类指令模板的情况下，α字段2552被解释为写掩码控制(Z)字段2552C，其内容区分由写掩码字段2570控制的写掩码应当是合并还是归零。

[0232] 在B类非存储器访问2505的指令模板的情况下，β字段2554的一部分被解释为RL字段2557A，其内容区分要执行不同扩充操作类型中的哪一种(例如，针对无存储器访问的写掩码控制部分舍入控制类型操作2512的指令模板和无存储器访问的写掩码控制VSIZE型操作2517的指令模板分别指定舍入2557A.1和向量长度(VSIZE)2557A.2)，而β字段2554的其余部分区分要执行所指定类型的操作中的哪一种。在无存储器访问2505的指令模板中，比例字段2560、位移字段2562A和位移比例字段2562B不存在。

[0233] 在无存储器访问的写掩码控制部分舍入控制型操作2510的指令模板中，β字段2554的其余部分被解释为舍入操作字段2559A，并且禁用异常事件报告(给定的指令不报告任何种类的浮点异常标志，并且不唤起任何浮点异常处置程序)。

[0234] 舍入操作控制字段2559A——正如舍入操作控制字段2558，其内容区分要执行一组舍入操作中的哪一个(例如，向上舍入、向下舍入、向零舍入以及就近舍入)。由此，舍入操作控制字段2559A允许逐指令地改变舍入模式。在其中处理器包括用于指定舍入模式的控制寄存器的一个实施例中，舍入操作控制字段2550的内容覆盖该寄存器值。

[0235] 在无存储器访问的写掩码控制VSIZE型操作2517的指令模板中，β字段2554的其余部分被解释为向量长度字段2559B，其内容区分要执行多个数据向量长度中的哪一个(例如，128字节、256字节或512字节)。

[0236] 在B类存储器访问2520的指令模板的情况下，β字段2554的一部分被解释为广播字段2557B，其内容区分是否要执行广播型数据操纵操作，而β字段2554的其余部分被解释为向量长度字段2559B。存储器访问2520的指令模板包括比例字段2560，并任选地包括位移字段2562A或位移比例字段2562B。

[0237] 针对通用向量友好指令格式2500，示出完整操作码字段2574包括格式字段2540、基础操作字段2542和数据元素宽度字段2564。尽管示出了其中完整操作码字段2574包括所

有这些字段的一个实施例,但是在不支持所有这些字段的实施例中,完整操作码字段2574包括少于所有的这些字段。完整操作码字段2574提供操作代码(操作码)。

[0238] 扩充操作字段2550、数据元素宽度字段2564和写掩码字段2570允许逐指令地以通用向量友好指令格式指定这些特征。

[0239] 写掩码字段和数据元素宽度字段的组合创建各种类型的指令,因为这些指令允许基于不同的数据元素宽度应用该掩码。

[0240] 在A类和B类内出现的各种指令模板在不同的情形下是有益的。在一些实施例中,不同处理器或处理器内的不同核可支持仅A类、仅B类、或者可支持这两类。举例而言,旨在用于通用计算的高性能通用乱序核可仅支持B类,旨在主要用于图形和/或科学(吞吐量)计算的核可仅支持A类,并且旨在用于通用计算和图形和/或科学(吞吐量)计算两者的核可支持A类和B类两者(当然,具有来自这两类的模板和指令的一些混合、但是并非来自这两类的所有模板和指令的核在本发明的范围内)。同样,单个处理器可包括多个核,这多个核全部都支持相同的类,或者其中不同的核支持不同的类。举例而言,在具有单独的图形核和通用核的处理器中,图形核中的旨在主要用于图形和/或科学计算的一个核可仅支持A类,而通用核中的一个或多个可以是具有旨在用于通用计算的仅支持B类的乱序执行和寄存器重命名的高性能通用核。不具有单独的图形核的另一处理器可包括既支持A类又支持B类的一个或多个通用有序或乱序核。当然,在不同实施例中,来自一类的特征也可在其他类中实现。将使以高级语言编写的程序成为(例如,及时编译或静态编译)各种不同的可执行形式,这些可执行形式包括:1)仅具有由用于执行的目标处理器支持的(多个)类的指令的形式;或者2)具有替代例程并具有控制流代码的形式,该替代例程使用所有类的指令的不同组合来编写,该控制流代码选择这些例程以基于由当前正在执行代码的处理器支持的指令来执行。

示例性专用向量友好指令格式

[0241] 图26A是图示根据实施例的示例性专用向量友好指令格式的框图。图26A示出专用向量友好指令格式2600,其指定各字段的位置、尺寸、解释和次序、以及那些字段中的一些字段的值,在这个意义上,该专用向量友好指令格式2600是专用的。专用向量友好指令格式2600可用于扩展x86指令集,并且由此字段中的一些字段与如在现有的x86指令集及其扩展(例如,AVX)中所使用的那些字段类似或相同。该格式保持与具有扩展的现有x86指令集的前缀编码字段、实操作码字节字段、MOD R/M字段、SIB字段、位移字段和立即数字段一致。图示来自图25的字段,来自图26A的字段映射到来自图25的字段。

[0242] 应当理解,虽然出于说明的目的在通用向量友好指令格式2500的上下文中参考专用向量友好指令格式2600描述了实施例,但是本发明不限于专用向量友好指令格式2600,除非另有声明。例如,通用向量友好指令格式2500构想了各种字段的各种可能的尺寸,而专用向量友好指令格式2600示出为具有特定尺寸的字段。作为具体示例,尽管在专用向量友好指令格式2600中数据元素宽度字段2564被图示为一位字段,但是本发明不限于此(即,通用向量友好指令格式2500构想数据元素宽度字段2564的其他尺寸)。

[0243] 通用向量友好指令格式2500包括以下列出的按照图26A中图示的顺序的如下字段。

[0244] EVEX前缀2602(字节0-3)——以四字节形式进行编码。

[0245] 格式字段2540(EVEX字节0,位[7:0])——第一字节(EVEX字节0)是格式字段2540,并且它包含0x62(在一个实施例中,为用于区分向量友好指令格式的唯一值)。

[0246] 第二－第四字节(EVEX字节1-3)包括提供专用能力的多个位字段。

[0247] REX字段2605(EVEX字节1,位[7-5])——由EVEX.R位字段(EVEX字节1,位[7]-R)、EVEX.X位字段(EVEX字节1,位[6]-X)以及(2557BEX字节1,位[5]-B)组成。EVEX.R、EVEX.X和EVEX.B位字段提供与对应的VEX位字段相同的功能,并且使用1补码的形式进行编码,即ZMM0被编码为1111B,ZMM15被编码为0000B。这些指令的其他字段对如在本领域中已知的寄存器索引的较低三个位(rrr、xxx和bbb)进行编码,由此可通过对EVEX.R、EVEX.X和EVEX.B相加来形成Rrrr、Xxxx和Bbbb。

[0248] REX'字段2510——这是REX'字段2510的第一部分,并且是用于对扩展的32个寄存器集合的较高16个或较低16个寄存器进行编码的EVEX.R'位字段(EVEX字节1,位[4]-R')。在一个实施例中,该位与以下指示的其他位一起以位反转的格式存储以(在公知x86的32位模式下)与BOUND指令进行区分,该BOUND指令的实操作码字节是62,但是在MOD R/M字段(在下文中描述)中不接受MOD字段中的值11;替代实施例不以反转的格式存储该指示的位以及以下其他指示的位。值1用于对较低16个寄存器进行编码。换句话说,通过组合EVEX.R'、EVEX.R以及来自其他字段的其他RRR来形成R'Rrrr。

[0249] 操作码映射字段2615(EVEX字节1,位[3:0]-mmmm)——其内容对隐含的前导操作码字节(0F、0F 38或0F 3)进行编码。

[0250] 数据元素宽度字段2564(EVEX字节2,位[7]-W)——由记号EVEX.W表示。EVEX.W用于定义数据类型(32位数据元素或64位数据元素)的粒度(尺寸)。

[0251] EVEX.vvvv 2620(EVEX字节2,位[6:3]-vvvv)——EVEX.vvvv的作用可包括如下:1)EVEX.vvvv对以反转(1补码)形式指定的第一源寄存器操作数进行编码,并且对具有两个或更多个源操作数的指令有效;2)EVEX.vvvv对针对特定向量位移以1补码的形式指定的目的地寄存器操作数进行编码;或者3)EVEX.vvvv不对任何操作数进行编码,该字段被预留,并且应当包含1111b。由此,EVEX.vvvv字段2620对以反转(1补码)的形式存储的第一源寄存器指定符的4个低阶位进行编码。取决于该指令,额外不同的EVEX位字段用于将指定符尺寸扩展到32个寄存器。

[0252] EVEX.U 2568类字段(EVEX字节2,位[2]-U)——如果EVEX.U＝0,则它指示A类或EVEX.U0;如果EVEX.U＝1,则它指示B类或EVEX.U1。

[0253] 前缀编码字段2625(EVEX字节2,位[1:0]-pp)——提供了用于基础操作字段的附加位。除了对以EVEX前缀格式的传统SSE指令提供支持以外,这也具有压缩SIMD前缀的益处(EVEX前缀仅需要2位,而不是需要字节来表达SIMD前缀)。在一个实施例中,为了支持使用以传统格式和以EVEX前缀格式两者的SIMD前缀(66H、F2H、F3H)的传统SSE指令,将这些传统SIMD前缀编码成SIMD前缀编码字段;并且在运行时在被提供给解码器的PLA之前被扩展成传统SIMD前缀(因此,在无需修改的情况下,PLA既可执行传统格式的这些传统指令又可执行EVEX格式的这些传统指令)。虽然较新的指令可将EVEX前缀编码字段的内容直接用作操作码扩展,但是为了一致性,特定实施例以类似的方式扩展,但允许由这些传统SIMD前缀指定的不同含义。替代实施例可重新设计PLA以支持2位SIMD前缀编码,并且由此不需要扩展。

[0254] α字段2552(EVEX字节3,位[7]-EH,也称为EVEX.EH、EVEX.rs、EVEX.RL、EVEX.写掩

码控制、以及EVEX.N;也以α图示)——如先前所述,该字段是针对上下文的。

[0255] β字段2554(EVEX字节3,位[6:4]-SSS,也称为EVEX.s_{2-0}、EVEX.r_{2-0}、EVEX.rr1、EVEX.LL0、EVEX.LLB,还以βββ图示)——如前所述,此字段是针对上下文的。

[0256] REX'字段2510——这是REX'字段的其余部分,并且是可用于对扩展的32个寄存器集合的较高16个或较低16个寄存器进行编码的EVEX.V'位字段(EVEX字节3,位[3]-V')。该位以位反转的格式存储。值1用于对较低16个寄存器进行编码。换句话说,通过组合EVEX.V'、EVEX.vvvv来形成V'VVVV。

[0257] 写掩码字段2570(EVEX字节3,位[2:0]-kkk)——其内容指定写掩码寄存器中的寄存器的索引,如先前所述。在一个实施例中,特定值EVEX.kkk=000具有暗示没有写掩码用于特定指令的特殊行为(这能以各种方式实现,包括使用硬连线到所有对象的写掩码或绕过掩码硬件的硬件来实现)。

[0258] 实操作码字段2630(字节4)还被称为操作码字节。操作码的一部分在该字段中被指定。

[0259] MOD R/M字段2640(字节5)包括MOD字段2642、Reg字段2644和R/M字段2646。如先前所述的,MOD字段2642的内容将存储器访问操作和非存储器访问操作区分开。Reg字段2644的作用可被归结为两种情形:对目的地寄存器操作数或源寄存器操作数进行编码;或者被视为操作码扩展,并且不用于对任何指令操作数进行编码。R/M字段2646的作用可包括如下:对引用存储器地址的指令操作数进行编码;或者对目的地寄存器操作数或源寄存器操作数进行编码。

[0260] 比例、索引、基址(SIB)字节(字节6)——如先前所述的,SIB2650的内容用于存储器地址生成。SIB.xxx 2654和SIB.bbb 2656——先前已经针对寄存器索引Xxxx和Bbbb提及了这些字段的内容。

[0261] 位移字段2562A(字节7-10)——当MOD字段2642包含10时,字节7-10是位移字段2562A,并且它与传统32位位移(disp32)一样地工作,并且以字节粒度工作。

[0262] 位移因数字段2562B(字节7)——当MOD字段2642包含01时,字节7是位移因数字段2562B。该字段的位置与以字节粒度工作的传统x86指令集8位位移(disp8)的位置相同。由于disp8是符号扩展的,因此它仅能在-128和127字节偏移之间寻址;在64字节高速缓存行的方面,disp8使用可被设为仅四个真正有用的值-128、-64、0和64的8位;由于常常需要更大的范围,所以使用disp32;然而,disp32需要4个字节。与disp8和disp32对比,位移因数字段2562B是disp8的重新解释;当使用位移因数字段2562B时,通过将位移因数字段的内容乘以存储器操作数访问的尺寸(N)来确定实际位移。该类型的位移被称为disp8*N。这减小了平均指令长度(单个字节用于位移,但具有大得多的范围)。此类经压缩的位移假设有效位移是存储器访问的粒度的倍数,并且由此地址偏移的冗余低阶位不需要被编码。换句话说,位移因数字段2562B替代传统x86指令集8位位移。由此,位移因数字段2562B以与x86指令集8位位移相同的方式被编码(因此,在ModRM/SIB编码规则中没有变化),唯一的不同在于,将disp8超载至disp8*N。换句话说,在编码规则或编码长度方面没有变化,而仅在有硬件对位移值的解释方面有变化(这需要将位移按比例缩放存储器操作数的尺寸以获得字节式地址偏移)。立即数字段2572如先前所述地操作。

完整操作码字段

[0263] 图26B是图示根据一个实施例的构成完整操作码字段2574的具有专用向量友好指令格式2600的字段的框图。具体地，完整操作码字段2574包括格式字段2540、基础操作字段2542和数据元素宽度(W)字段2564。基础操作字段2542包括前缀编码字段2625、操作码映射字段2615和实操作码字段2630。

寄存器索引字段

[0264] 图26C是图示根据一个实施例的构成寄存器索引字段2544的具有专用向量友好指令格式2600的字段的框图。具体地，寄存器索引字段2544包括REX 2605字段、REX'2610字段、MODR/M.reg字段2644、MODR/M.r/m字段2646、VVVV字段2620、xxx字段2654和bbb字段2656。

扩充操作字段

[0265] 图26D是图示根据一个实施例的构成扩充操作字段2550的具有专用向量友好指令格式2600的字段的框图。当类(U)字段2568包含0时，它表明EVEX.U0(A类2568A)；当它包含1时，它表明EVEX.U1(B类2568B)。当U＝0且MOD字段2642包含11(表明无存储器访问操作)时，α字段2552(EVEX字节3，位[7]-EH)被解释为rs字段2552A。当rs字段2552A包含1(舍入2552A.1)时，β字段2554(EVEX字节3，位[6:4]-SSS)被解释为舍入控制字段2554A。舍入控制字段2554A包括一位SAE字段2556和两位舍入操作字段2558。当rs字段2552A包含0(数据变换2552A.2)时，β字段2554(EVEX字节3，位[6:4]-SSS)被解释为三位数据变换字段2554B。当U＝0且MOD字段2642包含00、01或10(表明存储器访问操作)时，α字段2552(EVEX字节3，位[7]-EH)被解释为驱逐提示(EH)字段2552B，并且β字段2554(EVEX字节3，位[6:4]-SSS)被解释为三位数据操纵字段2554C。

[0266] 当U＝1时，α字段2552(EVEX字节3，位[7]-EH)被解释为写掩码控制(Z)字段2552C。当U＝1且MOD字段2642包含11(表明无存储器访问操作)时，β字段2554的一部分(EVEX字节3，位[4]-S_0)被解释为RL字段2557A；当它包含1(舍入2557A.1)时，β字段2554的其余部分(EVEX字节3，位[6-5]-S_{2-1})被解释为舍入操作字段2559A，而当RL字段2557A包含0(VSIZE 2557A.2)时，β字段2554的其余部分(EVEX字节3，位[6-5]-S_{2-1})被解释为向量长度字段2559B(EVEX字节3，位[6-5]-L_{1-0})。当U＝1且MOD字段2642包含00、01或10(表明存储器访问操作)时，β字段2554(EVEX字节3，位[6:4]-SSS)被解释为向量长度字段2559B(EVEX字节3，位[6-5]-L_{1-0})和广播字段2557B(EVEX字节3，位[4]-B)。

示例性寄存器架构

[0267] 图27是根据一个实施例的寄存器架构2700的框图。在所图示的实施例中，有32个512位宽的向量寄存器2710；这些寄存器被引用为zmm0到zmm31。较低的16个zmm寄存器的较低阶256个位覆盖(overlay)在寄存器ymm0-16上。较低的16个zmm寄存器的较低阶128个位(ymm寄存器的较低阶128个位)覆盖在寄存器xmm0-15上。专用向量友好指令格式2600对这些被覆盖的寄存器堆操作，如在以下表格中所图示。

可调节向量长度	类	操作	寄存器
不包括向量长度字段 2559B 的指令模板	A（图 25A; U=0)	2510、2515、2525、2530	zmm 寄存器(向量长度是 64 字节)
	B（图 25B; U=1)	2512	zmm 寄存器(向量长度是 64 字节)
包括向量长度字段 2559B 的指令模板	B（图 25B; U=1)	2517、2527	zmm、ymm、或 xmm 寄存器（向量长度是 64 字节、32 字节、或 16 字节），取决于向量长度字段 2559B

[0268] 换句话说,向量长度字段2559B在最大长度与一个或多个其他较短长度之间进行选择,其中每一个此类较短长度是前一长度的一半,并且不具有向量长度字段2559B的指令模板在最大向量长度上操作。此外,在一个实施例中,专用向量友好指令格式2600的B类指令模板对紧缩或标量单/双精度浮点数据以及紧缩或标量整数数据操作。标量操作是对zmm/ymm/xmm寄存器中的最低阶数据元素位置执行的操作;取决于实施例,较高阶数据元素位置要么保持与在指令之前相同,要么归零。

[0269] 写掩码寄存器2715——在所图示的实施例中,存在8个写掩码寄存器(k0至k7),每一写掩码寄存器的尺寸是64位。在替代实施例中,写掩码寄存器2715的尺寸是16位。如先前所述,在一个实施例中,向量掩码寄存器k0无法用作写掩码;当将正常指示k0的编码用作写掩码时,它选择硬连线的写掩码0xFFFF,从而有效地禁止写掩码用于那条指令。

[0270] 通用寄存器2725——在所示出的实施例中,有十六个64位通用寄存器,这些寄存器与现有的x86寻址模式一起使用以对存储器操作数寻址。这些寄存器通过名称RAX、RBX、RCX、RDX、RBP、RSI、RDI、RSP以及R8到R15来引用。

[0271] 标量浮点栈寄存器堆(x87栈)2745,在其上面重叠了MMX紧缩整数平坦寄存器堆2750——在所图示的实施例中,x87栈是用于使用x87指令集扩展来对32/64/80位浮点数据执行标量浮点操作的八元素栈;而使用MMX寄存器来对64位紧缩整数数据执行操作,以及为在MMX与XMM寄存器之间执行的一些操作保存操作数。

[0272] 替代实施例可以使用更宽的或更窄的寄存器。另外,替代实施例可以使用更多、更少或不同的寄存器堆和寄存器。

示例性核架构、处理器和计算机架构

[0273] 处理器核能以不同方式、出于不同的目的、在不同的处理器中实现。例如,此类核的实现可以包括: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乱序核;3)旨在主要用于图形和/或科学(吞吐量)计算的专用核。不同处理器的实现可包

括:1)CPU,其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乱序核;以及2)协处理器,其包括旨在主要用于图形和/或科学(吞吐量)的一个或多个专用核。此类不同的处理器导致不同的计算机系统架构,这些计算机系统架构可包括:1)在与CPU分开的芯片上的协处理器;2)在与CPU相同的封装中但在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,此类协处理器有时被称为专用逻辑或被称为专用核,该专用逻辑诸如,集成图形和/或科学(吞吐量)逻辑);以及4)芯片上系统,其可以将所描述的CPU(有时被称为(多个)应用核或(多个)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。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。

示例性核架构

有序和乱序核框图

[0274] 图28A是图示根据各实施例的示例性有序流水线和示例性的寄存器重命名的乱序发布/执行流水线的框图。图28B是示出根据各实施例的要包括在处理器中的有序架构核的示例性实施例和示例性的寄存器重命名的乱序发布/执行架构核的框图。图28A-图28B中的实线框图示有序流水线和有序核,而虚线框的任选增加图示寄存器重命名的、乱序发布/执行流水线和核。考虑到有序方面是乱序方面的子集,将描述乱序方面。

[0275] 在图28A中,处理器流水线2800包括取出级2802、长度解码级2804、解码级2806、分配级2808、重命名级2810、调度(也被称为分派或发布)级2812、寄存器读取/存储器读取级2814、执行级2816、写回/存储器写入级2818、异常处置级2822和提交级2824。

[0276] 图28B示出处理器核2890,该处理器核2890包括前端单元2830,该前端单元2830耦合到执行引擎单元2850,并且前端单元2830和执行引擎单元2850两者都耦合到存储器单元2870。核2890可以是精简指令集计算(RISC)核、复杂指令集计算(CISC)核、超长指令字(VLIW)核、或混合或替代的核类型。作为又一选项,核2890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、通用计算图形处理单元(GPGPU)核、图形核,等等。

[0277] 前端单元2830包括分支预测单元2832,该分支预测单元2832耦合到指令高速缓存单元2834,该指令高速缓存单元2834耦合到指令转换后备缓冲器(TLB)2836,该指令转换后备缓冲器2836耦合到指令取出单元2838,该指令取出单元2838耦合到解码单元2840。解码单元2840(或解码器)可对指令解码,并且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2840可使用各种不同的机制来实现。合适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(PLA)、微代码只读存储器(ROM)等。在一个实施例中,核2890包括存储用于某些宏指令的微代码的微代码ROM或其他介质(例如,在解码单元2840中,或以其他方式在前端单元2830内)。解码单元2840耦合到执行引擎单元2850中的重命名/分配器单元2852。

[0278] 执行引擎单元2850包括重命名/分配器单元2852,该重命名/分配器单元2852耦合到引退单元2854和一个或多个调度器单元的集合2856。(多个)调度器单元2856表示任何数量的不同调度器,包括预留站、中央指令窗等。(多个)调度器单元2856耦合到(多个)物理寄存器堆单元2858。(多个)物理寄存器堆单元2858中的每一个物理寄存器堆单元表示一个或多个物理寄存器堆,其中不同的物理寄存器堆存储一种或多种不同的数据类型,诸如,标量

整数、标量浮点、紧缩整数、紧缩浮点、向量整数、向量浮点，状态(例如，作为要执行的下一条指令的地址的指令指针)等等。在一个实施例中，(多个)物理寄存器堆单元2858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。(多个)物理寄存器堆单元2858由引退单元2854重叠，以图示可实现寄存器重命名和乱序执行的各种方式(例如，使用(多个)重排序缓冲器和(多个)引退寄存器堆；使用(多个)未来文件、(多个)历史缓冲器、(多个)引退寄存器堆；使用寄存器映射和寄存器池，等等)。引退单元2854和(多个)物理寄存器堆单元2858耦合到(多个)执行集群2860。(多个)执行集群2860包括一个或多个执行单元的集合2862以及一个或多个存储器访问单元的集合2864。执行单元2862可执行各种操作(例如，移位、加法、减法、乘法)并可对各种数据类型(例如，标量浮点、紧缩整数、紧缩浮点、向量整数、向量浮点)执行。尽管一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，但是其他实施例可包括仅一个执行单元或全都执行所有功能的多个执行单元。(多个)调度器单元2856、(多个)物理寄存器堆单元2858和(多个)执行集群2860示出为可能有多个，因为某些实施例为某些类型的数据/操作创建分开的流水线(例如，标量整数流水线、标量浮点/紧缩整数/紧缩浮点/向量整数/向量浮点流水线，和/或各自具有其自身的调度器单元、(多个)物理寄存器堆单元和/或执行集群的存储器访问流水线——并且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集群具有(多个)存储器访问单元2864的某些实施例)。还应当理解，在使用分开的流水线的情况下，这些流水线中的一个或多个可以是乱序发布/执行，并且其余流水线可以是有序的。

[0279] 存储器访问单元的集合2864耦合到存储器单元2870，该存储器单元2870包括数据TLB单元2872，该数据TLB单元2872耦合到数据高速缓存单元2874，该数据高速缓存单元2874耦合到第二级(L2)高速缓存单元2876。在一个示例性实施例中，存储器访问单元2864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到存储器单元2870中的数据TLB单元2872。指令高速缓存单元2834还耦合到存储器单元2870中的第二级(L2)高速缓存单元2876。L2高速缓存单元2876耦合到一个或多个其他级别的高速缓存，并最终耦合到主存储器。

[0280] 作为示例，示例性寄存器重命名的乱序发布/执行核架构可如下所述地实现流水线2800：1)指令取出2838执行取出级2802和长度解码级2804；2)解码单元2840执行解码级2806；3)重命名/分配器单元2852执行分配级2808和重命名级2810；4)(多个)调度器单元2856执行调度级2812；5)(多个)物理寄存器堆单元2858和存储器单元2870执行寄存器读取/存储器读取级2814；执行集群2860执行执行级2816；6)存储器单元2870和(多个)物理寄存器堆单元2858执行写回/存储器写入级2818；7)各单元可牵涉到异常处置级2822；以及8)引退单元2854和(多个)物理寄存器堆单元2858执行提交级2824。

[0281] 核2890可支持一个或多个指令集(例如，x86指令集(具有已与较新版本一起添加的一些扩展)；加利福尼亚州桑尼维尔市的MIPS技术公司的MIPS指令集；加利福尼亚州桑尼维尔市的ARM控股公司的ARM指令集(具有诸如NEON的任选的附加扩展))，其中包括本文中描述的(多条)指令。在一个实施例中，核2890包括用于支持紧缩数据指令集扩展(例如，AVX1、AVX2)的逻辑，由此允许使用紧缩数据来执行由许多多媒体应用使用的操作。

[0282] 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，各种方式包括时分多线程化、同时多线程化(其中单个物理核为物理核正在同时多线程化的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的诸如英特尔®超线程化技术中的同时多线程化)。

[0283] 尽管在乱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。尽管所图示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2834/2874以及共享的L2高速缓存单元2876，但是替代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，第一级(L1)内部高速缓存或多个级别的内部高速缓存。在一些实施例中，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。

具体的示例性有序核架构

[0284] 图29A-图29B图示更具体的示例性有序核架构的框图，该核将是芯片中的若干逻辑块(包括相同类型和/或不同类型的其他核)中的一个逻辑块。取决于应用，逻辑块通过高带宽互连网络(例如，环形网络)与一些固定的功能逻辑、存储器I/O接口和其他必要的I/O逻辑进行通信。

[0285] 图29A是根据实施例的单个处理器核以及它至管芯上互连网络2902的连接及其第二级(L2)高速缓存的本地子集2904的框图。在一个实施例中，指令解码器2900支持具有紧缩数据指令集扩展的x86指令集。L1高速缓存2906允许对进入标量和向量单元中的、对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2908和向量单元2910使用分开的寄存器集合(分别为标量寄存器2912和向量寄存器2914)，并且在这些寄存器之间传输的数据被写入到存储器，并随后从第一级(L1)高速缓存2906读回，但是替代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堆之间传输而无需被写入和读回的通信路径)。

[0286] L2高速缓存的本地子集2904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，每个处理器核一个本地子集。每个处理器核具有到其自身的L2高速缓存的本地子集2904的直接访问路径。由处理器核读取的数据被存储在其L2高速缓存子集2904中，并且可以与其他处理器核访问其自身的本地L2高速缓存子集并行地被快速访问。由处理器核写入的数据被存储在其自身的L2高速缓存子集2904中，并在必要的情况下从其他子集转储清除。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高速缓存和其他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。

[0287] 图29B是根据实施例的图29A中的处理器核的一部分的展开图。图29B包括L1高速缓存2904的L1数据高速缓存2906A部分，以及关于向量单元2910和向量寄存器2914的更多细节。具体地，向量单元2910是16宽向量处理单元(VPU)(见16宽ALU 2928)，该单元执行整数、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2920支持对寄存器输入的混合，通过数值转换单元2922A-B支持数值转换，并且通过复制单元2924支持对存储器输入的复制。写掩码寄存器2926允许掩蔽所得的向量写入。

[0288] 图30是根据实施例的可具有多于一个的核、可具有集成存储器控制器、以及可具有集成图形器件的处理器3000的框图。图30中的实线框图示具有单个核3002A、系统代理3010、一个或多个总线控制器单元的集合3016的处理器3000，而虚线框的任选增加图示具

有多个核3002A-N、系统代理单元3010中的一个或多个集成存储器控制器单元的集合3014以及专用逻辑3008的替代处理器3000。

[0289] 因此，处理器3000的不同实现可包括：1)CPU，其中专用逻辑3008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3002A-N是一个或多个通用核(例如，通用有序核、通用乱序核、这两者的组合)；2)协处理器，其中核3002A-N是旨在主要用于图形和/或科学(吞吐量)的大量专用核；以及3)协处理器，其中核3002A-N是大量通用有序核。因此，处理器3000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)、嵌入式处理器，等等。该处理器可以被实现在一个或多个芯片上。处理器3000可以是一个或多个基板的一部分，和/或可使用多种工艺技术(诸如例如，BiCMOS、CMOS、或NMOS)中的任何技术被实现在一个或多个基板上。

[0290] 存储器层次结构包括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的集合3006、以及耦合到集成存储器控制器单元的集合3014的外部存储器(未示出)。共享高速缓存单元的集合3006可包括一个或多个中间级别的高速缓存，诸如，第二级(L2)、第三级(L3)、第四级(L4)或其他级别的高速缓存、末级高速缓存(LLC)和/或以上各项的组合。虽然在一个实施例中，基于环的互连单元3012将专用逻辑3008(集成图形逻辑是专用逻辑的示例并且在本文中也被称为专用逻辑)、共享高速缓存单元的集合3006以及系统代理单元3010/(多个)集成存储器控制器单元3014互连，但是替代实施例可使用任何数量的公知技术来互连此类单元。在一个实施例中，在一个或多个高速缓存单元3006与核3002A-N之间维持一致性。

[0291] 在一些实施例中，一个或多个核3002A-N能够实现多线程化。系统代理3010包括协调和操作核3002A-N的那些部件。系统代理单元3010可包括例如功率控制单元(PCU)和显示单元。PCU可以是对核3002A-N以及专用逻辑3008的功率状态进行调节所需的逻辑和部件，或可包括这些逻辑和部件。显示单元用于驱动一个或多个外部连接的显示器。

[0292] 核3002A-N在架构指令集方面可以是同构的或异构的；即，核3002A-N中的两个或更多个核可能能够执行相同的指令集，而其他核可能能够执行该指令集的仅仅子集或不同的指令集。

示例性计算机架构

[0293] 图31-34是示例性计算机架构的框图。本领域中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。一般地，能够包含如本文中所公开的处理器和/或其他执行逻辑的各种各样的系统或电子设备一般都是合适的。

[0294] 现在参考图31，所示出的是根据本发明一个实施例的系统3100的框图。系统3100可以包括一个或多个处理器3110、3115，这些处理器耦合到控制器中枢3120。在一个实施例中，控制器中枢3120包括图形存储器控制器中枢(GMCH)3190和输入/输出中枢(IOH)3150(其可以在分开的芯片上)；GMCH 3190包括存储器和图形控制器，存储器3140和协处理器3145耦合到该存储器和图形控制器；IOH 3150将输入/输出(I/O)设备3160耦合到GMCH

3190。或者，存储器和图形控制器中的一个或这两者被集成在(如本文中所描述的)处理器内，存储器3140和协处理器3145直接耦合到处理器3110，并且控制器中枢3120与IOH 3150处于单个芯片中。

[0295] 附加的处理器3115的任选性在图31中通过虚线来表示。每一处理器3110、3115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3000的某一版本。

[0296] 存储器3140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3120经由诸如前端总线(FSB)之类的多分支总线、诸如快速路径互连(QPI)之类的点对点接口、或者类似的连接3195来与(多个)处理器3110、3115进行通信。

[0297] 在一个实施例中，协处理器3145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、嵌入式处理器，等等。在一个实施例中，控制器中枢3120可以包括集成图形加速器。

[0298] 在物理资源3110、3115之间可以存在包括架构、微架构、热、功耗特性等一系列品质度量方面的各种差异。

[0299] 在一个实施例中，处理器3110执行控制一般类型的数据处理操作的指令。嵌入在这些指令内的可以是协处理器指令。处理器3110将这些协处理器指令识别为具有应当由附连的协处理器3145执行的类型。因此，处理器3110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3145。(多个)协处理器3145接受并执行所接收的协处理器指令。

[0300] 现在参见图32，所示出的是根据本发明的实施例的第一更具体的示例性系统3200的框图。如图32中所示，多处理器系统3200是点对点互连系统，并且包括经由点对点互连3250耦合的第一处理器3270和第二处理器3280。处理器3270和3280中的每一个都可以是处理器3000的某一版本。在一个实施例中，处理器3270和3280分别是处理器3110和3115，而协处理器3238是协处理器3145。在另一实施例中，处理器3270和3280分别是处理器3110和协处理器3145。

[0301] 处理器3270和3280示出为分别包括集成存储器控制器(IMC)单元3272和3282。处理器3270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3276和3278；类似地，第二处理器3280包括P-P接口3286和3288。处理器3270、3280可以经由使用点对点(P-P)接口电路3278、3288的P-P接口3250来交换信息。如图32中所示，IMC 3272和3282将处理器耦合到相应的存储器，即存储器3232和存储器3234，这些存储器可以是本地附连到相应处理器的主存储器的部分。

[0302] 处理器3270、3280可各自经由使用点对点接口电路3276、3294、3286、3298的各个P-P接口3252、3254来与芯片组3290交换信息。芯片组3290可以任选地经由高性能接口3292来与协处理器3238交换信息。在一个实施例中，协处理器3238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、嵌入式处理器，等等。

[0303] 共享高速缓存(未示出)可被包括在任一处理器中，或在这两个处理器的外部但经由P-P互连与这些处理器连接，使得如果处理器被置于低功率模式，则任一个或这两个处理器的本地高速缓存信息可被存储在共享高速缓存中。

[0304] 芯片组3290可以经由接口3296耦合到第一总线3216。在一个实施例中，第一总线

3216可以是外围部件互连(PCI)总线或诸如PCI快速总线或另一第三代I/O互连总线之类的总线，但是本发明的范围不限于此。

[0305] 如图32中所示，各种I/O设备3214可连同总线桥3218一起耦合到第一总线3216，该总线桥3218将第一总线3216耦合到第二总线3220。在一个实施例中，诸如协处理器、高吞吐量MIC处理器、GPGPU、加速器(诸如例如，图形加速器或数字信号处理(DSP)单元)、现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3215耦合到第一总线3216。在一个实施例中，第二总线3220可以是低引脚数(LPC)总线。在一个实施例中，各种设备可耦合到第二总线3220，这些设备包括例如键盘和/或鼠标3222、通信设备3227以及存储单元3228，该存储单元3228诸如可包括指令/代码和数据3230的盘驱动器或者其他大容量存储设备。此外，音频I/O 3224可以被耦合到第二总线3220。注意，其他架构是可能的。例如，代替图32的点对点架构，系统可以实现多分支总线或其他此类架构。

[0306] 现在参考图33，示出的是根据本发明的实施例的第二更具体的示例性系统3300的框图。图32和33中的类似元件使用类似的附图标记，并且从图33中省略了图32的某些方面以避免混淆图33的其他方面。

[0307] 图33图示处理器3270、3280可分别包括集成存储器和I/O控制逻辑(“CL”)3272和3282。因此，CL 3272、3282包括集成存储器控制器单元，并包括I/O控制逻辑。图33图示不仅存储器3232、3234耦合到CL 3272、3282，而且I/O设备3314也耦合到控制逻辑3272、3282。传统I/O设备3315被耦合到芯片组3290。

[0308] 现在参考图34，示出的是根据本发明的实施例的SoC 3400的框图。图30中的类似要素使用类似的附图标记。另外，虚线框是更先进的SoC上的任选的特征。在图34中，(多个)互连单元3402被耦合到：应用处理器3410，其包括一个或多个核的集合3002A-N以及(多个)共享高速缓存单元3006，一个或多个核的集合3002A-N包括高速缓存单元3004A-N；系统代理单元3010；(多个)总线控制器单元3016；(多个)集成存储器控制器单元3014；一个或多个协处理器的集合3420，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3430；直接存储器访问(DMA)单元3432；以及用于耦合到一个或多个外部显示器的显示单元3440。在一个实施例中，(多个)协处理器3420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。

[0309] 本文公开的机制的各实施例可以被实现在硬件、软件、固件或此类实现方式的组合中。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。

[0310] 可将程序代码(诸如，图32中图示的代码3230)应用于输入指令，以执行本文中描述的功能并生成输出信息。可以按已知方式将输出信息应用于一个或多个输出设备。为了本申请的目的，处理系统包括具有处理器的任何系统，该处理器诸如例如，数字信号处理器(DSP)、微控制器、专用集成电路(ASIC)或微处理器。

[0311] 程序代码可以用高级的面向过程的编程语言或面向对象的编程语言来实现，以便与处理系统通信。如果需要，也可用汇编语言或机器语言来实现程序代码。事实上，本文中描述的机制不限于任何特定的编程语言的范围。在任何情况下，该语言可以是编译语言或

解释语言。

[0312] 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，该指令表示处理器中的各种逻辑，该指令在被机器读取时使得该机器制造用于执行本文中所述的技术的逻辑。被称为“IP核”的此类表示可以被存储在有形的机器可读介质上，并可被供应给各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。

[0313] 此类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制品的非暂态、有形布置，其包括存储介质，诸如硬盘；任何其他类型的盘，包括软盘、光盘、紧致盘只读存储器(CD-ROM)、可重写紧致盘(CD-RW)以及磁光盘；半导体器件，诸如，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(EPROM)、闪存、电可擦除可编程只读存储器(EEPROM)；相变存储器(PCM)；磁卡或光卡；或适于存储电子指令的任何其他类型的介质。

[0314] 因此，实施例还包括非暂态的有形机器可读介质，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/或系统特征。这些实施例也被称为程序产品。

仿真(包括二进制变换、代码变形等)

[0315] 在一些情况下，指令转换器可用于将指令从源指令集转换至目标指令集。例如，指令转换器可以将指令变换(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、仿真或以其他方式转换成要由核处理的一条或多条其他指令。指令转换器可以用软件、硬件、固件、或其组合来实现。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且部分在处理器外。

[0316] 图35是根据实施例的对照使用软件指令转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。在所图示的实施例中，指令转换器是软件指令转换器，但替代地，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35示出可使用x86编译器3504来编译高级语言3502形式的程序，以生成可由具有至少一个x86指令集核的处理器3516原生执行的x86二进制代码3506。具有至少一个x86指令集核的处理器3516表示通过兼容地执行或以其他方式处理以下各项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的任何处理器：1)英特尔x86指令集核的指令集的实质部分，或2)目标为在具有至少一个x86指令集核的英特尔处理器上运行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的应用或其他软件的目标代码版本。x86编译器3504表示可操作用于生成x86二进制代码3506(例如，目标代码)的编译器，该二进制代码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3516上执行。类似地，图35示出可以使用替代的指令集编译器3508来编译高级语言3502形式的程序，以生成可以由不具有至少一个x86指令集核的处理器3514(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)原生执行的替代的指令集二进制代码3510。指令转换器3512用于将x86二进制代码3506转换成可以由不具有x86指令集核的处理器3514原生执行的代码。该转换后的代码不大可能与替代的指令集二进制代码3510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，并且由来自替代指令集的指令构成。因此，指令转换器3512通过仿真、模拟或任何其他过程来表示允许不具有x86指令集处理器或核的

处理器或其他电子设备执行x86二进制代码3506的软件、固件、硬件或其组合。

进一步的示例

[0317] 示例1提供一种示例性处理器，包括：取出电路，用于取出具有格式的指令，该格式具有用于指定操作码、源矩阵的位置和目的地矩阵的位置的字段，其中操作码指示处理器用于将所指定的源矩阵变换为具有行交错(RowInt)格式的所指定的目的地矩阵；解码电路，用于对所取出的指令进行解码；以及执行电路，用于通过经由以下操作将所指定的源矩阵变换为所指定的RowInt格式化的目的地矩阵来对经解码的指令作出响应：以行为主的顺序或列为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵，K宽度子矩阵具有K列和足够的行以保存J个元素。

[0318] 示例2包括示例1的示例性处理器的实质内容，其中所指定的源矩阵和目的地矩阵中的每一个包括向量寄存器的集合、片寄存器的集合、以及存储器位置的集合中的一个。

[0319] 示例3包括示例1的示例性处理器的实质内容，其中指令格式进一步包括用于指定J和K的字段。

[0320] 示例4包括示例1的示例性处理器的实质内容，其中指令格式进一步包括用于指定所指定的源目的地矩阵的元素尺寸和目的地矩阵的元素尺寸的字段，所指定的元素尺寸包括二元数位、半字节、字节、字、双字和四字中的一个。

[0321] 示例5包括示例1的示例性处理器的实质内容，其中指令格式进一步包括用于指定所指定的源矩阵的元素格式和目的地矩阵的元素格式的字段，所指定的元素格式包括紧缩或标量单精度浮点数据、或紧缩或标量双精度浮点数据、以及紧缩或标量整数数据。

[0322] 示例6包括示例1的示例性处理器的实质内容，其中所指定的源矩阵包括M x N元素数组，并且所指定的目的地矩阵包括一半的行和两倍的列，其中指令的格式进一步用于指定M和N中的至少一个。

[0323] 示例7包括示例1的示例性处理器的实质内容，其中指令格式进一步包括用于指定掩码的字段，所指定的掩码是针对每个目的地元素具有一个位的多位的值，该位用于控制目的地元素是否将被更新，或用于控制目的地元素将被归零还是合并。

[0324] 示例8提供一种由处理器执行的示例性方法，该方法包括：使用取出电路取出具有格式的指令，该格式具有用于指定操作码、源矩阵的位置和目的地矩阵的位置的字段，其中操作码指示处理器用于将所指定的源矩阵变换为具有行交错(RowInt)格式的所指定的目的地矩阵；使用解码电路对所取出的指令进行解码；以及使用执行电路通过经由以下操作将所指定的源矩阵变换为所指定的RowInt格式化的目的地矩阵来对经解码的指令作出响应：以行为主的顺序或列为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵，K宽度子矩阵具有K列和足够的行以保存J个元素。

[0325] 示例9包括示例8的示例性方法的实质内容，其中所指定的源矩阵和目的地矩阵中的每一个包括向量寄存器的集合、片寄存器的集合、以及存储器位置的集合中的一个。

[0326] 示例10包括示例8的示例性方法的实质内容，其中J和K被附加的指令操作数中的一个或多个指定，被传递为操作码的前缀或后缀，被编码在立即数中，以及被已经由软件编程的控制寄存器指定。

[0327] 示例11包括示例8的示例性方法的实质内容，其中指令格式进一步包括用于指定所指定的源矩阵的元素尺寸和目的地矩阵的元素尺寸的字段，所指定的元素尺寸包括二元

数位、半字节、字节、字、双字和四字中的一个。

[0328] 示例12包括示例8的示例性方法的实质内容，其中指令格式进一步包括用于指定所指定的源矩阵的元素格式和目的地矩阵的元素格式的字段，所指定的元素格式包括紧缩或标量单精度浮点数据、或紧缩或标量双精度浮点数据、以及紧缩或标量整数数据。

[0329] 示例13包括示例8的示例性方法的实质内容，其中所指定的源矩阵用于包括M x N元素数组，并且所指定的目的地矩阵包括一半的行和两倍的列，其中指令的格式进一步用于指定M和N中的至少一个。

[0330] 示例14包括示例8的示例性方法的实质内容，其中指令格式进一步包括用于指定掩码的字段，所指定的掩码是针对每个目的地元素具有一个位的多位的值，该位用于控制目的地元素是否将被更新，或用于控制目的地元素将被归零还是合并。

[0331] 示例15提供一种包含指令的示例性非暂态机器可读介质，指令当被处理器执行时使处理器通过以下操作来作出响应：使用取出电路取出具有格式的指令，该格式具有用于指定操作码、源矩阵的位置和目的地矩阵的位置的字段，其中操作码指示处理器用于将所指定的源矩阵变换为具有行交错(RowInt)格式的所指定的目的地矩阵；使用解码电路对所取出的指令进行解码；以及使用执行电路通过经由以下操作将所指定的源矩阵变换为所指定的RowInt格式化的目的地矩阵来对经解码的指令作出响应：以行为主的顺序或列为主的顺序使所指定的源矩阵的每个J元素子列的J个元素交错为所指定的目的地矩阵的K宽度子矩阵，K宽度子矩阵具有K列和足够的行以保存J个元素。

[0332] 示例16包括示例15的示例性非暂态机器可读介质的实质内容，其中所指定的源矩阵和目的地矩阵中的每一个包括向量寄存器的集合、片寄存器的集合、以及存储器位置的集合中的一个。

[0333] 示例17包括示例15的示例性非暂态机器可读介质的实质内容，其中J和K被附加的指令操作数中的一个或多个指定，被传递为操作码的前缀或后缀，被编码在立即数中，以及被已经由软件编程的控制寄存器指定。

[0334] 示例18包括示例15的示例性非暂态机器可读介质的实质内容，其中指令格式进一步包括用于指定所指定的源矩阵的元素尺寸和目的地矩阵的元素尺寸的字段，所指定的元素尺寸包括二元数位、半字节、字节、字、双字和四字中的一个。

[0335] 示例19包括示例15的示例性非暂态机器可读介质的实质内容，其中指令格式进一步包括用于指定所指定的源矩阵的元素格式和目的地矩阵的元素格式的字段，所指定的元素格式包括紧缩或标量单精度浮点数据、或紧缩或标量双精度浮点数据、以及紧缩或标量整数数据。

[0336] 示例20包括示例15的示例性非暂态机器可读介质的实质内容，其中所指定的源矩阵包括M x N元素数组，并且所指定的目的地矩阵包括一半的行和两倍的列，其中指令的格式进一步用于指定M和N中的至少一个。

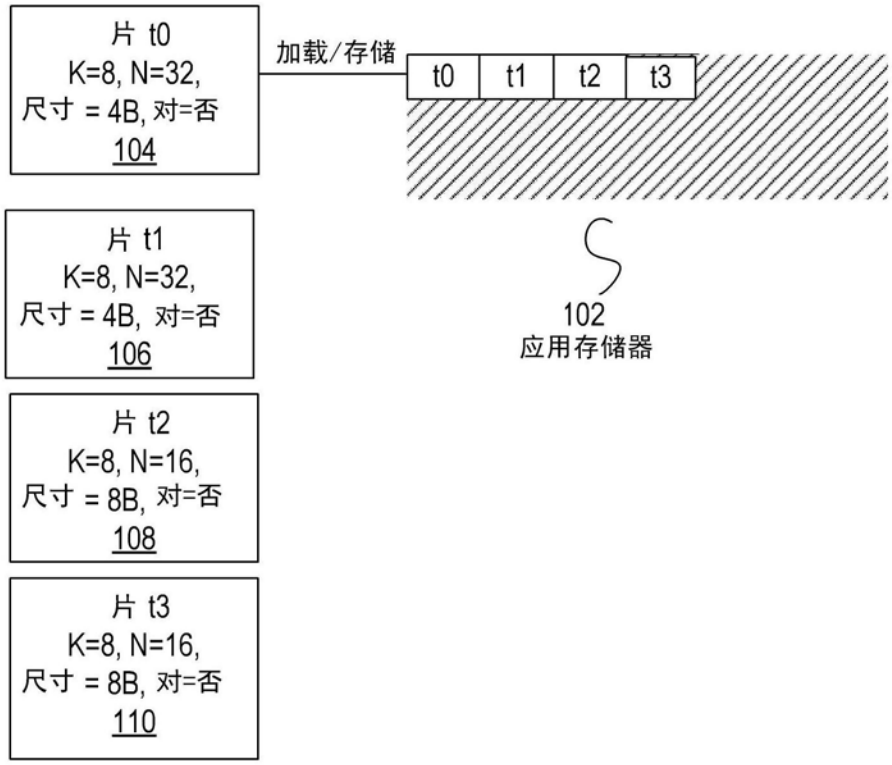

片 t0
K=8, N=32,
尺寸 = 4B, 对=否
104
加载/存储
t0
t1
t2
t3
片 t1
K=8, N=32,
尺寸 = 4B, 对=否
106
102
应用存储器
片 t2
K=8, N=16,
尺寸 = 8B, 对=否
108
片 t3
K=8, N=16,
尺寸 = 8B, 对=否
110

图1A

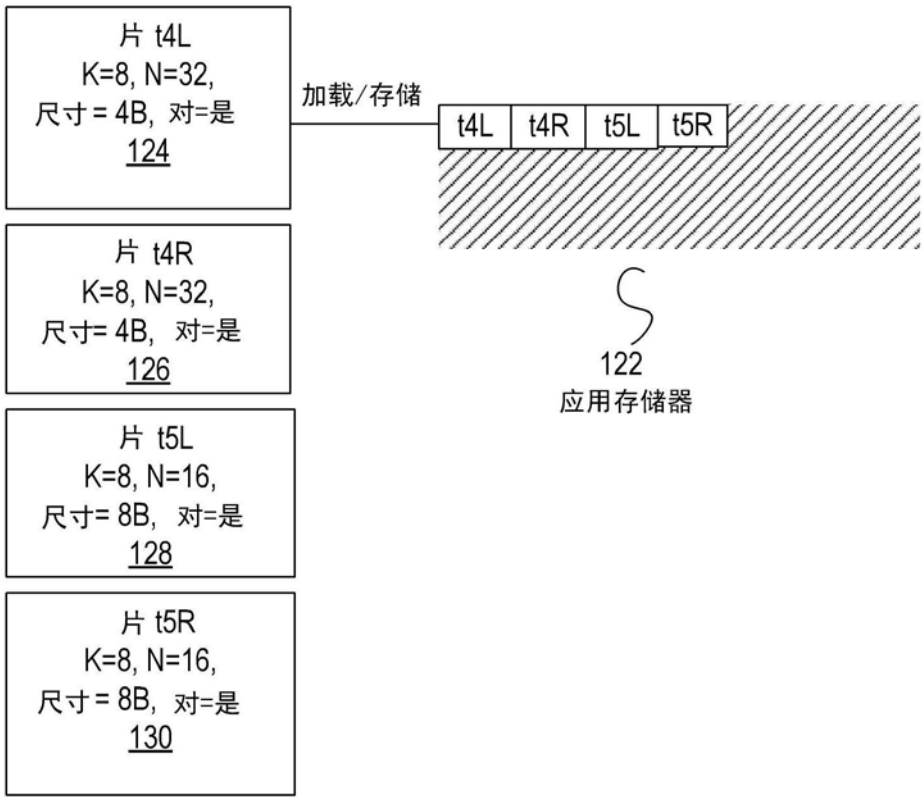

片 t4L
K=8, N=32,
尺寸 = 4B, 对=是
124
加载/存储
t4L
t4R
t5L
t5R
片 t4R
K=8, N=32,
尺寸= 4B, 对=是
126
122
应用存储器
片 t5L
K=8, N=16,
尺寸= 8B, 对=是
128
片 t5R
K=8, N=16,
尺寸 = 8B, 对=是
130

图1B

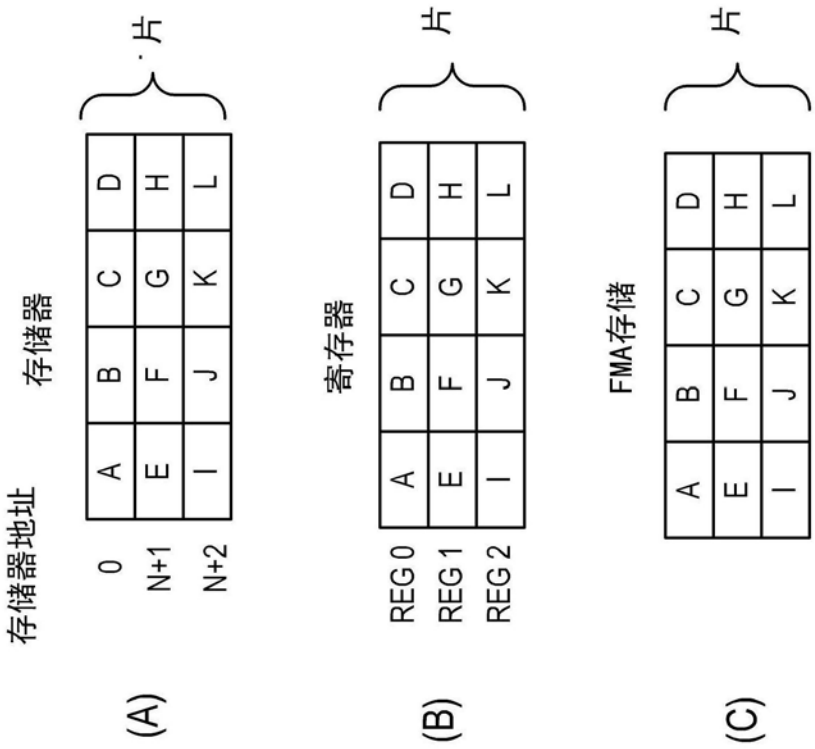


图2

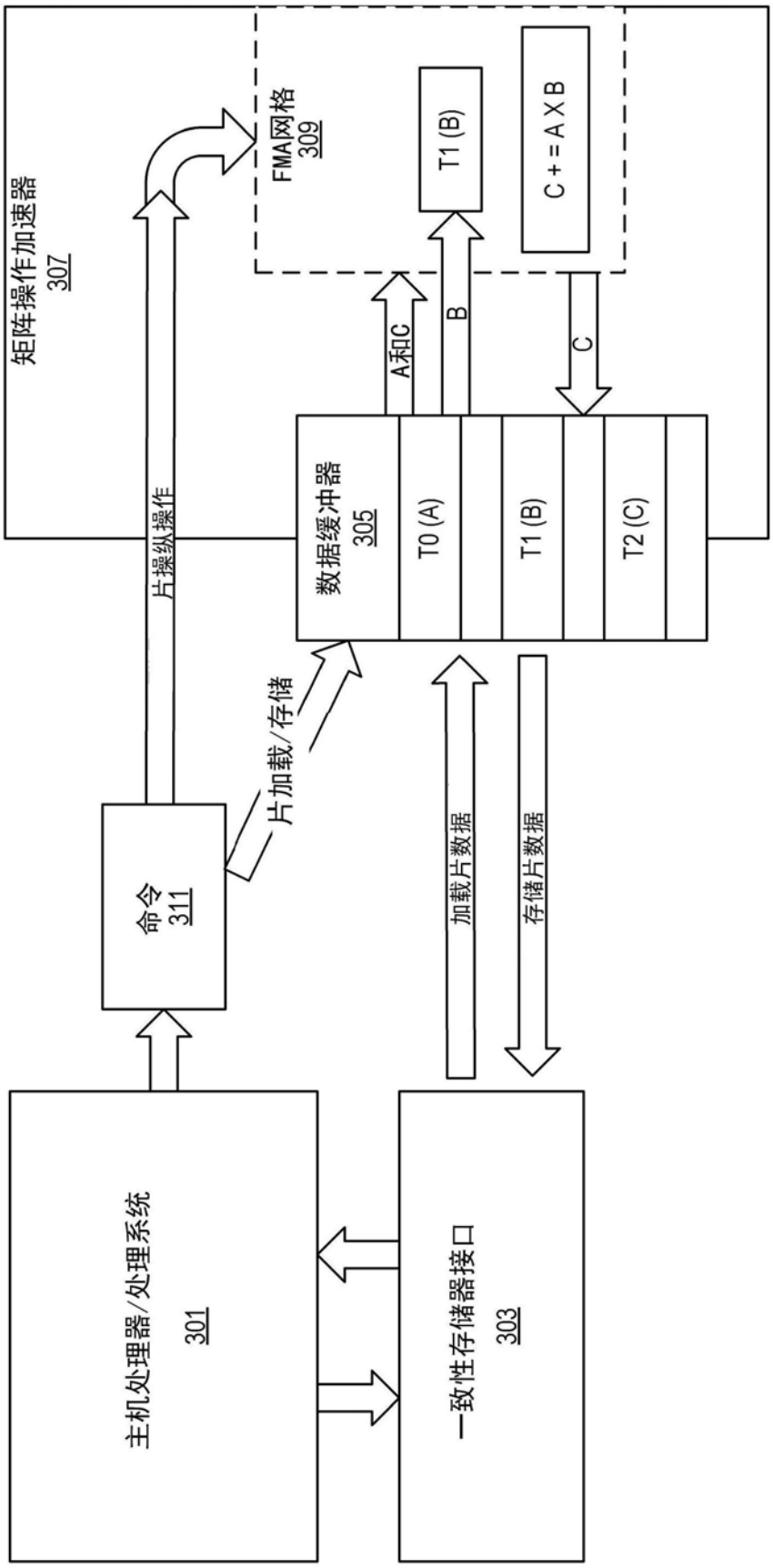

矩阵操作加速器
307
FMA网格
309
T1 (B)
C + = A X B
A和C
B
C
数据缓冲器
305
T0 (A)
T1 (B)
T2 (C)
片操纵操作
片加载/存储
命令
311
加载片数据
存储片数据
主机处理器/处理系统
301
一致性存储器接口
303

图3

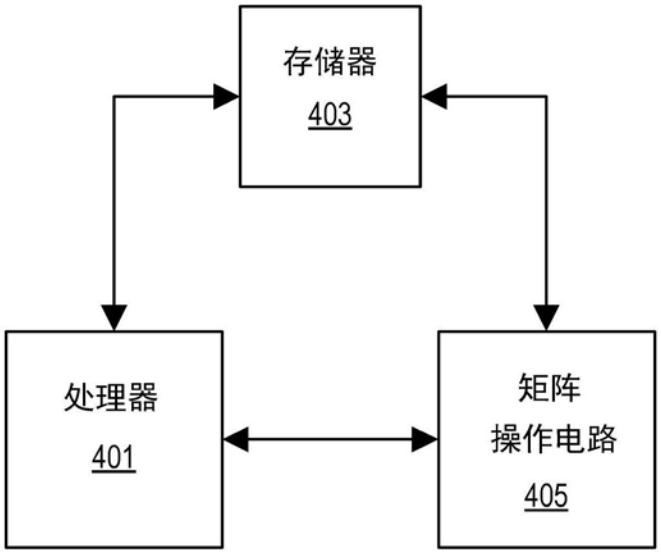

存储器
403
处理器
401
矩阵
操作电路
405

图4

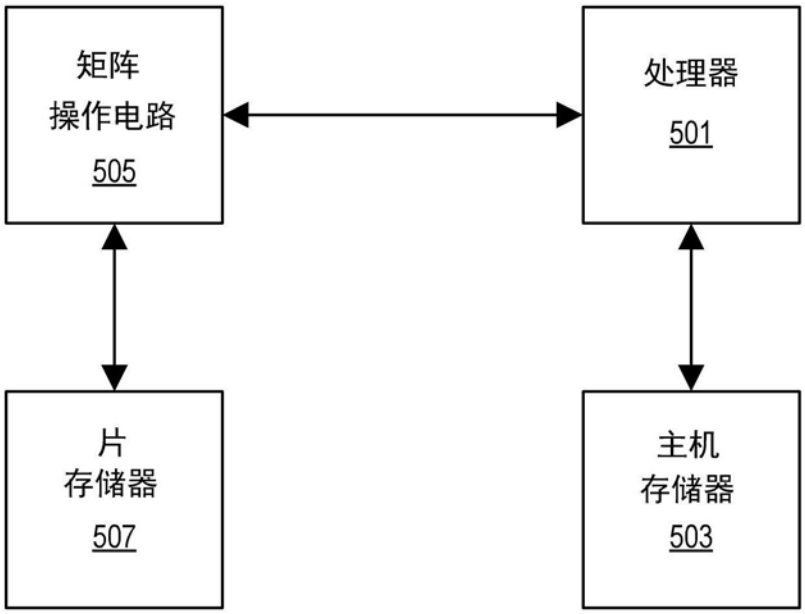

矩阵
操作电路
505
处理器
501
片
存储器
507
主机
存储器
503

图5

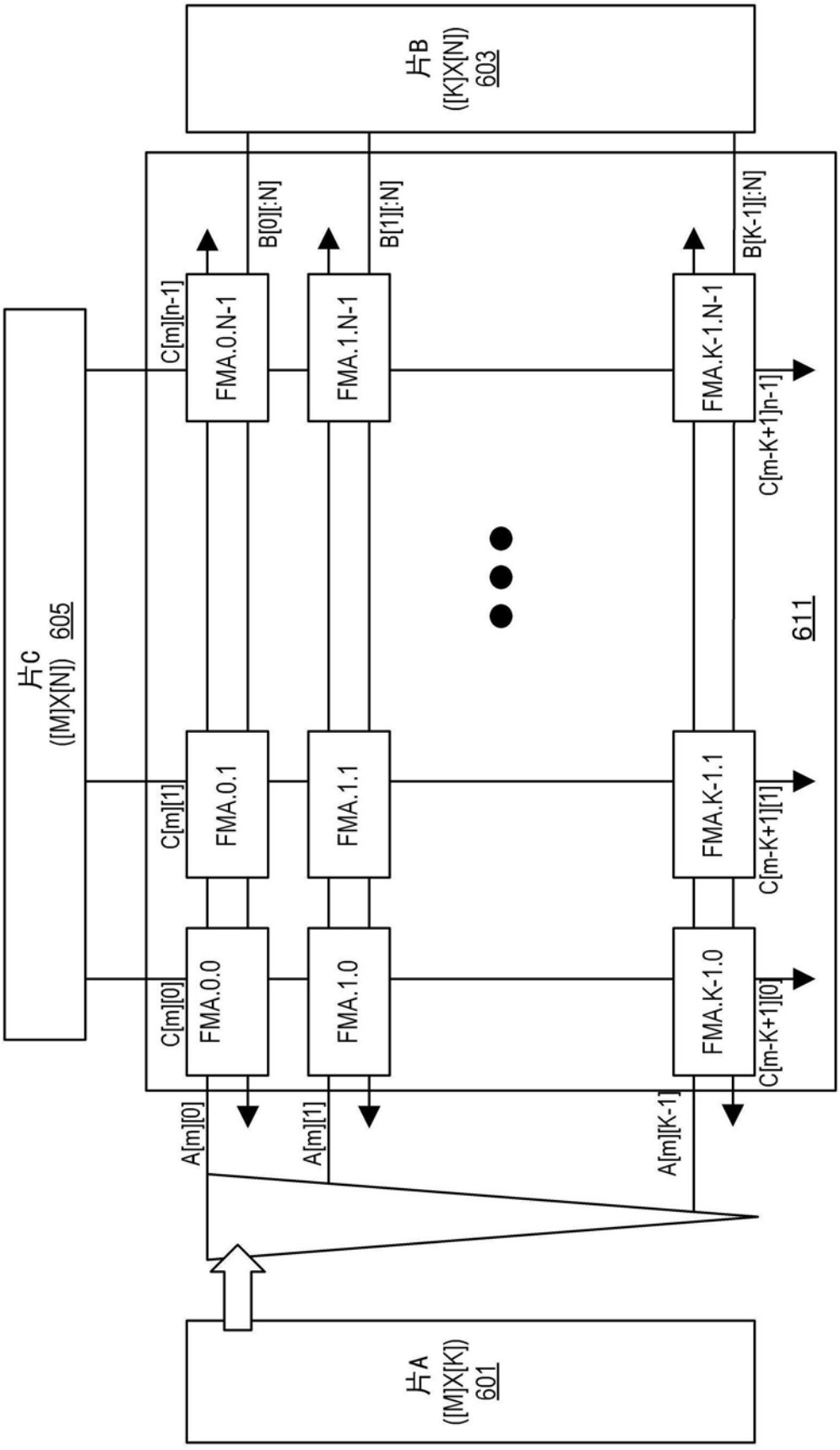

片B
([K]X[N])
603
B[0][:N]
B[1][:N]
B[K-1][:N]
C[m][n-1]
FMA.0.N-1
FMA.1.N-1
FMA.K-1.N-1
C[m-K+1][n-1]
片C
([M]X[N]) 605
611
C[m][1]
FMA.0.1
FMA.1.1
FMA.K-1.1
C[m-K+1][1]
C[m][0]
FMA.0.0
FMA.1.0
FMA.K-1.0
C[m-K+1][0]
A[m][0]
A[m][1]
A[m][K-1]
片A
([M]X[K])
601

图6

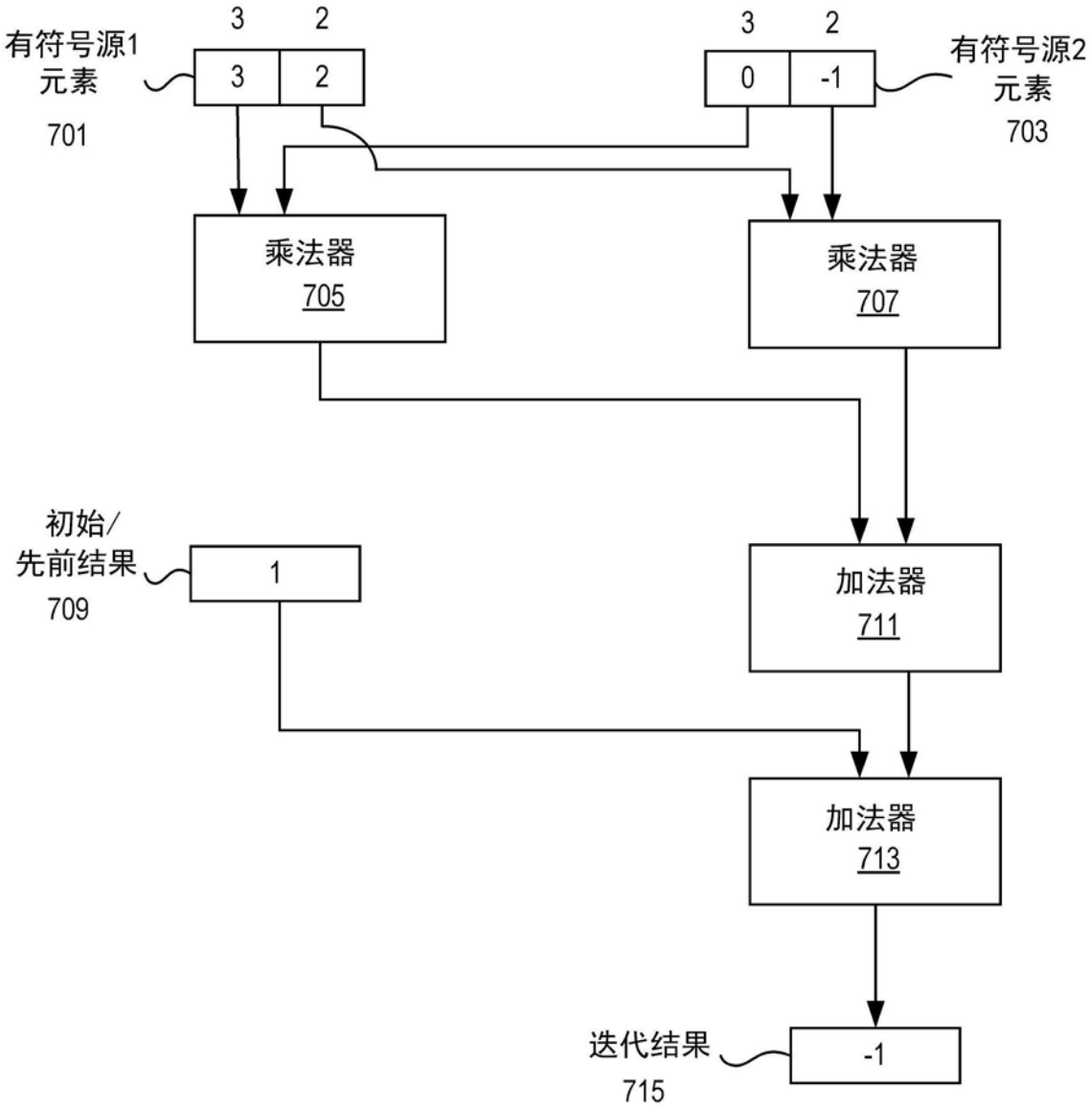

有符号源1
元素
701
3
2
3
2
有符号源2
元素
703
3
2
0
-1
乘法器
705
乘法器
707
初始/
先前结果
709
1
加法器
711
加法器
713
迭代结果
715
-1

图7

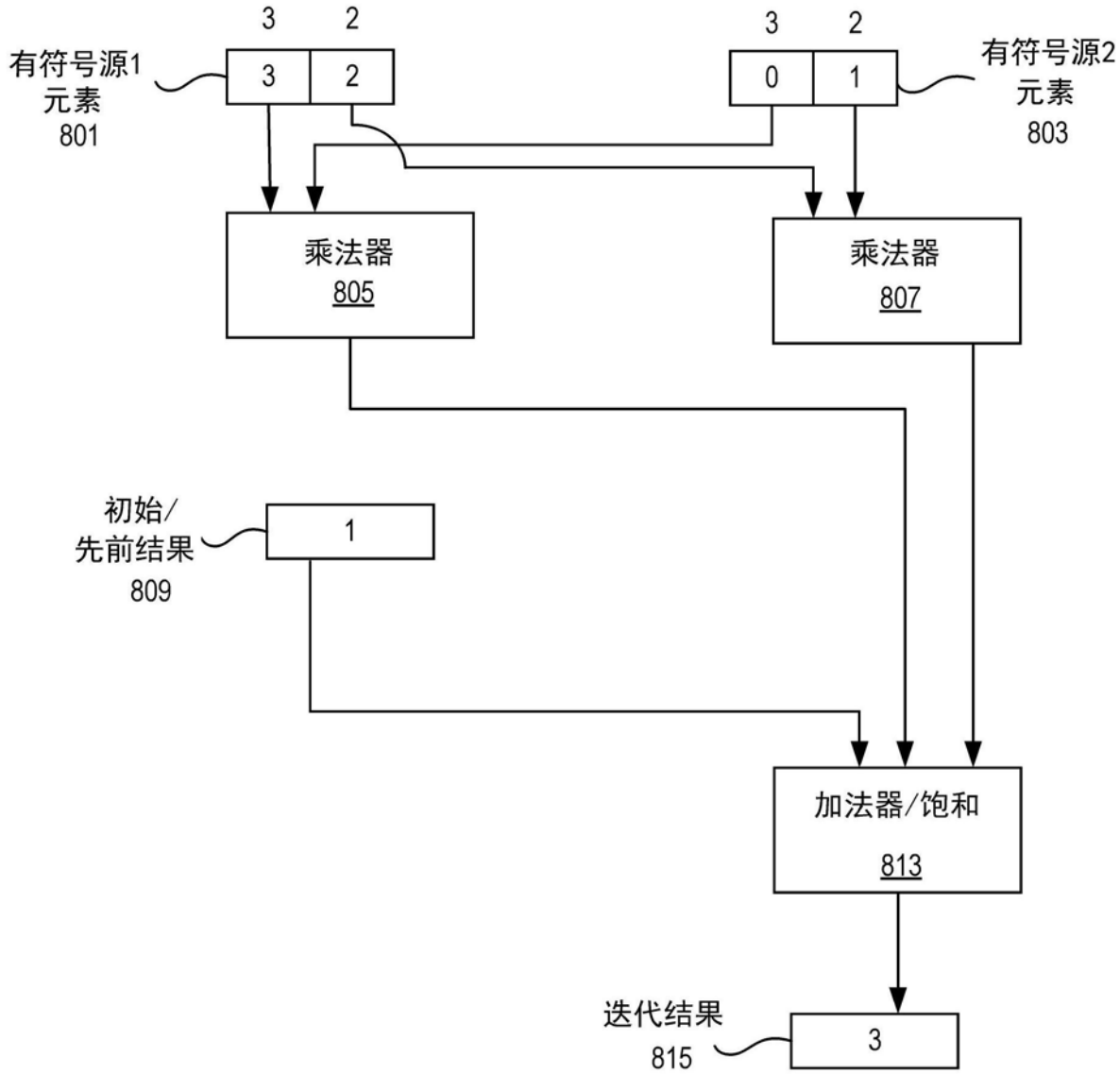

3
2
有符号源1
元素
801
3
2
3
2
0
1
有符号源2
元素
803
乘法器
805
乘法器
807
初始/
先前结果
809
1
加法器/饱和
813
迭代结果
815
3

图8

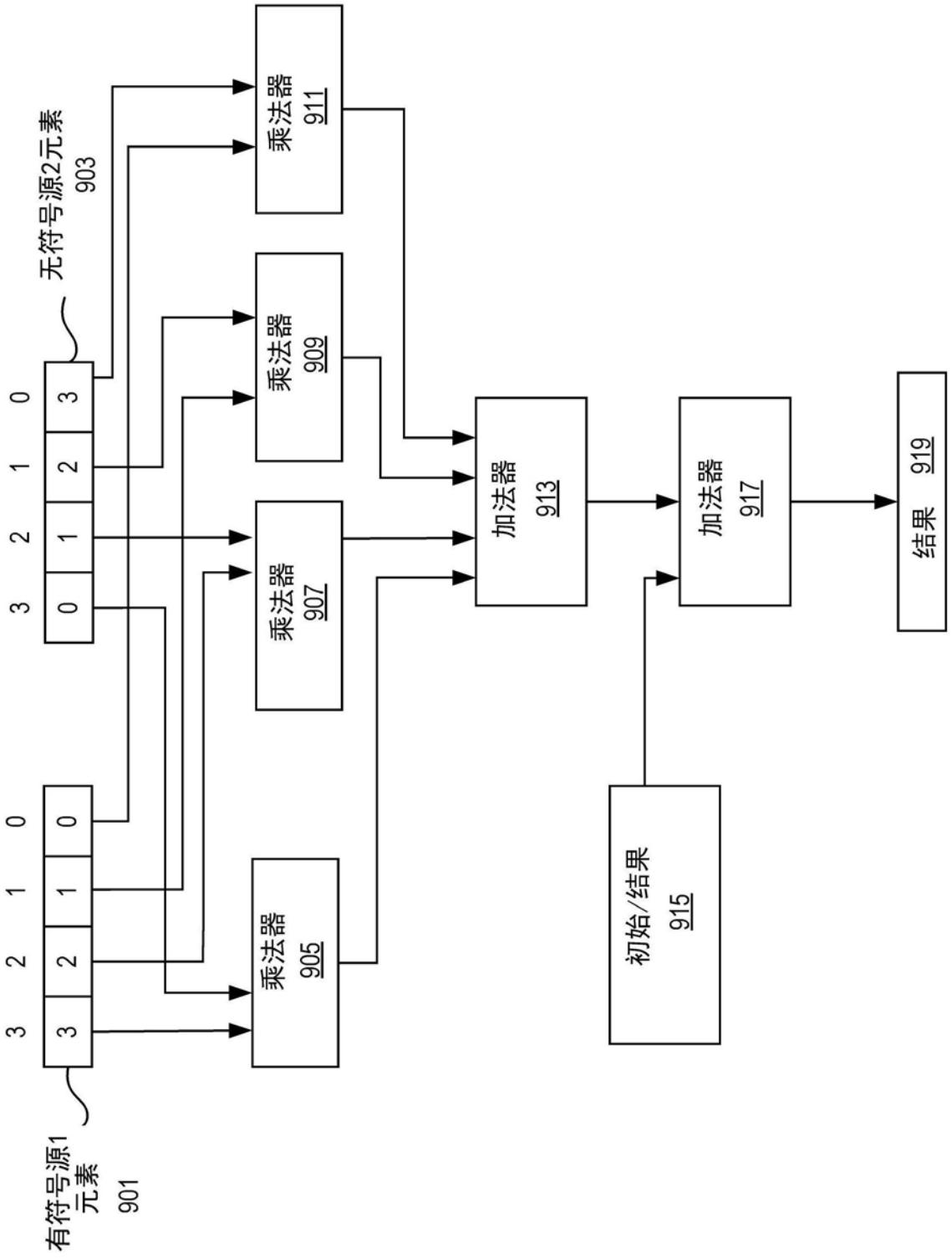

有符号源1元素
901
无符号源2元素
903
3 2 1 0
乘法器
905
乘法器
907
乘法器
909
乘法器
911
初始/结果
915
加法器
913
加法器
917
结果 919

图9

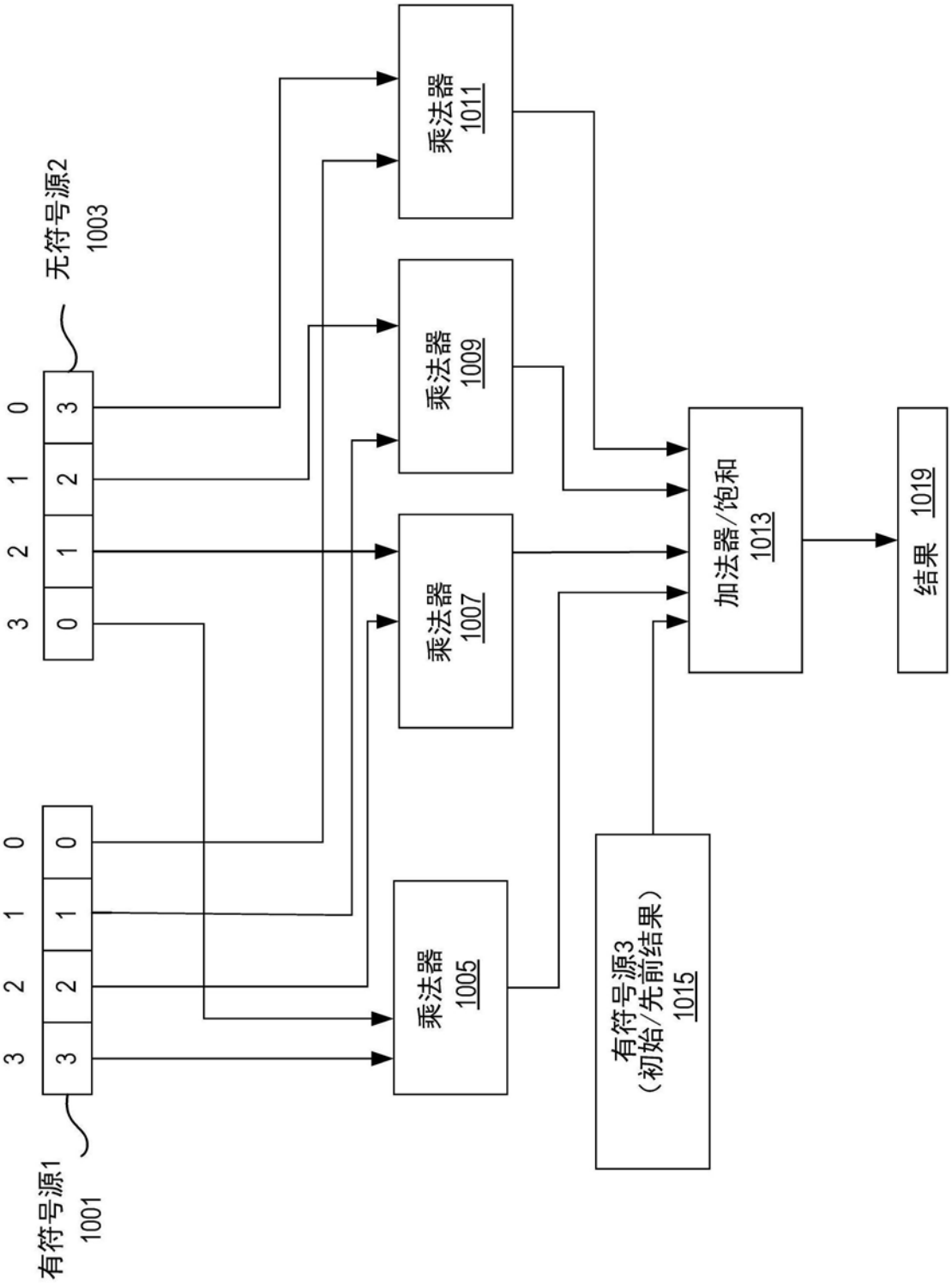

乘法器
1011
乘法器
1009
乘法器
1007
乘法器
1005
无符号源2
1003
有符号源1
1001
3 2 1 0
3 2 1 0
加法器/饱和
1013
结果 1019
有符号源3
(初始/先前结果)
1015

图10

累加器2X输入尺寸 1101

源	位	累加器	位
字节	8	字/HPFP	16
字	16	整数32/SPFP	32
SPFP/整数32	32	整数64/DPFP	64

累加器4X输入尺寸 1103

源	位	累加器	位
字节	8	整数32/SPFP	32
字	16	整数64/DPFP	64

累加器8X输入尺寸 1105

源	位	累加器	位
字节	8	整数64/DPFP	64

图11

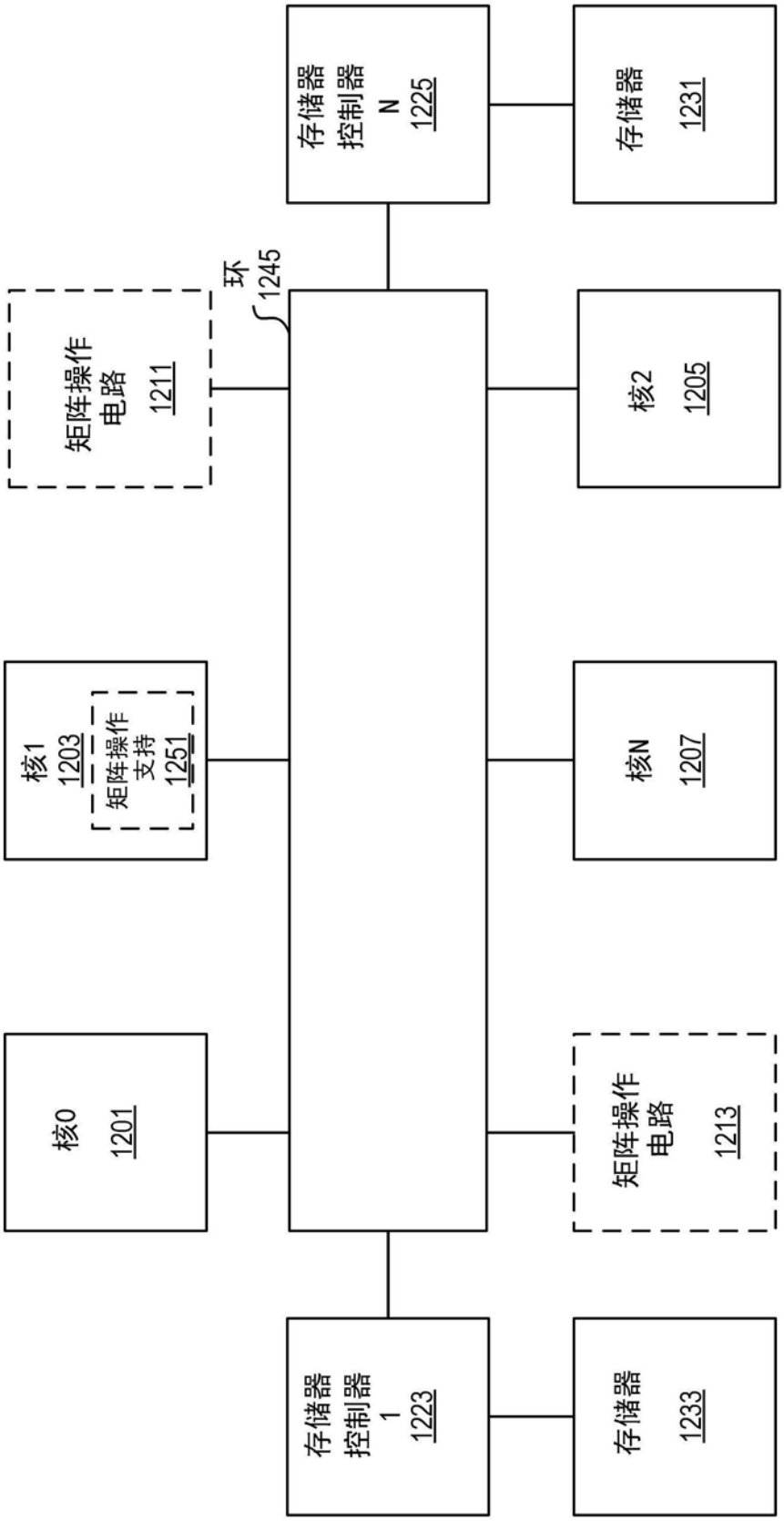

存储器控制器N 1225
存储器 1231
环 1245
矩阵操作电路 1211
核2 1205
核1 1203
矩阵操作支持 1251
核N 1207
核0 1201
矩阵操作电路 1213
存储器控制器1 1223
存储器 1233

图12

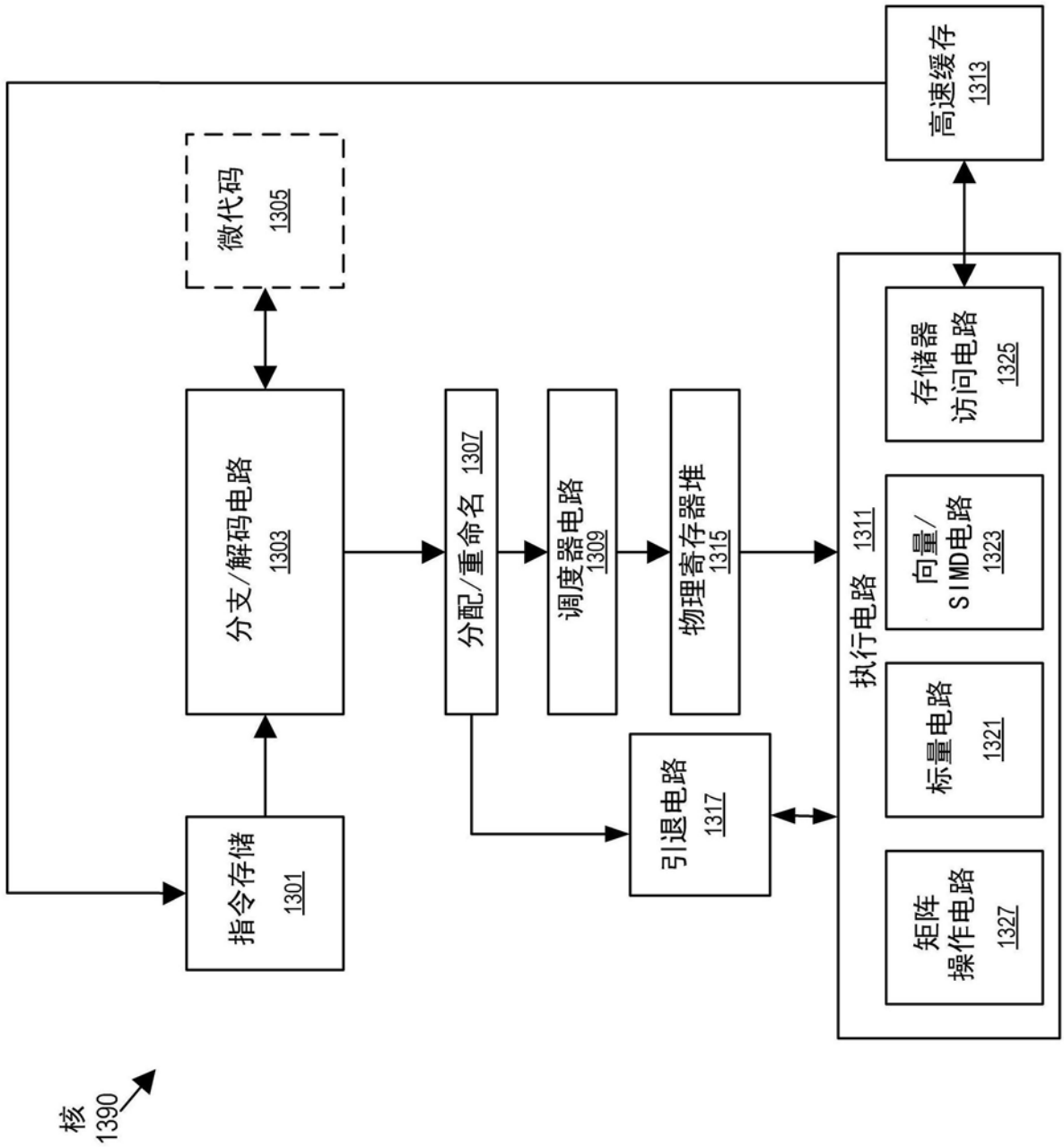

高速缓存 1313
微代码 1305
分支/解码电路 1303
分配/重命名 1307
调度器电路 1309
物理寄存器堆 1315
执行电路 1311
存储器访问电路 1325
向量/SIMD电路 1323
标量电路 1321
矩阵操作电路 1327
引退电路 1317
指令存储 1301
核 1390

图13

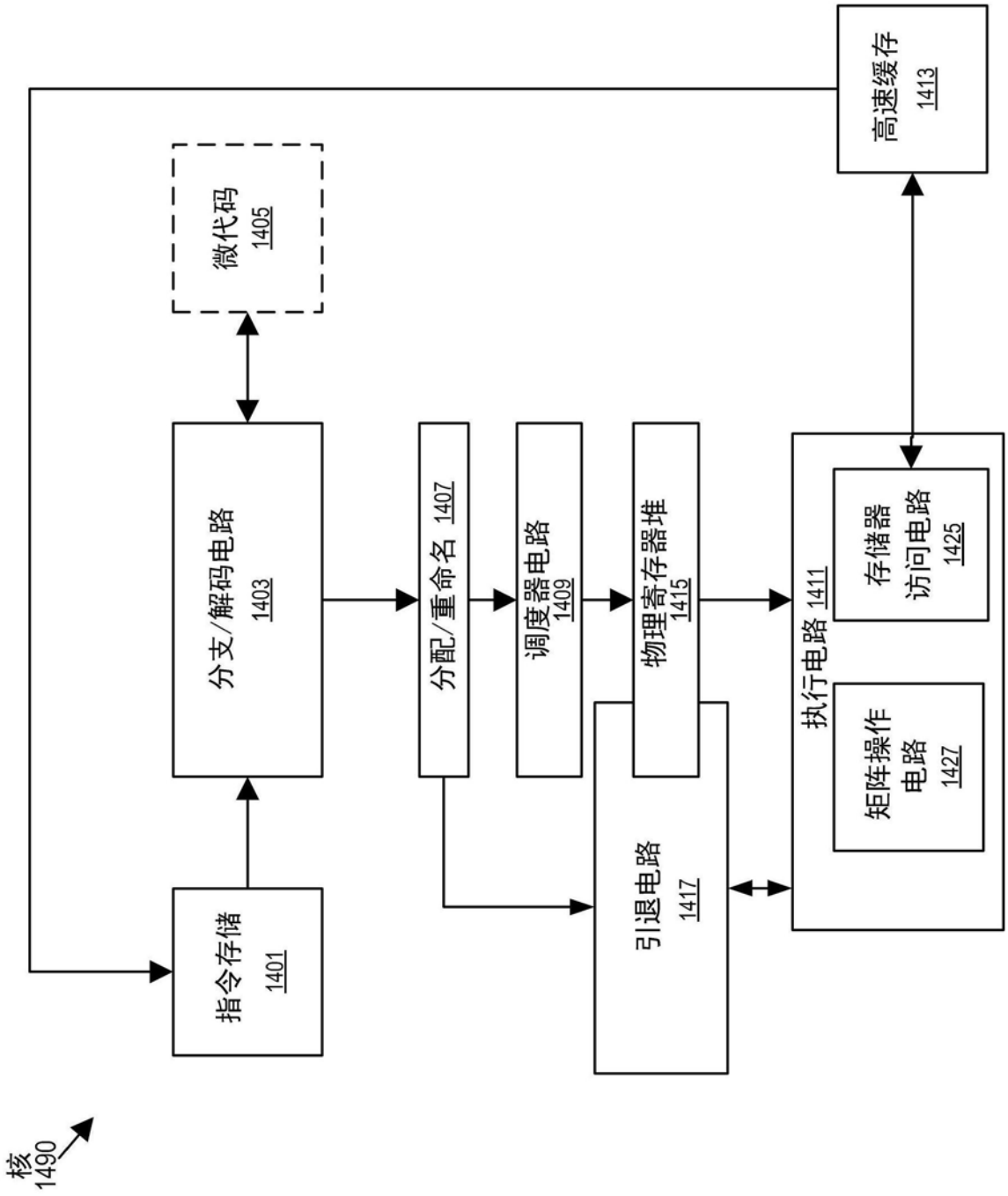

高速缓存
1413
微代码
1405
分支/解码电路
1403
分配/重命名 1407
调度器电路
1409
物理寄存器堆
1415
执行电路 1411
存储器
访问电路
1425
矩阵操作
电路
1427
引退电路
1417
指令存储
1401
核
1490

图14

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{bmatrix}$$

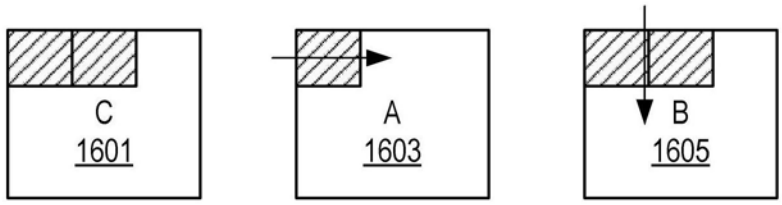
地址	值
0	A_{11}
1	A_{12}
2	A_{13}
3	A_{21}
4	A_{22}
5	A_{23}

行为主

地址	值
0	A_{11}
1	A_{21}
2	A_{12}
3	A_{22}
4	A_{13}
5	A_{23}

列为主

图15




```
TILECONFIG [RAX]
//假定一些外循环驱动高速缓存分片（未示出）
{
TILELOAD TMM0, RSI+RDI  //SRCDST，RSI指向C，RDI具有
TILELOAD TMM1, RSI+RDI+N //C的第二片，在SIMD尺度N中展开
MOV KK, 0
LOOP:
TILELOAD TMM2, R8+R9  //SRC2是A的跨步式加载，重新用于2条TMMA指令。
TILELOAD TMM3, R10+R11 //SRC1是B的跨步式加载
TILEMULTIPLY TMM0, TMM2,  //更新C的左片
TILELOAD TMM3, R10+R11+N  //SRC1加载有来自下一最右片的B
TILEMULTIPLY TMM1, TMM2,  //更新C的右片
ADD R8, K //用循环外部已知的常数更新指针
ADD R10, K*R11
ADD KK, K
CMP KK, LIMIT
JNE LOOP
TILESTORE RSI+RDI, TMM0 //更新存储器中的C矩阵
TILESTORE RSI+RDI+M, TMM1
} //外循环的结尾
TILERELEASE //将片返回至初始状态
```

图16

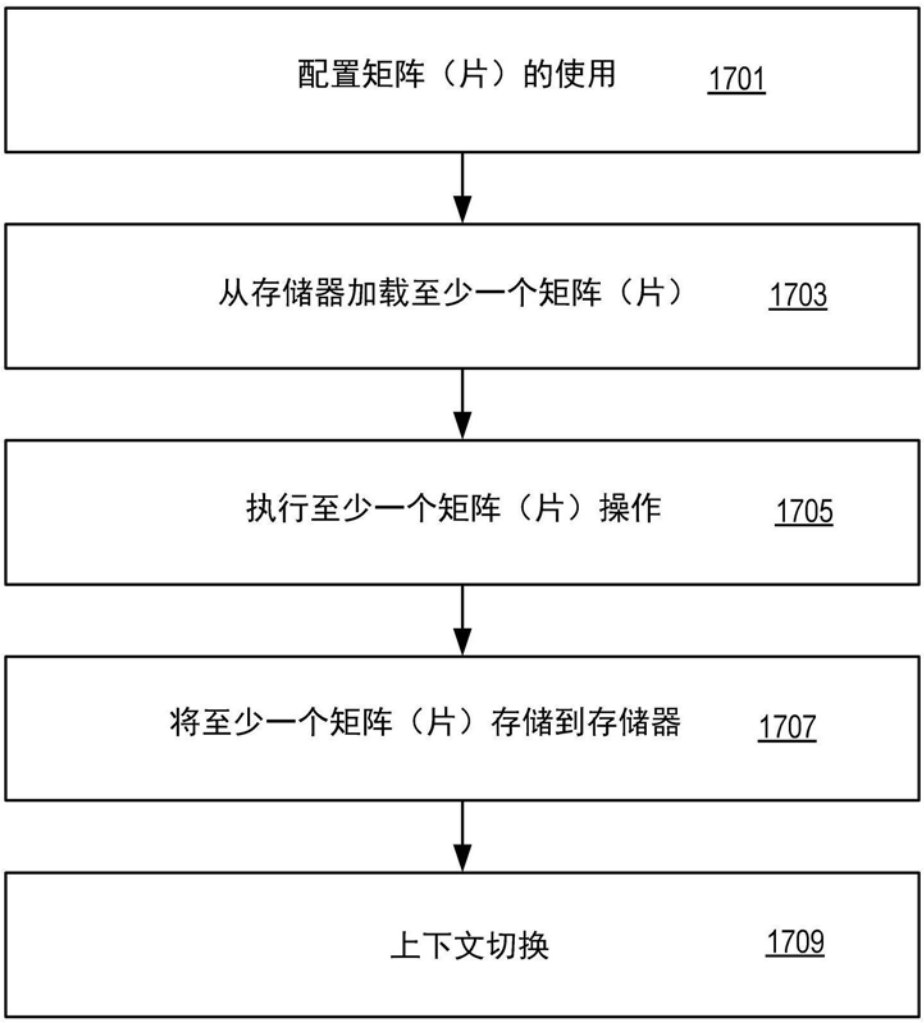

配置矩阵（片）的使用
1701
从存储器加载至少一个矩阵（片）
1703
执行至少一个矩阵（片）操作
1705
将至少一个矩阵（片）存储到存储器
1707
上下文切换
1709

图17

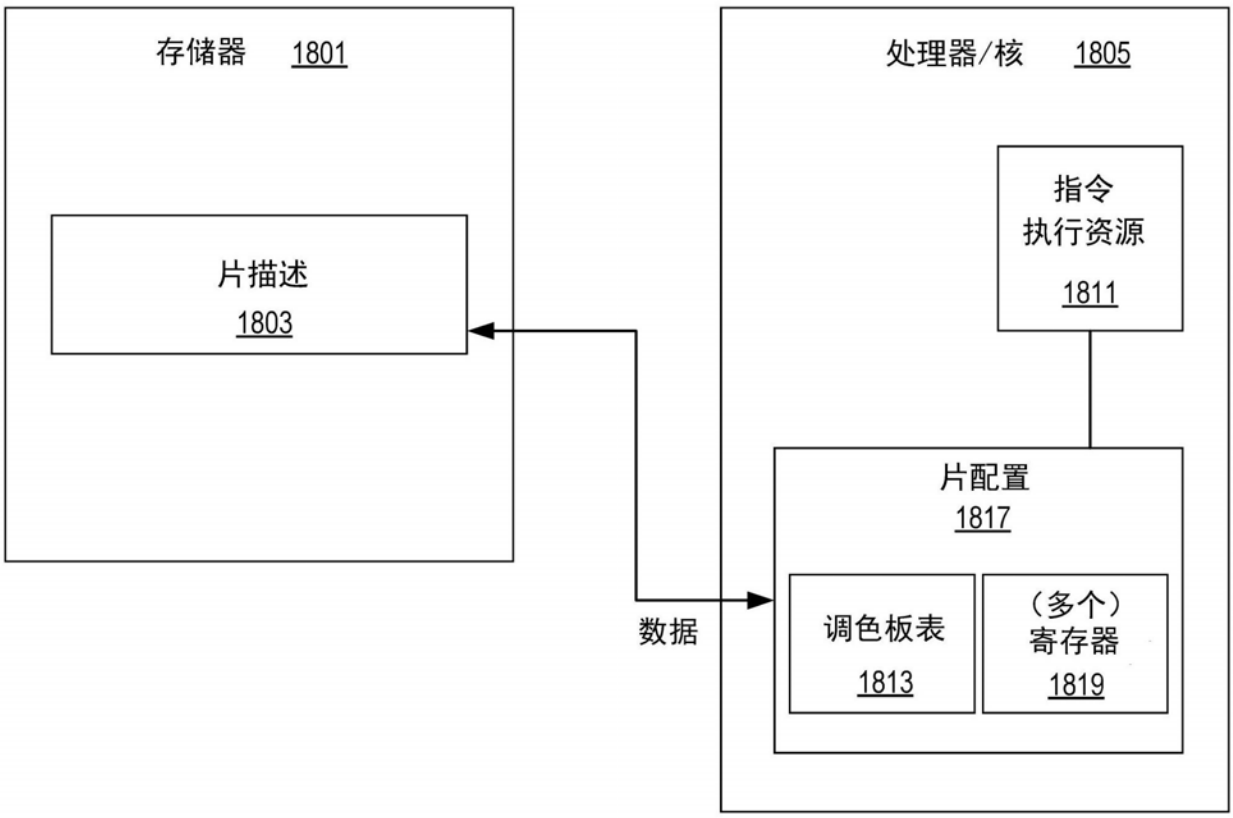

存储器 1801
片描述
1803
处理器/核 1805
指令
执行资源
1811
片配置
1817
调色板表
1813
(多个)
寄存器
1819
数据

图18

调色板ID 1901	STARTM 1903
STARTP 1905	对指示符 1907
0	0
0	0

...

0	0
TMM0 行 1913	TMM0 列 1915
TMM1 行	TMM1 列
▪ ▪ ▪	
TMM15 行	TMM15 列
0	

图19

图20(A)

图20(B)

图20(C)

图20(D)

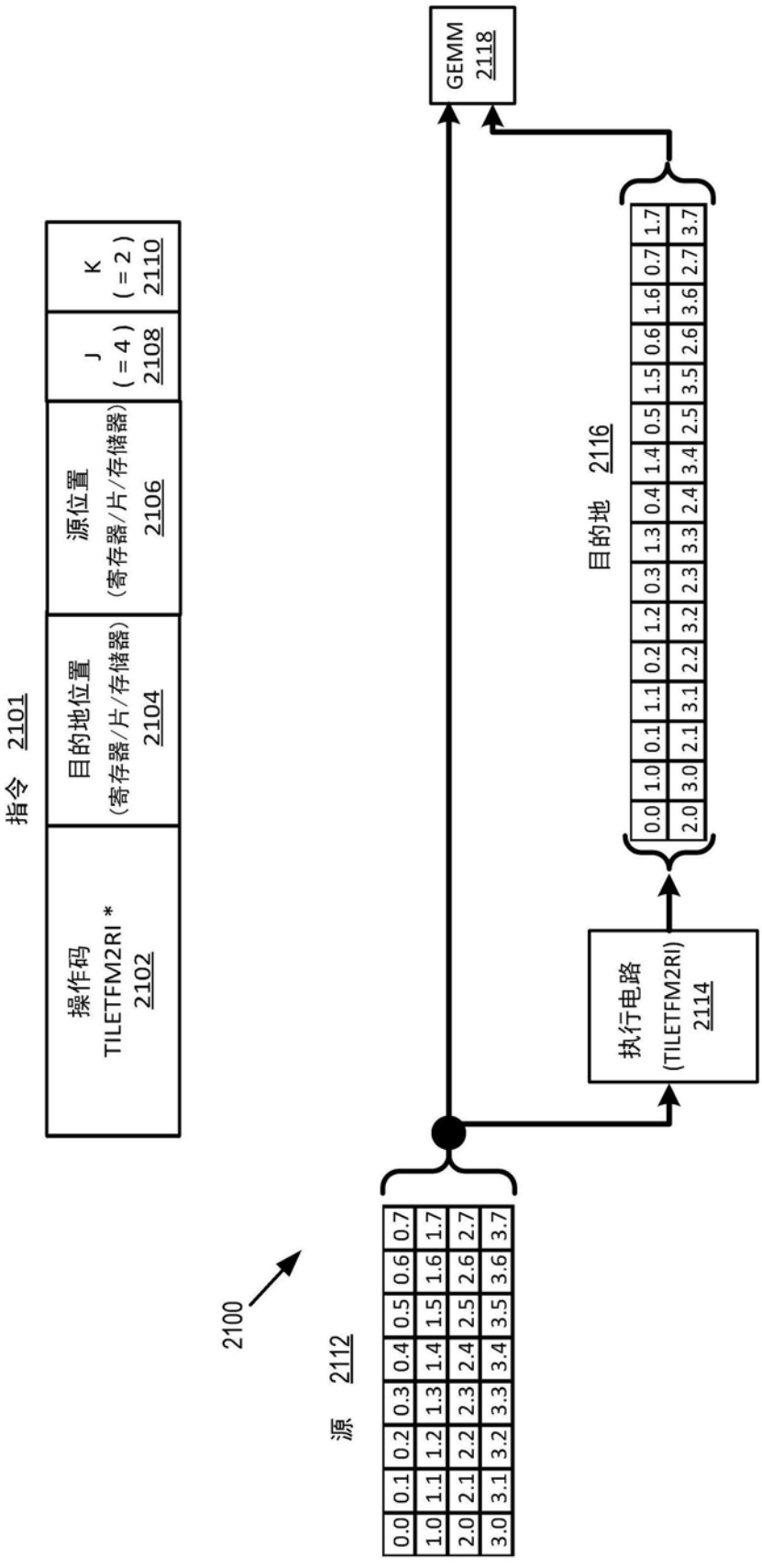

指令 2101
操作码 TILETFM2RI * 2102
目的地位置 (寄存器/片/存储器) 2104
源位置 (寄存器/片/存储器) 2106
J (=4) 2108
K (=2) 2110
2100
源 2112
目的地 2116
执行电路 (TILETFM2RI) 2114
GEMM 2118

图21

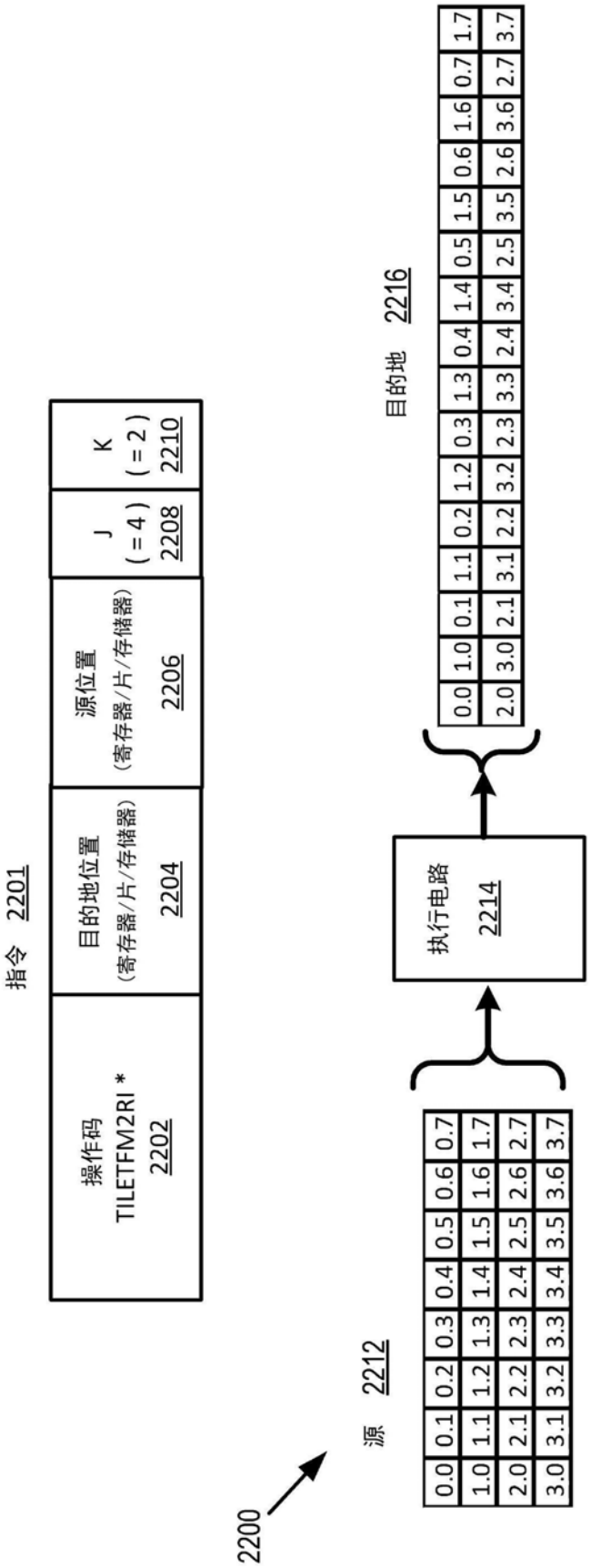

指令 2201
操作码
TILETFM2RI *
2202
目的地位置
(寄存器/片/存储器)
2204
源位置
(寄存器/片/存储器)
2206
J
(=4)
2208
K
(=2)
2210
2200
源 2212
目的地 2216
执行电路
2214

图22A

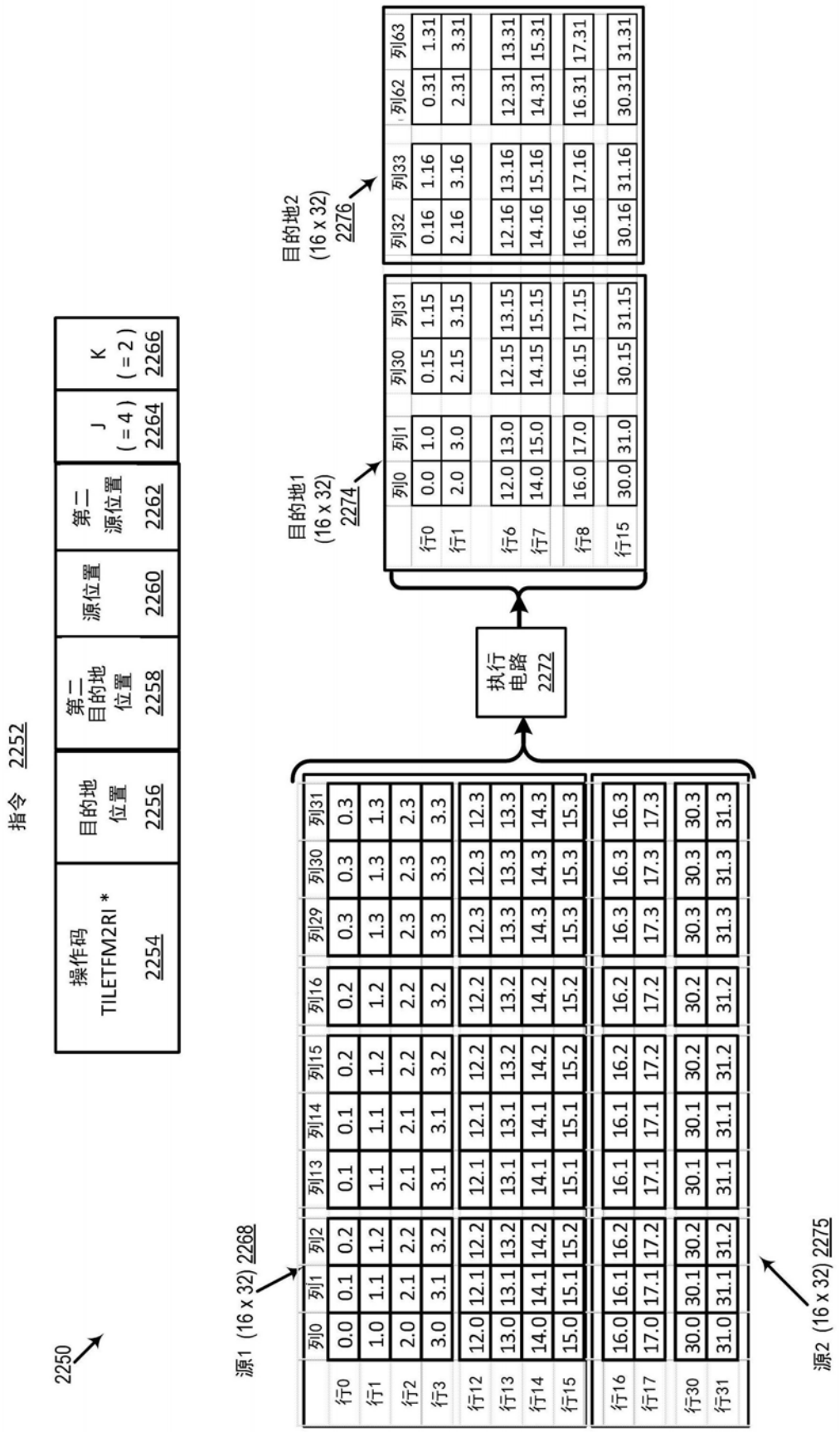

指令 2252
操作码 TILETFM2RI * 2254
目的地位置 2256
第二目的地位置 2258
源位置 2260
第二源位置 2262
J (=4) 2264
K (=2) 2266
2250
源1 (16 x 32) 2268
源2 (16 x 32) 2275
执行电路 2272
目的地1 (16 x 32) 2274
目的地2 (16 x 32) 2276

图22B

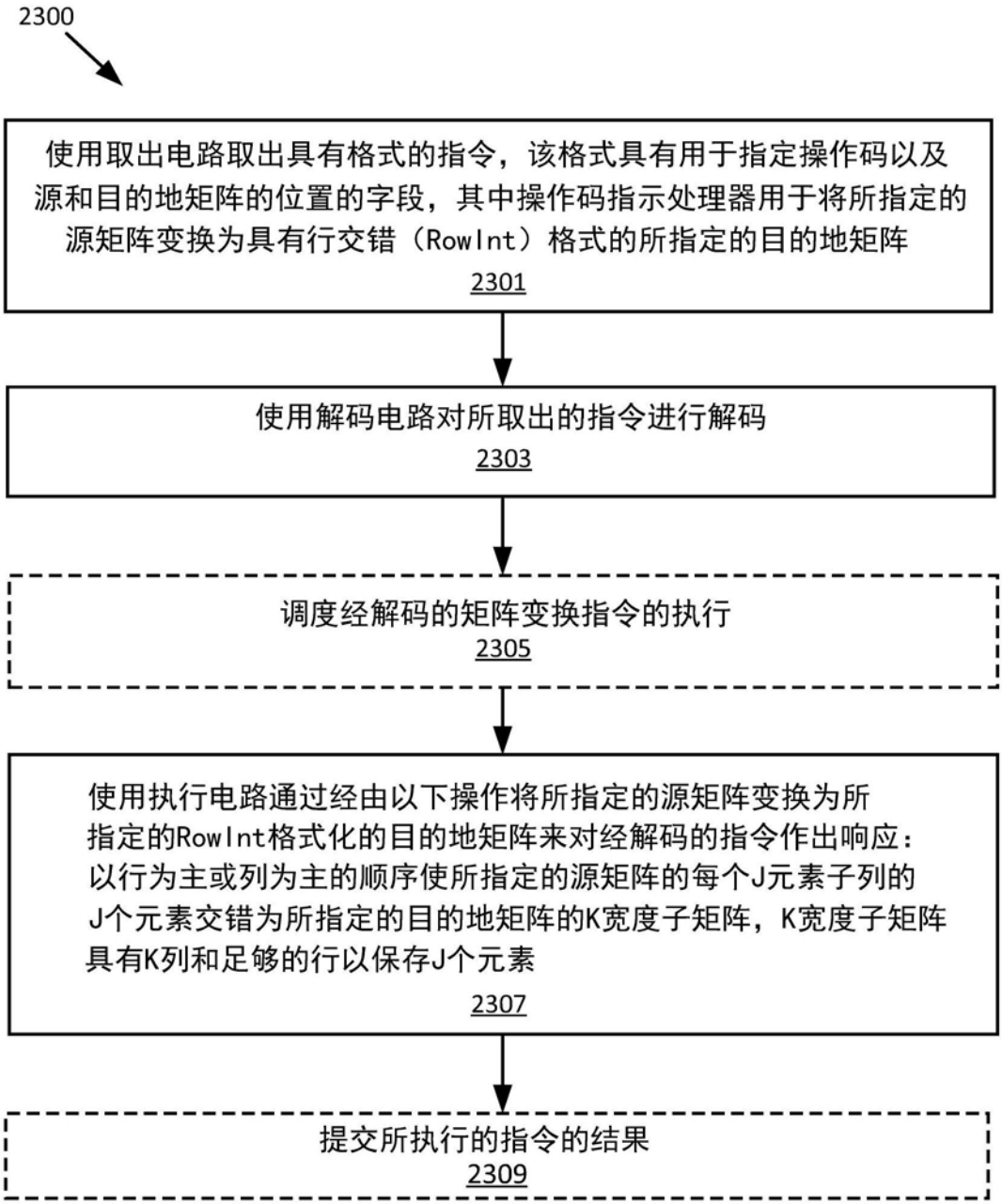


图23

TILETFM2RI 指令 2400

操作码 TILETFM2RI * 2402	目的地位置 2404	源位置 2406	第二目的地位置 2408	第二源位置 2410	J 2412	K 2414	元素尺寸 2416	元素格式 2418	M（源行） 2420	N（源列） 2422	掩码 2424

图24

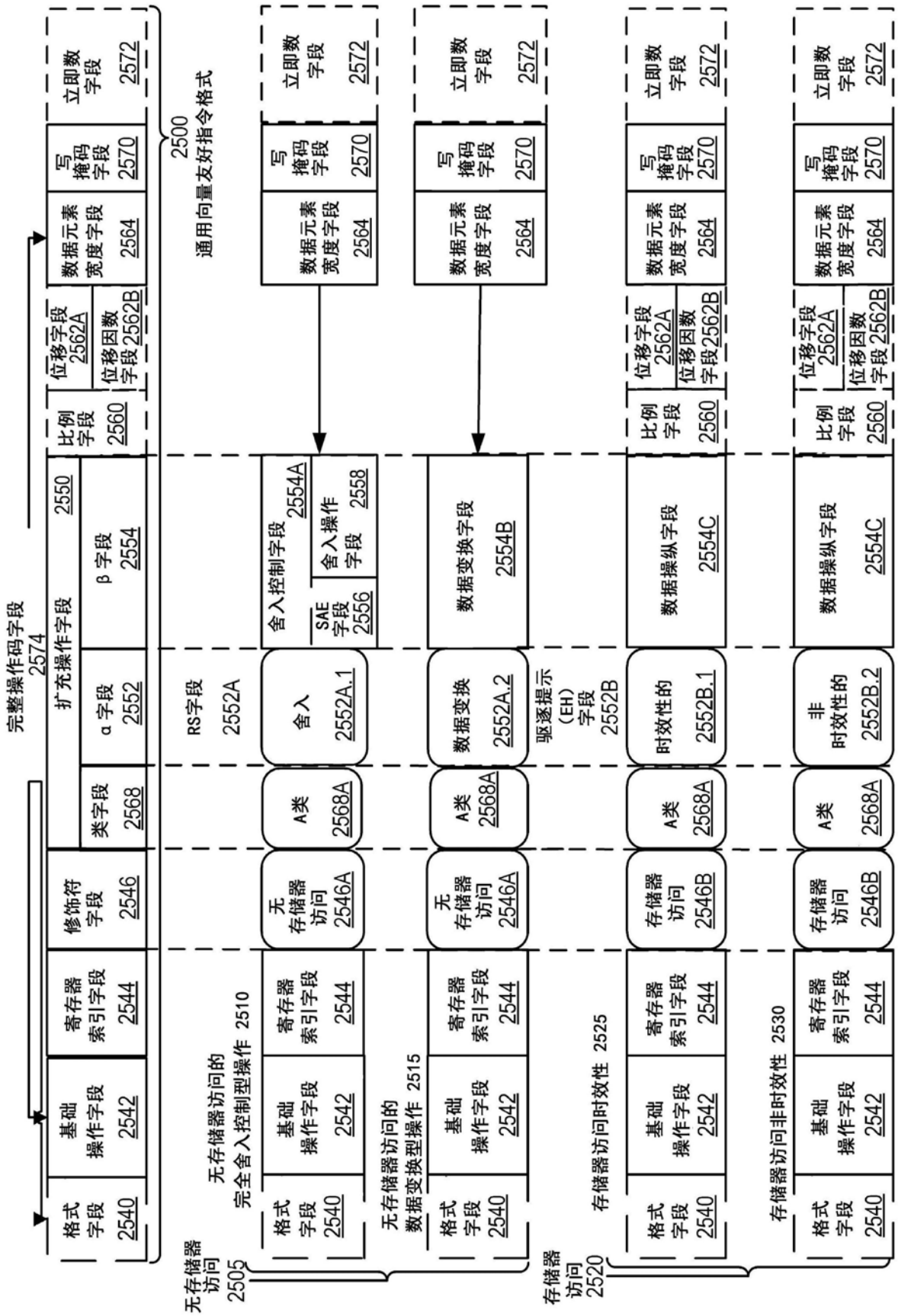

完整操作码字段 2574
2500
通用向量友好指令格式
格式字段 2540
基础操作字段 2542
寄存器索引字段 2544
修饰符字段 2546
扩充操作字段 2550
类字段 2568
α 字段 2552
β 字段 2554
比例字段 2560
位移字段 2562A
位移因数字段 2562B
数据元素宽度字段 2564
写掩码字段 2570
立即数字段 2572
RS字段 2552A
舍入控制字段 2554A
SAE字段 2556
舍入操作字段 2558
数据变换字段 2554B
驱逐提示(EH)字段 2552B
数据操纵字段 2554C
无存储器访问 2505
无存储器访问的完全舍入控制型操作 2510
舍入 2552A.1
数据变换 2552A.2
无存储器访问的数据变换型操作 2515
存储器访问 2520
存储器访问时效性 2525
时效性的 2552B.1
存储器访问非时效性 2530
非时效性的 2552B.2
无存储器访问 2546A
存储器访问 2546B
A类 2568A

图25A

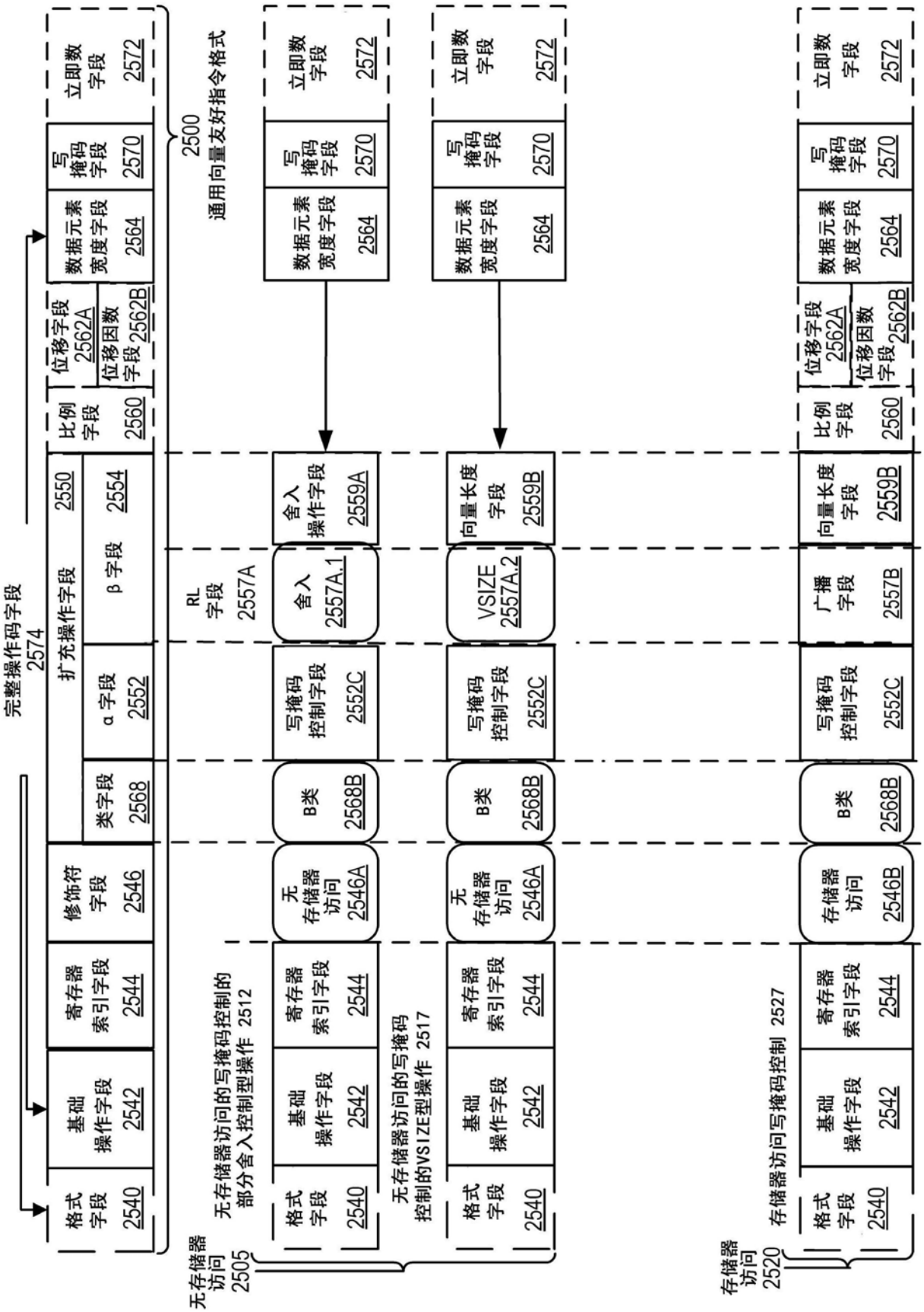

2500
通用向量友好指令格式
完整操作码字段
2574
格式字段 2540
基础操作字段 2542
寄存器索引字段 2544
修饰符字段 2546
扩充操作字段 2550
类字段 2568
α字段 2552
β字段 2554
比例字段 2560
位移字段 2562A
位移因数字段 2562B
数据元素宽度字段 2564
写掩码字段 2570
立即数字段 2572
无存储器访问 2505
无存储器访问的写掩码控制的部分舍入控制型操作 2512
无存储器访问 2546A
B类 2568B
写掩码控制字段 2552C
RL字段 2557A
舍入 2557A.1
舍入操作字段 2559A
无存储器访问的写掩码控制的VSIZE型操作 2517
VSIZE 2557A.2
向量长度字段 2559B
存储器访问 2520
存储器访问写掩码控制 2527
存储器访问 2546B
广播字段 2557B

图25B

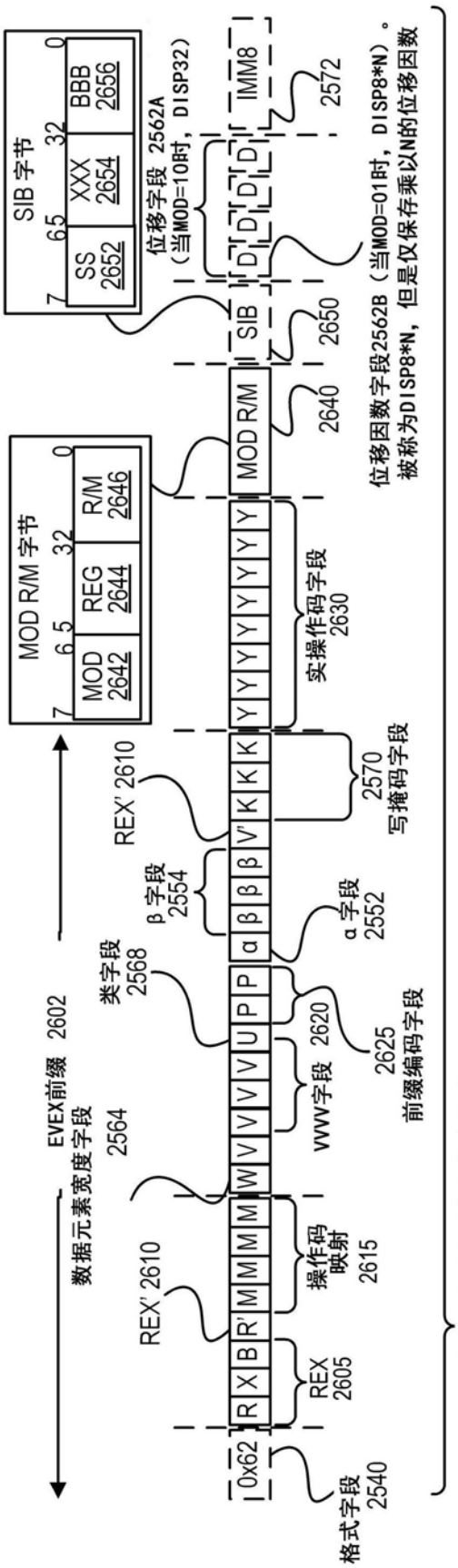

SIB 字节
SS 2652
XXX 2654
BBB 2656
位移字段 2562A
（当MOD=10时，DISP32）
MOD R/M 字节
MOD 2642
REG 2644
R/M 2646
SIB
MOD R/M
IMM8
2572
2650
2640
位移因数字段2562B（当MOD=01时，DISP8*N）。
被称为DISP8*N，但是仅保存乘以N的位移因数
实操作码字段 2630
2570 写掩码字段
REX' 2610
β 字段 2554
α 字段 2552
类字段 2568
EVEX前缀 2602
数据元素宽度字段 2564
REX'' 2610
VVVV字段 2620
操作码映射 2615
2625 前缀编码字段
REX 2605
格式字段 2540
0x62
2600 专用向量友好指令格式

图26A

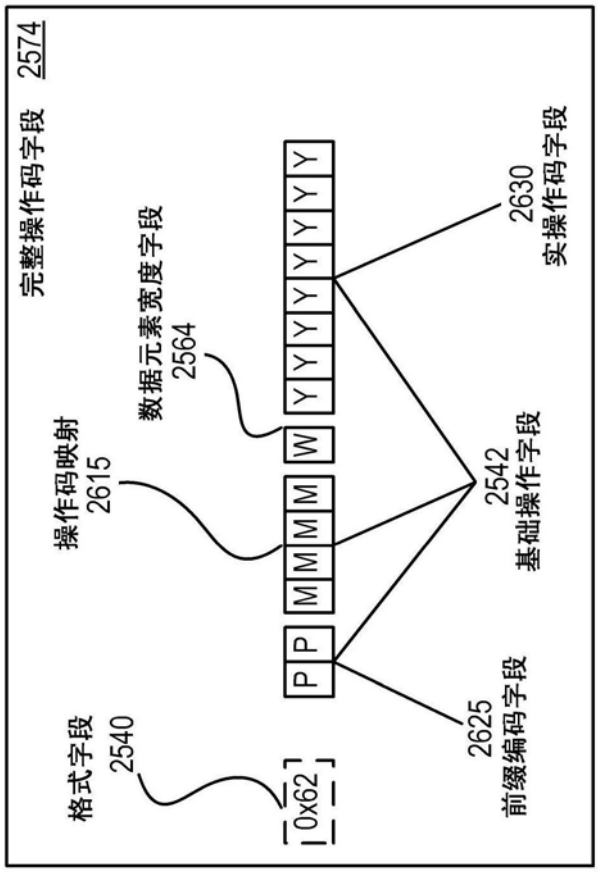

完整操作码字段 2574
格式字段 2540
操作码映射 2615
数据元素宽度字段 2564
0x62
P P M M M W Y Y Y Y Y Y Y Y
2625 前缀编码字段
2542 基础操作字段
2630 实操作码字段

图26B

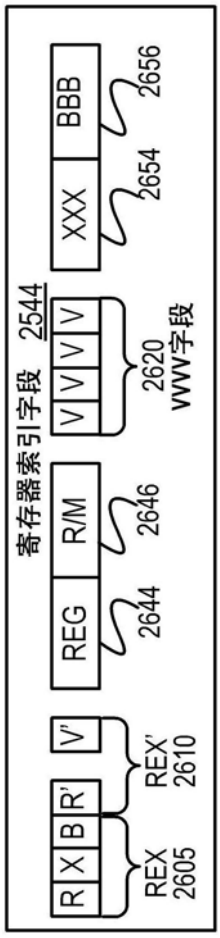

寄存器索引字段 2544
R
X
B
R'
V'
REG
R/M
V
V
V
V
XXX
BBB
REX
2605
REX'
2610
2644
2646
2620
VVV字段
2654
2656

图26C

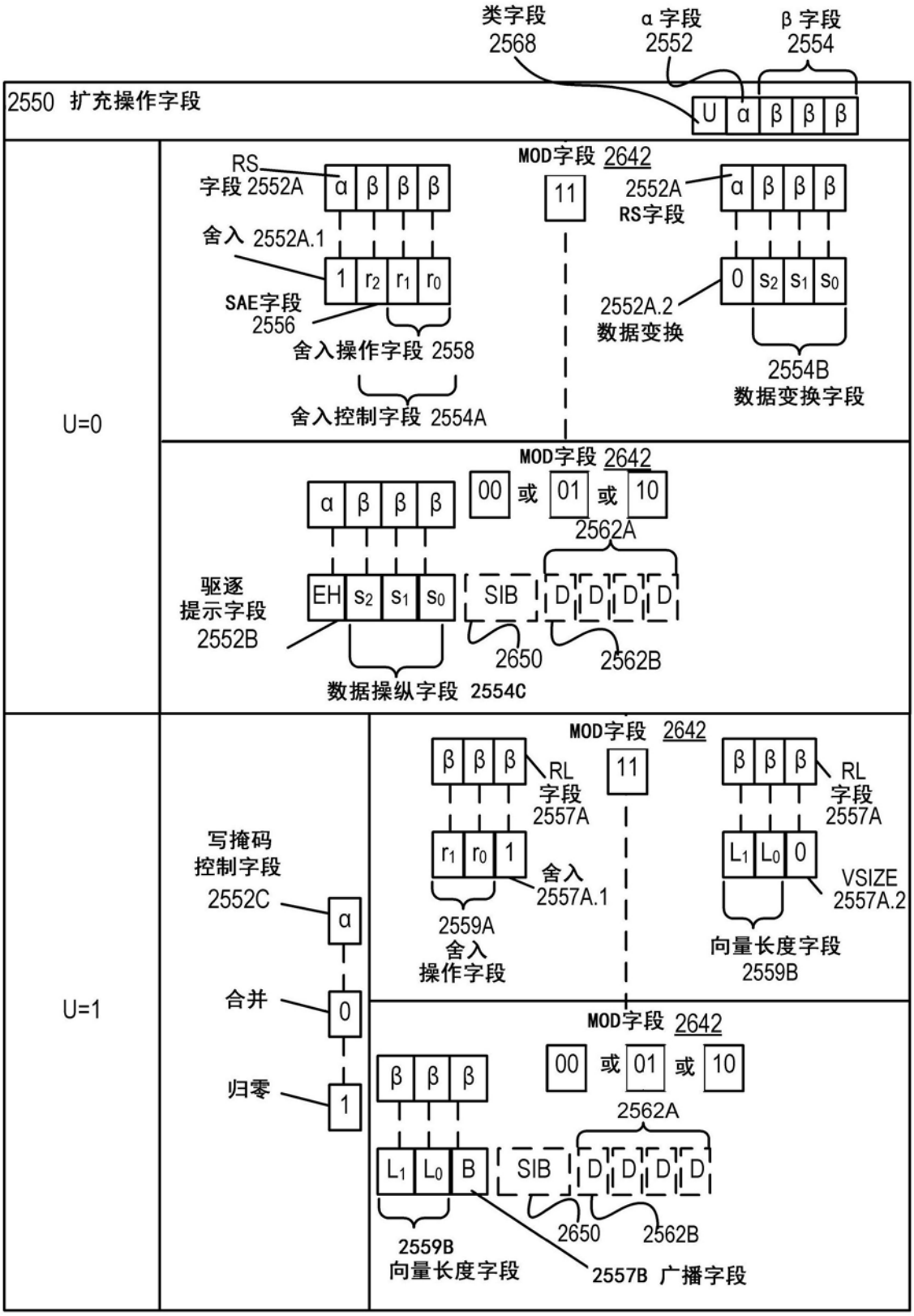

类字段 2568
α 字段 2552
β 字段 2554
2550 扩充操作字段
U α β β β
U=0
RS 字段 2552A
舍入 2552A.1
SAE字段 2556
舍入操作字段 2558
舍入控制字段 2554A
MOD字段 2642
11
2552A RS字段
0 s2 s1 s0
2552A.2 数据变换
2554B 数据变换字段
MOD字段 2642
00 或 01 或 10
2562A
驱逐 提示字段 2552B
EH s2 s1 s0
SIB D D D D
2650
2562B
数据操纵字段 2554C
U=1
写掩码 控制字段 2552C
合并
归零
MOD字段 2642
RL 字段 2557A
r1 r0 1
舍入 2557A.1
2559A 舍入 操作字段
L1 L0 0
VSIZE 2557A.2
向量长度字段 2559B
MOD字段 2642
00 或 01 或 10
2562A
L1 L0 B
SIB D D D D
2650
2562B
2559B 向量长度字段
2557B 广播字段

图26D

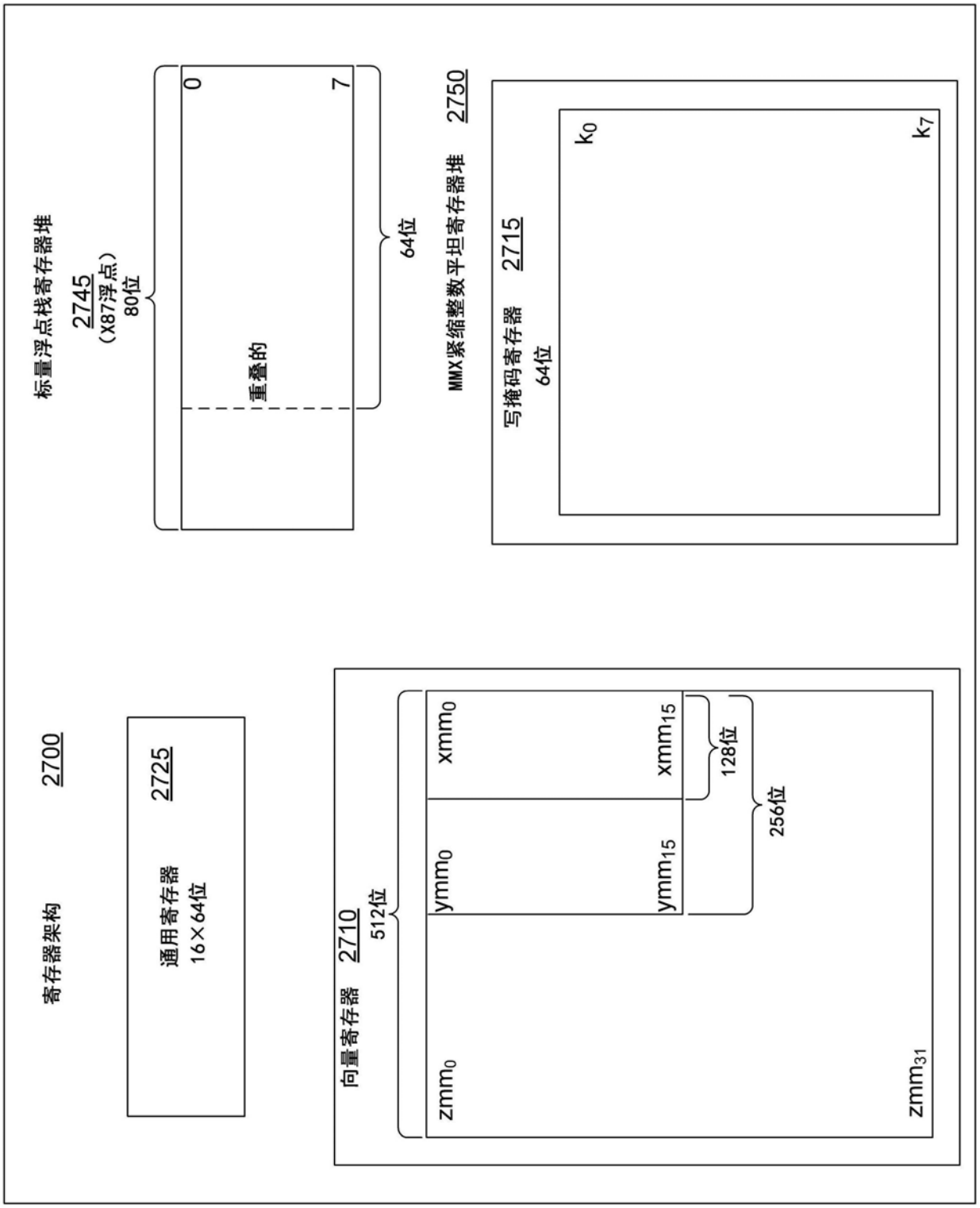

寄存器架构 2700
通用寄存器 2725
16×64位
向量寄存器 2710
512位
zmm0
ymm0
xmm0
ymm15
xmm15
128位
256位
zmm31
标量浮点栈寄存器堆
2745
(X87浮点)
80位
0
7
重叠的
64位
MMX紧缩整数平坦寄存器堆 2750
写掩码寄存器 2715
64位
k0
k7

图27

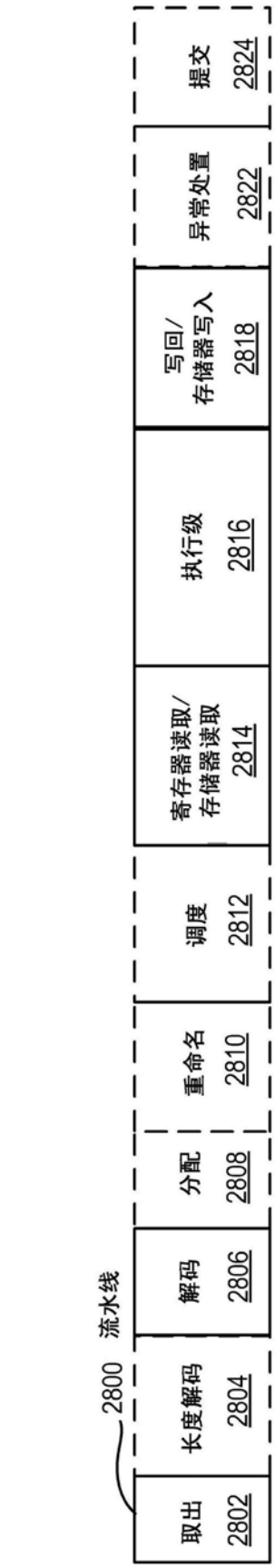

2800
流水线
取出
2802
长度解码
2804
解码
2806
分配
2808
重命名
2810
调度
2812
寄存器读取/
存储器读取
2814
执行级
2816
写回/
存储器写入
2818
异常处置
2822
提交
2824

图28A

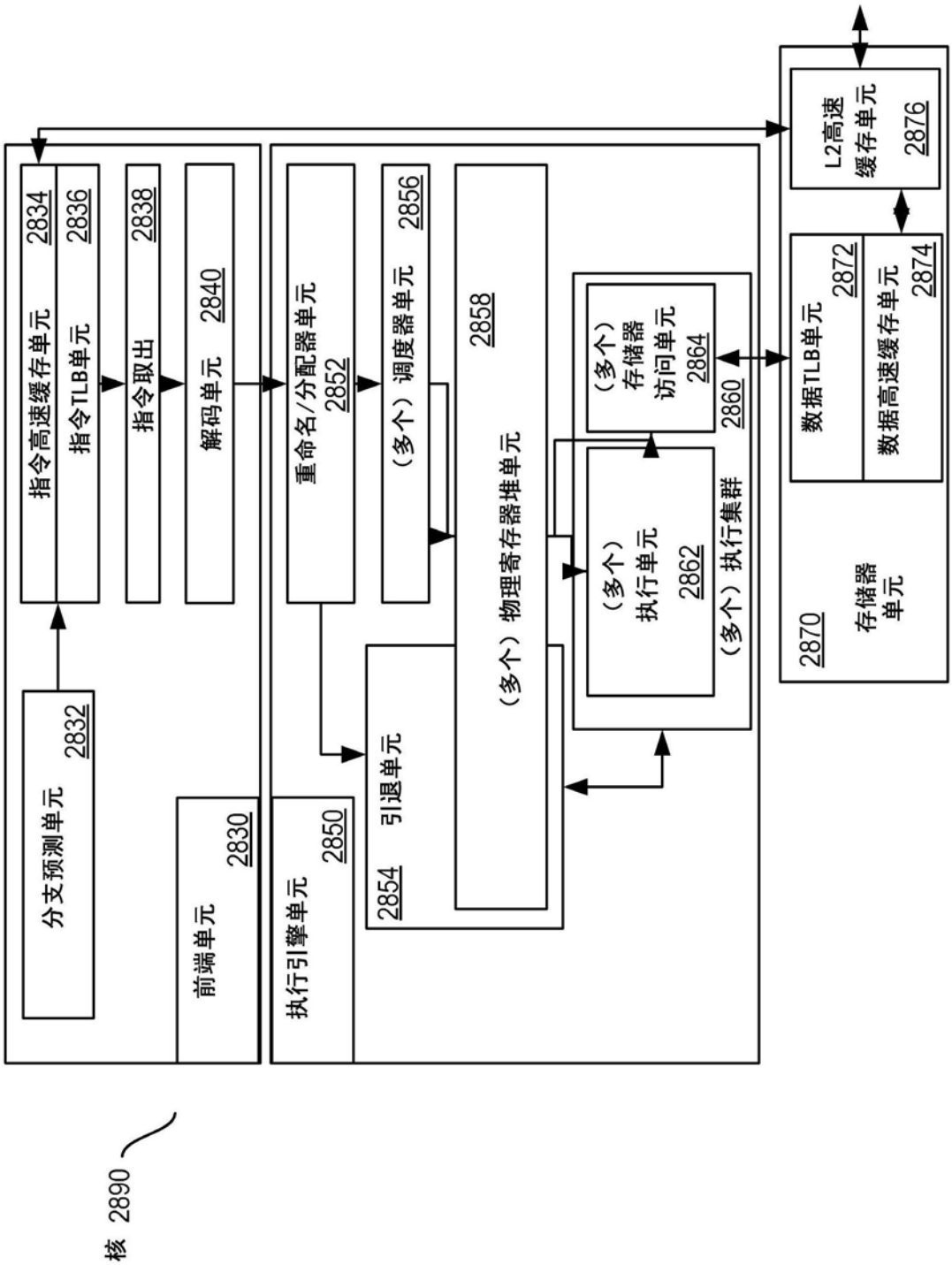

核 2890
分支预测单元 2832
指令高速缓存单元 2834
指令TLB单元 2836
指令取出 2838
解码单元 2840
前端单元 2830
执行引擎单元 2850
重命名/分配器单元 2852
2854 引退单元
（多个）调度器单元 2856
2858 （多个）物理寄存器堆单元
（多个）执行单元 2862
（多个）存储器访问单元 2864
（多个）执行集群 2860
2870 存储器单元
数据TLB单元 2872
数据高速缓存单元 2874
L2高速缓存单元 2876

图28B

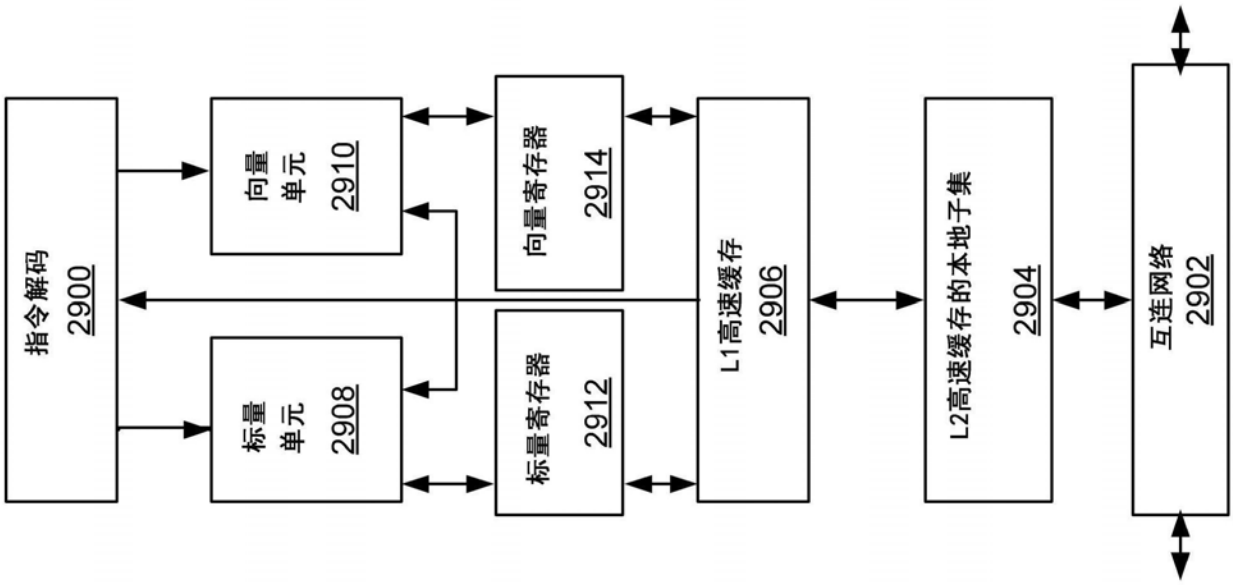

指令解码
2900
向量
单元
2910
标量
单元
2908
向量寄存器
2914
标量寄存器
2912
L1高速缓存
2906
L2高速缓存的本地子集
2904
互连网络
2902

图29A

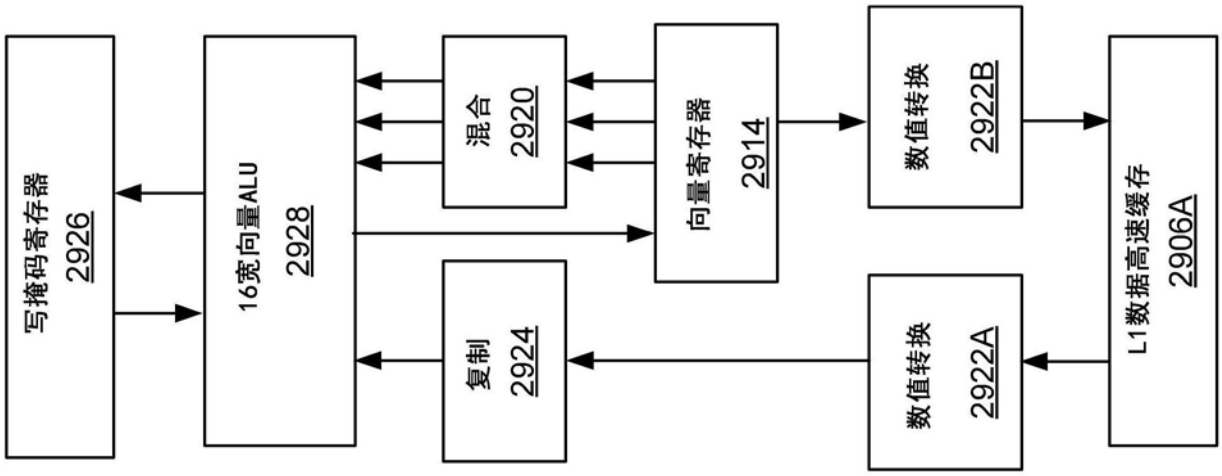

写掩码寄存器
2926
16宽向量ALU
2928
混合
2920
向量寄存器
2914
数值转换
2922B
L1数据高速缓存
2906A
复制
2924
数值转换
2922A

图29B

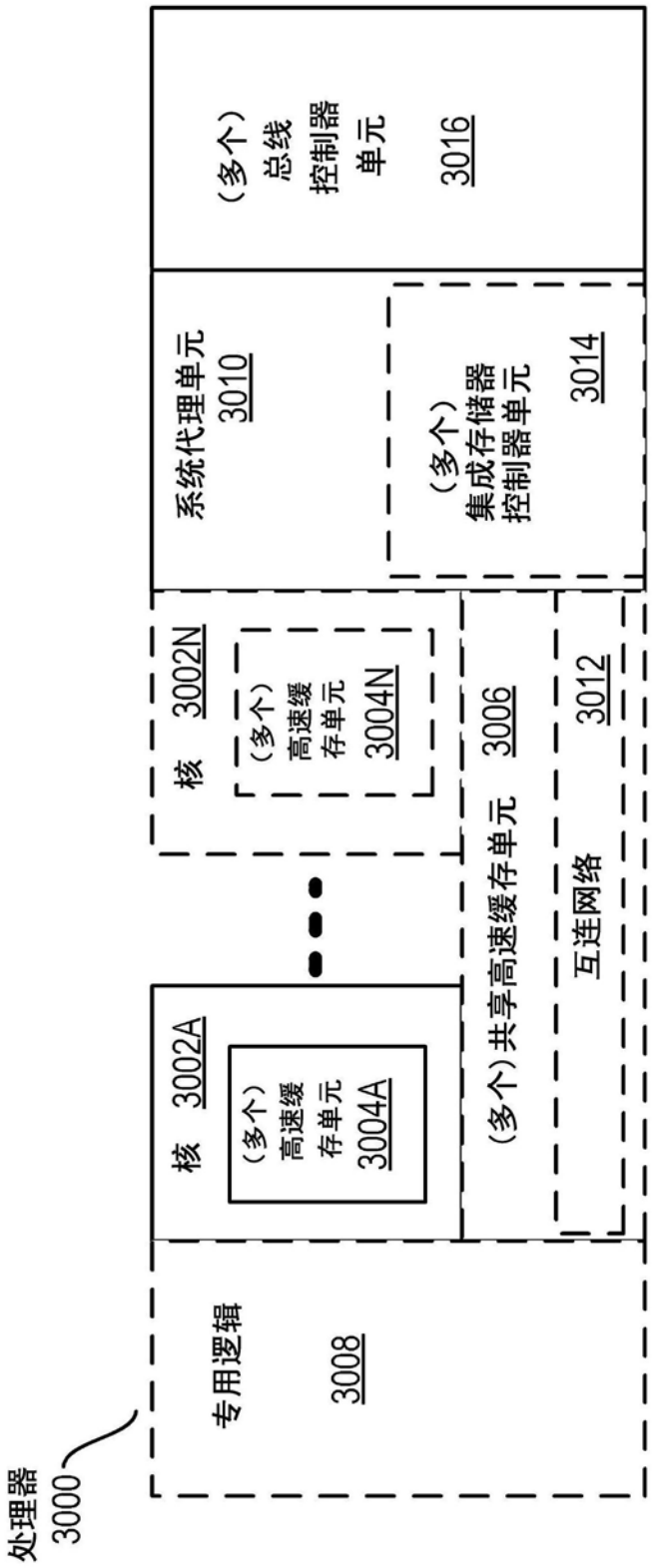

处理器
3000
专用逻辑
3008
核 3002A
(多个)
高速缓
存单元
3004A
核 3002N
(多个)
高速缓
存单元
3004N
(多个)共享高速缓存单元 3006
互连网络 3012
系统代理单元
3010
(多个)
集成存储器
控制器单元
3014
(多个)
总线
控制器
单元
3016

图30

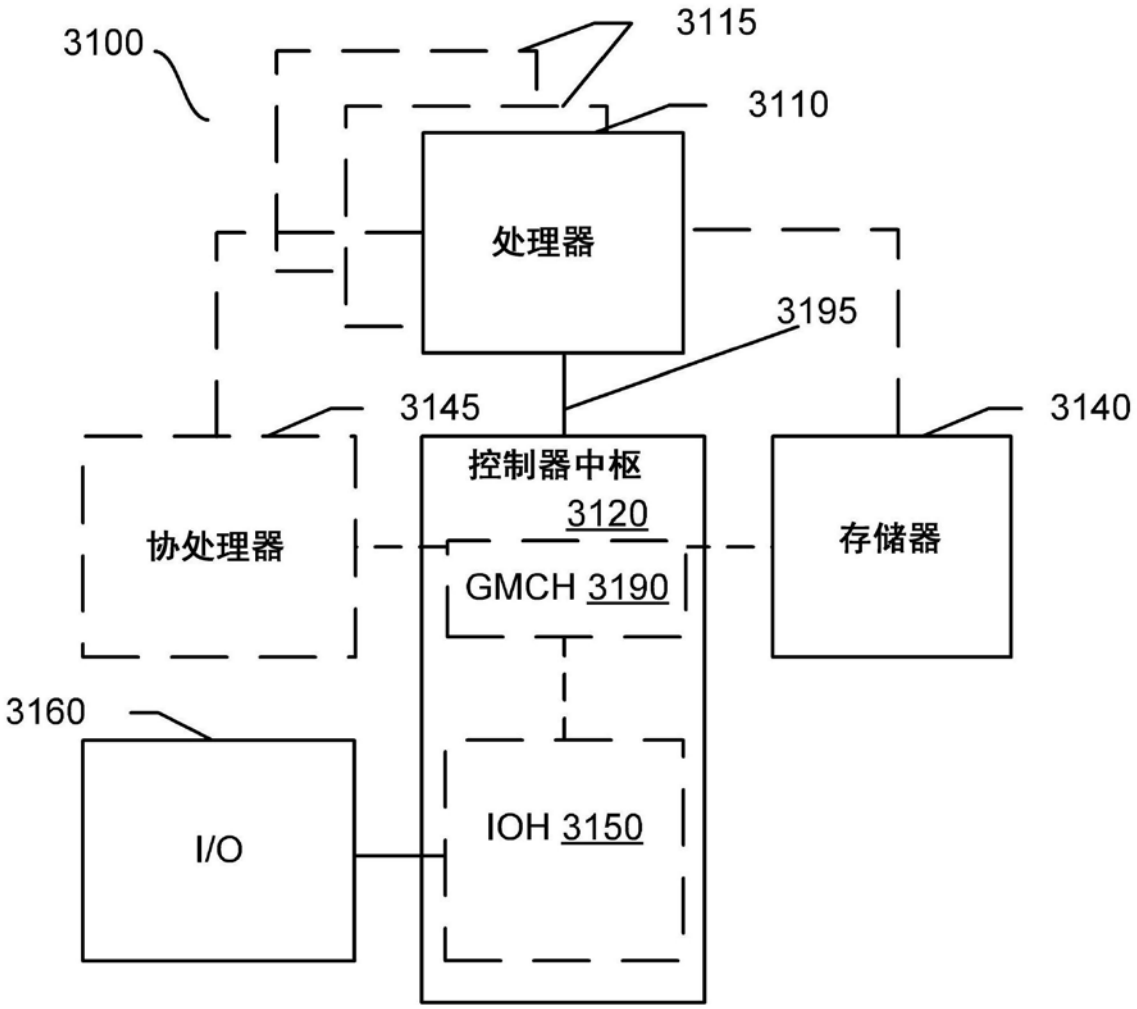

3100
3115
3110
处理器
3195
3145
3140
控制器中枢
3120
协处理器
存储器
GMCH 3190
3160
I/O
IOH 3150

图31

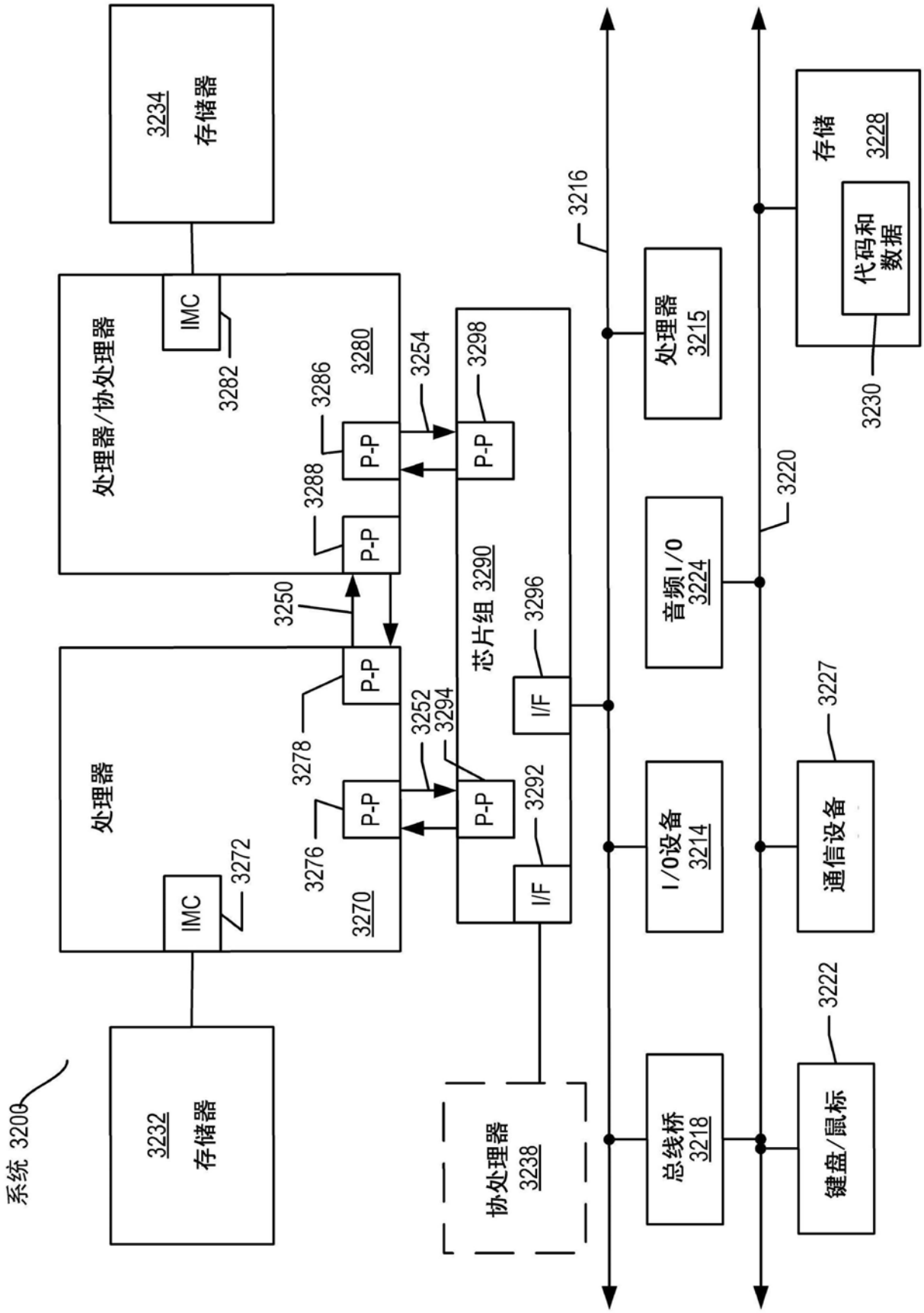

系统 3200
3232 存储器
3234 存储器
IMC
3272
IMC
3282
处理器
处理器/协处理器
3270
3276
3278
3280
3286
3288
P-P
P-P
P-P
P-P
3250
3252
3254
3294
3298
芯片组 3290
3292
3296
I/F
I/F
协处理器 3238
3216
处理器 3215
音频I/O 3224
I/O设备 3214
总线桥 3218
3220
3227
通信设备
3222
键盘/鼠标
存储 3228
代码和数据
3230

图32

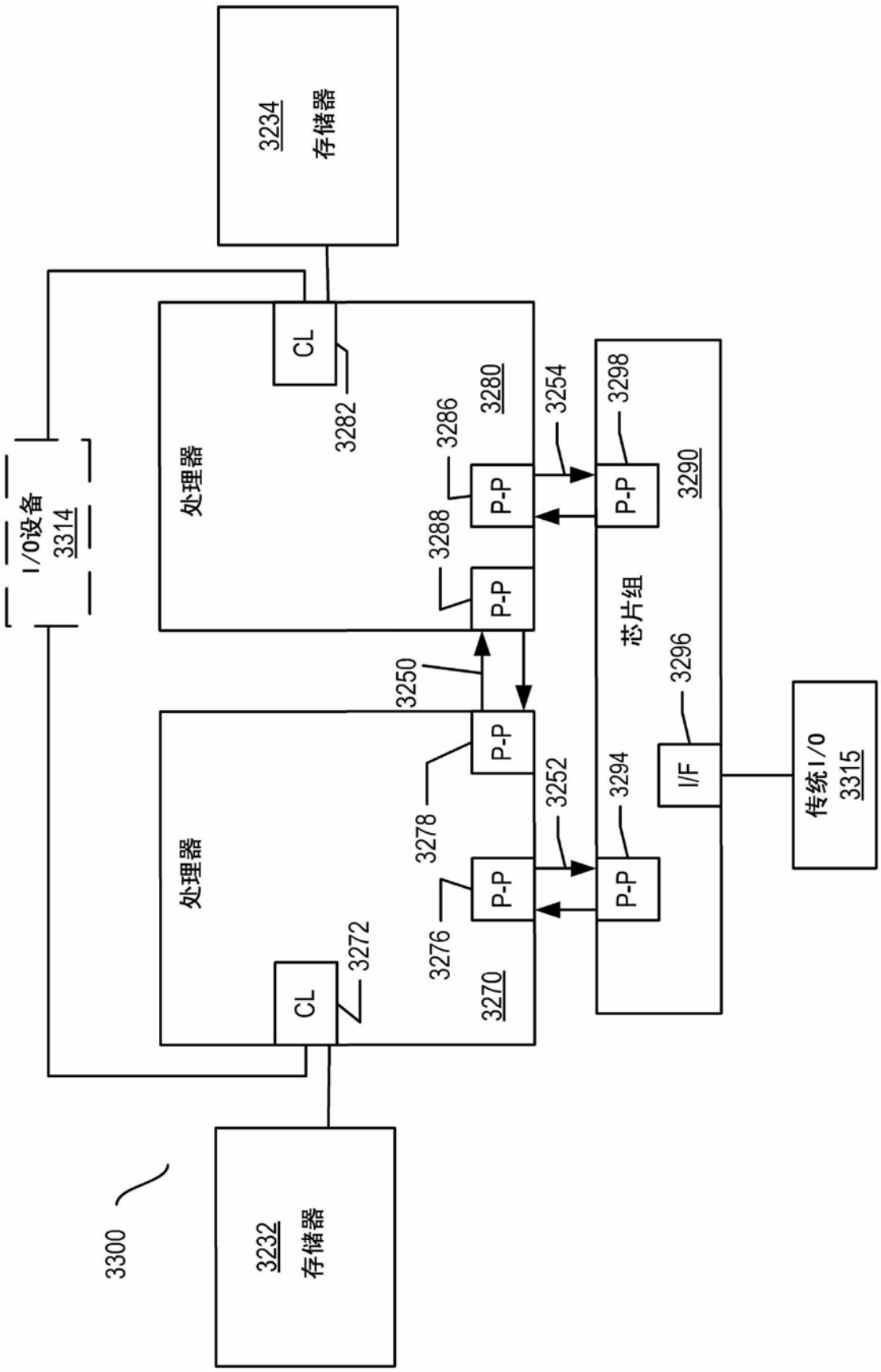

3234
存储器
CL
3282
处理器
3280
3286
P-P
3288
P-P
3254
3298
P-P
3290
芯片组
I/O设备
3314
3250
P-P
3278
处理器
3252
3294
P-P
3276
3270
CL
3272
3296
I/F
传统I/O
3315
3300
3232
存储器

图33

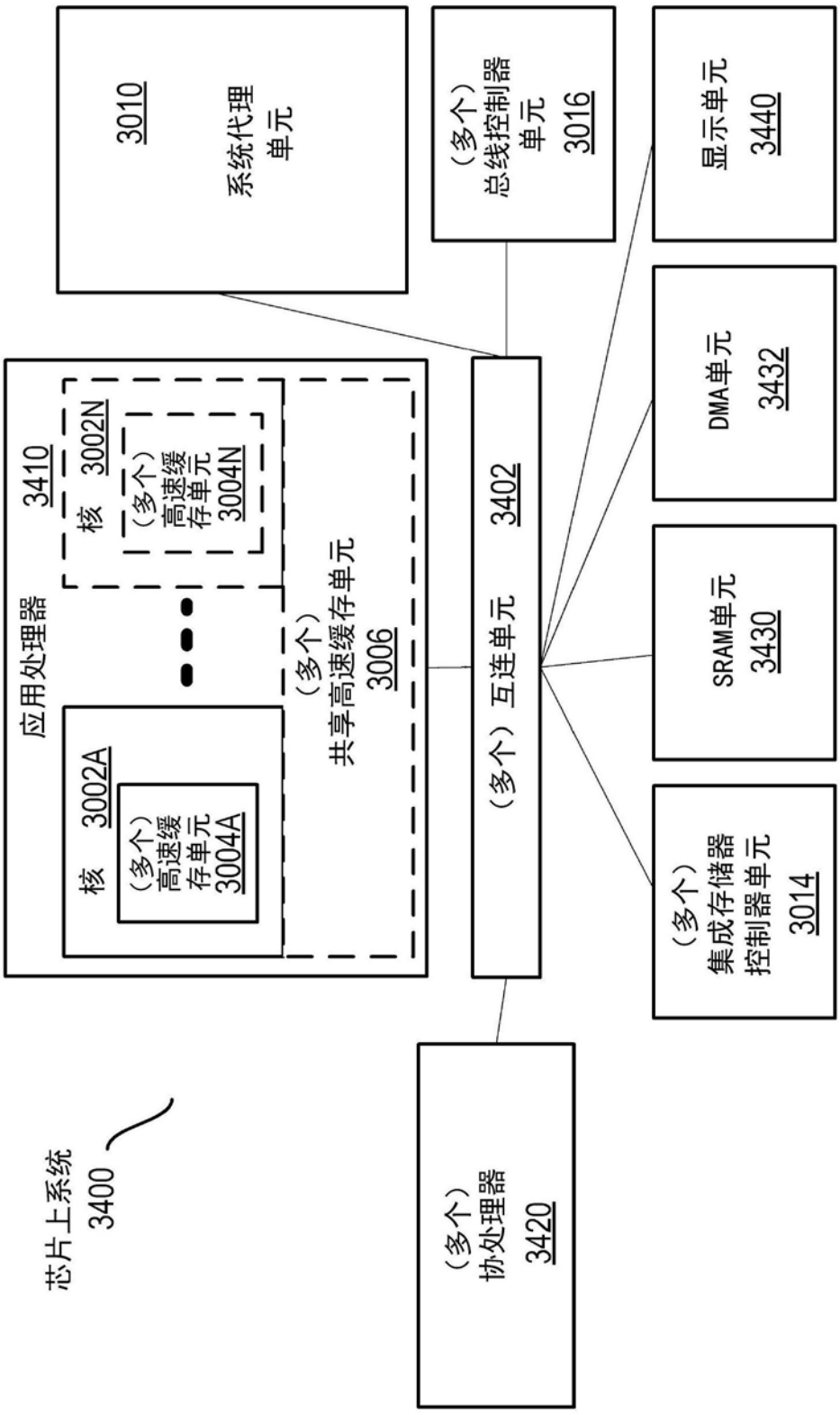

芯片上系统
3400
应用处理器
3410
核 3002A
(多个)高速缓存单元 3004A
核 3002N
(多个)高速缓存单元 3004N
(多个)共享高速缓存单元 3006
(多个)协处理器 3420
(多个)互连单元 3402
3010 系统代理单元
(多个)总线控制器单元 3016
显示单元 3440
DMA单元 3432
SRAM单元 3430
(多个)集成存储器控制器单元 3014

图34

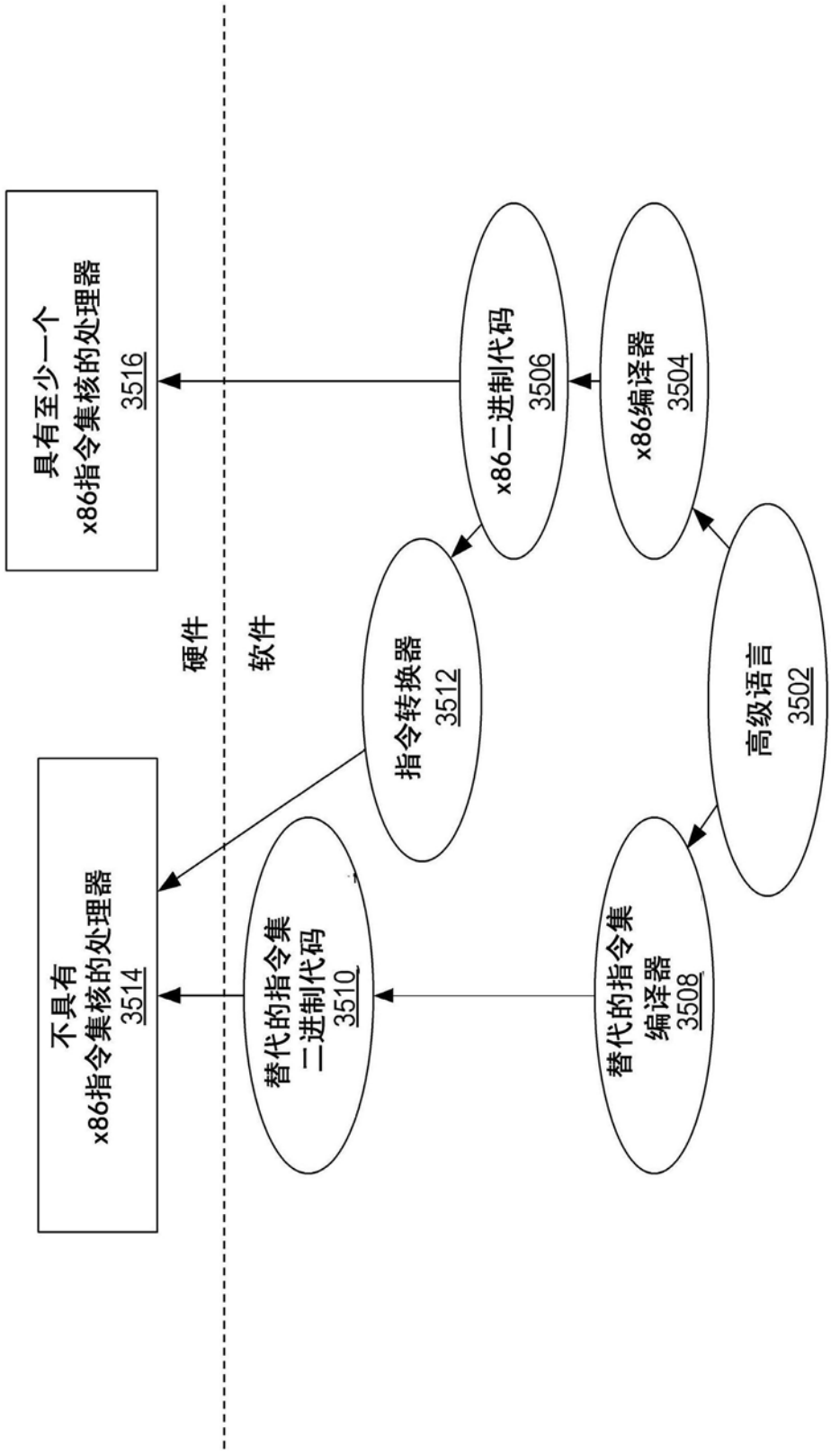

具有至少一个
x86指令集核的处理器
3516
x86二进制代码
3506
x86编译器
3504
硬件
软件
指令转换器
3512
高级语言
3502
不具有
x86指令集核的处理器
3514
替代的指令集
二进制代码
3510
替代的指令集
编译器
3508

图35