(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 118829969 A
(43)申请公布日 2024.10.22

(21)申请号 202280093265.3
(22)申请日 2022.12.15
(30)优先权数据
2203431.8 2022.03.11 GB
(85)PCT国际申请进入国家阶段日
2024.09.06
(86)PCT国际申请的申请数据
PCT/GB2022/053244 2022.12.15
(87)PCT国际申请的公布数据
WO2023/170373 EN 2023.09.14
(71)申请人 ARM有限公司
地址 英国剑桥
(72)发明人 亚利阳德罗·马丁内兹·文森特
孙彭
(74)专利代理机构 北京东方亿思知识产权代理有限责任公司 11258
专利代理师 杨佳婧
(51)Int.Cl.
G06F 9/30(2006.01)
G06F 9/38(2006.01)

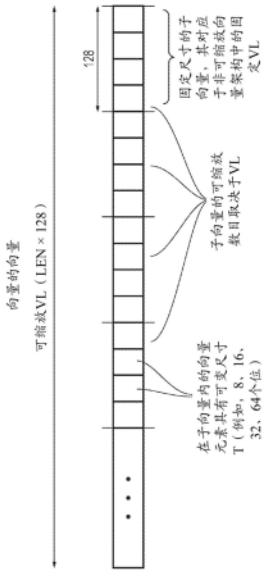
权利要求书3页 说明书19页 附图17页

(54)发明名称

用于可缩放向量指令集架构的子向量支持指令

(57)摘要

一种设备,具有:处理电路(16),该处理电路用以执行数据处理;和指令解码电路(10),该指令解码电路用以响应于根据可缩放向量指令集架构所定义的程序指令的解码而执行该数据处理,该可缩放向量指令集架构支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列可在具有支持不同最大向量长度的硬件的设备上执行。该指令解码电路和该处理电路支持将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素。响应于该子向量支持指令,该指令解码电路控制该处理电路在子向量粒度下针对该给定向量执行操作。每个子向量具有相等子向量长度。



CN 118829969 A

1.一种设备,所述设备包括:

处理电路,所述处理电路用以执行数据处理;和

指令解码电路,所述指令解码电路用以响应于根据可缩放向量指令集架构所定义的程序指令的解码而执行所述数据处理,所述可缩放向量指令集架构支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列能够在具有支持不同最大向量长度的硬件的设备上执行;其中:

所述指令解码电路和所述处理电路被配置为支持将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素,每个子向量具有相等子向量长度;并且

所述指令解码电路被配置为响应于所述子向量支持指令,控制所述处理电路在子向量粒度下针对所述给定向量执行操作。

2.根据权利要求1所述的设备,其中每个子向量具有在待使用所述子向量支持指令来执行的给定指令序列的编译时间下是已知的子向量长度。

3.根据权利要求1和2中任一项所述的设备,其中在所述给定指令序列的编译时间下,所述给定向量包括多少个子向量是未知的。

4.根据任一前述权利要求所述的设备,其中所述指令解码电路被配置为响应于所述子向量支持指令,控制所述处理电路以响应于执行所述子向量支持指令的相同实例而处理所述子向量中的每个子向量。

5.根据任一前述权利要求所述的设备,其中每个子向量具有与用于所述给定向量的向量长度无关的架构定义固定尺寸的子向量长度。

6.根据权利要求5所述的设备,其中所述架构定义固定尺寸对应于规定用于根据预定非可缩放向量指令集架构处理的向量指令的架构定义最大向量长度。

7.根据权利要求5和6中任一项所述的设备,其中所述架构定义固定尺寸是128个位。

8.根据任一前述权利要求所述的设备,其中每个子向量的每个向量元素具有可变元素尺寸,并且所述子向量长度与针对每个子向量内的每个向量元素而使用哪个元素尺寸无关。

9.根据任一前述权利要求所述的设备,其中对于至少一个子向量支持指令,在子向量粒度下执行的所述操作是针对每个子向量而对所述子向量内的向量元素执行的操作,而与其它子向量中的元素无关。

10.根据任一前述权利要求所述的设备,其中对于至少一个子向量支持指令,在子向量粒度下执行的所述操作是针对子向量内的每个元素位置而对所述多个子向量中的每个子向量内的所述元素位置处的相应向量元素执行的操作。

11.根据任一前述权利要求所述的设备,其中对于至少一个子向量支持指令,在子向量粒度下执行的所述操作是用以设置预测值的所选择预测位的操作或是取决于所述预测值的所选择预测位而执行操作的操作,其中所选择预测位是对应于向量的子向量尺寸部分的预测位。

12.根据任一前述权利要求所述的设备,其中指令解码器被配置为响应于子向量支持置换指令,控制所述处理电路以针对向量结果的每个子向量将所述子向量设置成一个或多个向量元素的置换,所述一个或多个向量元素是从至少一个向量操作数的对应定位子向量

内的向量元素中选择的。

13.根据任一前述权利要求所述的设备,其中所述指令解码器被配置为响应于子向量支持缩减指令,控制所述处理电路在子向量粒度下执行至少一个缩减操作,每个缩减操作用以将操作数向量的多个向量元素缩减成结果内的单一数据值。

14.根据权利要求13所述的设备,其中以子向量内的子向量支持缩减指令来说,对于每个缩减操作,所述多个向量元素包括在所述操作数向量的对应子向量内的所述相应向量元素。

15.根据权利要求13和14中任一项所述的设备,其中以子向量间的子向量支持缩减指令来说,对于每个缩减操作,所述多个向量元素包括在所述操作数向量的多个子向量内的对应元素位置处的所述向量元素。

16.根据任一前述权利要求所述的设备,其中所述指令解码器被配置为响应于子向量支持加载/存储指令,控制所述处理电路执行加载/存储操作,以在子向量粒度下于存储器系统与至少一个向量寄存器之间转移一个或多个子向量。

17.根据权利要求16所述的设备,其中所述子向量支持加载/存储指令是与预测值关联的预测指令;并且

所述指令解码器被配置为响应于所述子向量支持加载/存储指令,控制所述处理电路以基于在子向量粒度下从所述预测值选择的预测位而控制执行或屏蔽所述一个或多个子向量的每次转移。

18.根据任一前述权利要求所述的设备,其中所述指令解码器被配置为响应于子向量支持递增/递减指令,控制所述处理电路以基于向量中有多少个子向量尺寸部分是通过在子向量粒度下从预测值选择的所述预测值的位指示为有效而递增或递减操作数值。

19.根据权利要求18所述的设备,其中所述预测值是下列中的一者:

由所述子向量支持递增/递减指令指定作为预测操作数的预测值;以及

由所述子向量支持递增/递减指令指定的预测模式识别符所表示的预测值,所述预测模式识别符在子向量粒度下指定预测位的预定模式。

20.根据任一前述权利要求所述的设备,其中所述指令解码器被配置为响应于子向量支持预测设置指令,控制所述处理电路执行预测设置操作以在子向量粒度下设置预测值的位,以指示向量中有哪些子向量是有效的。

21.根据权利要求20所述的设备,其中所述预测设置操作包括基于下列中的一者来设置所述预测值:

预测模式识别符,所述预测模式识别符指定要在子向量粒度下施加的预测位的预定模式;以及

子向量粒度比较操作,所述子向量粒度比较操作是基于第一操作数与第二操作数的比较。

22.一种方法,所述方法包括:

使用指令解码电路来解码根据可缩放向量指令集架构所定义的程序指令,所述可缩放向量指令集架构支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列能够在具有支持不同最大向量长度的硬件的设备上执行;以及

控制处理电路以响应于所述程序指令的解码而执行数据处理;其中:

所述指令解码电路和所述处理电路支持将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素,每个子向量具有相等子向量长度;并且

响应于所述子向量支持指令,所述指令解码电路控制所述处理电路在子向量粒度下针对所述给定向量执行操作。

23.一种用以控制主机数据处理设备以提供用于目标码的执行的指令执行环境的计算机程序;所述计算机程序包括:

指令解码程序逻辑,所述指令解码程序逻辑用以解码所述目标码的指令以响应于所述目标码的所述指令而控制所述主机数据处理设备执行数据处理;其中:

所述指令解码程序逻辑支持根据可缩放向量指令集架构所定义的程序指令的解码,所述可缩放向量指令集架构支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列能够在具有支持不同最大向量长度的硬件的设备上执行;

所述指令解码程序逻辑包括子向量支持指令解码程序逻辑,以解码将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素,每个子向量具有相等子向量长度;并且

所述指令解码程序逻辑被配置为响应于所述子向量支持指令,控制所述主机数据处理设备在子向量粒度下针对所述给定向量执行操作。

24.一种存储介质,所述存储介质存储根据权利要求23所述的计算机程序。

用于可缩放向量指令集架构的子向量支持指令

[0001] 本技术涉及数据处理领域。

[0002] 指令集架构(instruction set architecture,ISA)定义在开发特定软件程序时软件开发者或编译器可用的指令集,并且以对应方式定义需要由硬件的处理器具体实施支持以允许硬件与根据该ISA编写的软件兼容的指令集。例如,ISA可针对每个指令定义指令的编码、其输入操作数和结果值的表示和用于将指令的输入操作数映射到结果的函数。

[0003] 向量ISA支持至少一个向量指令,该至少一个向量指令在包括在单一寄存器内表示的两个或更多个独立向量元素的向量操作数上操作和/或产生包括两个或更多个独立向量元素的向量结果。向量指令可以SIMD(单指令、多数据)的方式处理,以允许响应于单一指令而对不同的数据值执行多个独立计算。向量指令可用于,例如,允许将以高层码编写的指令的标量回圈向量化,使得可响应于向量化回圈的单一迭代而执行对应于多个回圈迭代的处理。这通过减少实行特定数据处理量所需要提取、解码和执行的指令的数目而有助于改进性能。

[0004] 至少一些示例提供一种设备,该设备包括:处理电路,该处理电路用以执行数据处理;和指令解码电路,该指令解码电路用以响应于根据可缩放向量指令集架构所定义的程序指令的解码而执行该数据处理,该可缩放向量指令集架构支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列能够在具有支持不同最大向量长度的硬件的设备上执行;其中:该指令解码电路和该处理电路被配置为支持将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素,每个子向量具有相等子向量长度;并且该指令解码电路被配置为响应于该子向量支持指令,控制该处理电路在子向量粒度下针对该给定向量执行操作。

[0005] 至少一些示例提供一种方法,该方法包括:使用指令解码电路来解码根据可缩放向量指令集架构所定义的程序指令,该可缩放向量指令集架构支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列能够在具有支持不同最大向量长度的硬件的设备上执行;和控制处理电路以响应于所述程序指令的解码而执行数据处理;其中:该指令解码电路和该处理电路支持将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素,每个子向量具有相等子向量长度;并且响应于该子向量支持指令,该指令解码电路控制该处理电路在子向量粒度下针对该给定向量执行操作。

[0006] 至少一些示例提供一种计算机程序,该计算机程序用以控制主机数据处理设备以提供用于目标码的执行的指令执行环境;该计算机程序包括:指令解码程序逻辑,该指令解码程序逻辑用以解码该目标码的指令以响应于该目标码的该指令而控制该主机数据处理设备执行数据处理;其中:该指令解码程序逻辑支持根据可缩放向量指令集架构所定义的程序指令的解码,该可缩放向量指令集架构支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列能够在具有支持不同最大向量长度的硬件的设备上执行;该指令解码程序逻辑包括子向量支持指令解码程序逻辑,用以解码将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素,每个子向量具有相等子向量长度;并且该指令解码程序逻辑被配置为响应于该子向量支持指令,控制该主机数据处理设备在

子向量粒度下针对该给定向量执行操作。

[0007] 该计算机程序可存储在存储介质上。该存储介质可为非暂态存储介质。

[0008] 本技术的进一步方面、特征和优点将由于结合附图阅读的以下示例描述而显而易见,在所述附图中:

[0009] 图1例示了支持向量ISA的数据处理设备的示例;

[0010] 图2示出用于可缩放向量ISA的寄存器的示例;

[0011] 图3示出预测和可变向量元素尺寸的示例;

[0012] 图4例示了使支持不同最大向量长度的不同硬件具体实施得以执行相同指令序列的可缩放向量ISA的示例;

[0013] 图5示出被视为子向量的向量的向量的示例;

[0014] 图6示出处理子向量支持指令的方法;

[0015] 图7至图11例示了子向量支持置换指令(sub-vector-supporting permute instruction)的数个示例;

[0016] 图12和图13例示了子向量支持缩减指令的示例;

[0017] 图14和图15例示了子向量支持加载/存储指令的示例;

[0018] 图16和图17例示了子向量支持预测设置指令的示例;

[0019] 图18例示了子向量支持递增/递减指令的示例;并且

[0020] 图19例示了模拟器具体实施。

[0021] 设备具有:处理电路,该处理电路用以执行数据处理;和指令解码电路,该指令解码电路用以响应于根据可缩放向量指令集架构(ISA)所定义的程序指令的解码而执行该数据处理。可缩放向量ISA(还称为“向量长度未知(vector length agnostic)”向量ISA)支持在可缩放向量长度的向量上操作的向量指令,以使相同指令序列能够在具有支持不同最大向量长度的硬件的设备上操作。这是有用的原因在于其允许处理器具体实施的不同硬件设计者可取决于其设计优先考量为高性能或减少电路面积和功率消耗而选择不同的最大向量长度,而软件开发者则不需要针对特定硬件平台来量身定制其软件,这是因为无论特定硬件平台支持的特定最大向量长度为何,根据可缩放向量ISA编写的软件都可横跨任何支持可缩放向量ISA的硬件平台来执行。因此,待用于可缩放向量ISA的特定向量指令的向量长度在编译时间下是未知的(ISA本身未经定义为固定的,也不由软件本身的参数所指定)。响应于可缩放向量ISA的给定向量指令而执行的操作可取决于针对特定硬件具体实施所选择的向量长度而不同(例如,与支持较小最大向量长度的硬件相比,支持较大最大向量长度的硬件可针对给定向量指令来处理更大数目的向量元素)。因此,具有较短向量长度的具体实施可比具有较长向量长度的具体实施需要更大数目的回圈迭代,以实行特定功能。

[0022] 虽然可缩放向量ISA对于实现可在具有不同最大向量长度的处理器具体实施之间轻易地移植的平台独立程序码的开发可能是非常有用的,但可能存在显著数量的通过假设已知向量长度来编译的传统码。将传统软件重新开发以便与可缩放向量ISA一起使用,可能需要可观的软件开发工作量。尤其如此的原因在于:一般用以改善向量化码的性能的一些软件技术(如软件管线化或回圈展开)可能依赖用于指令的向量长度在编译时间下是已知的。因此,由于在传统码中使用的一些技术可能在可缩放向量ISA中不可用,所以可能无法直接将非可缩放向量ISA的指令重映射成可缩放向量ISA的指令。这可能是采用可缩放向量

ISA的显著阻碍,并且可导致一些软件开发者选择不使用可缩放向量ISA,使得:即使是在支持使用可缩放向量ISA的大的最大向量长度的处理器具体实施上(该处理器具体实施还可能出于回溯兼容性的原因而支持非可缩放向量ISA),可能有显著数量的在支持可缩放向量ISA的较新处理器上执行的软件仍然使用性能效率较低的非可缩放向量ISA(其使用相对短的最大向量长度)。这意味着硬件的完整性能能力可能不为许多程序所使用。

[0023] 在下文讨论的示例中,指令解码电路和处理电路在可缩放向量ISA内支持将给定向量视为包括多个子向量的子向量支持指令,其中每个子向量包括多个向量元素并且每个子向量具有相等子向量长度。响应于该子向量支持指令,该指令解码电路控制所述处理电路在子向量粒度下针对该给定向量执行操作。由于可将非可缩放向量ISA的每个向量映射到可缩放ISA中的向量的子向量中的一个子向量,因此这有助于减少实现可缩放向量ISA的采用所需的软件开发工作量。这使将非可缩放软件映射到可缩放软件更加简单,并且因此减少使用可缩放向量ISA的阻碍,使得更大分量的在支持可缩放向量ISA的设备上执行的码更有可能实际上使用可缩放向量ISA,其通常将会改进横跨一系列处理器具体实施的平均性能,这是由于支持较长向量长度的高端处理器可更有可能得以利用这些较长向量长度来改进性能。

[0024] 每个子向量可具有在待使用子向量支持指令来执行的给定指令序列的编译时间下是已知的子向量长度。这是有帮助的,这是由于其允许将可缩放向量ISA的向量操作数或结果定义为“向量的向量(vector of vectors)”(其包括若干个各具有已知向量长度的较小子向量),使得根据使用非可缩放向量ISA编写的软件所定义的多个向量可根据可缩放向量ISA而组合成较大向量。由于每个子向量的长度在编译时间下是已知的,因此可在子向量的粒度下施加依赖在编译时间下的向量长度的知识的任何性能改进软件技术,进而使软件开发者更容易将根据非可缩放向量ISA编写的码映射成根据可缩放向量ISA所定义的码,同时保持所述软件技术。

[0025] 在待使用子向量支持指令来执行的给定指令序列的编译时间下,给定向量包括多少个子向量是未知的。换句话说,给定向量的整体向量长度可为如可缩放向量ISA(其可定义不同处理器具体实施可允许的各种最大向量长度)所允许的可缩放向量长度。这意味着:在支持一系列不同的性能/电力点的同时,子向量支持指令仍可从可缩放向量ISA的平台独立性质中获益。然而,仍然可使用依赖待用作为粒度的向量长度的知识以供向量化的软件优化,这是由于:在可使用子向量支持指令将可变数目的子向量容纳在可缩放向量长度的较大给定向量中以前,软件优化可参考具有已知子向量长度的子向量而施加。例如,可将先前已使用非可缩放向量ISA采向量化形式实施的向量化回圈的可变数目的迭代映射成可缩放向量ISA中的向量化回圈,其中原始向量化回圈的每个迭代对应于通过可缩放向量ISA的子向量支持指令所处理的向量的子向量中的一个子向量。由于其可免除在根据ISA将标量回圈转换回向量化可缩放码之前从向量化非可缩放码恢复为标量回圈的需求,因此这使码的编译和软件开发变得更加简单;代替地,将非可缩放ISA的向量指令直接映射成可缩放ISA的可缩放向量指令可更加简单,而不需要介入性的解向量化步骤(其它编译器可直接针对包括子向量处理指令的可缩放ISA来进行编译,而不需要使编译是以针对非可缩放ISA所编译的先前码为基础)。在编译时间下已知的子向量长度可与用于给定向量的向量长度无关,使得无论给定硬件具体实施针对整体给定向量所使用的实际向量长度为何,子向量长

度对给定软件的给定指令都相同。

[0026] 响应于子向量支持指令,指令解码电路可控制处理电路以响应于执行子向量支持指令的相同实例而处理子向量中的每个子向量。例如,在子向量粒度下执行的操作中的每个操作可平行地处理。替代地,在子向量粒度下执行的操作可循序地、或部分循序地且部分平行地、或以管线方式来处理。无论在子向量粒度下执行的各种操作的确切时序为何,在子向量粒度下执行的操作中的每个操作是响应于子向量支持指令的单一执行示例而执行,使得可实现向量ISA的SIMD益处。

[0027] 可使子向量长度在编译时间下已知的方法有若干种。对于可缩放向量ISA的一些具体实施,ISA可将子向量长度定义为可由软件码本身指定的可变软件定义参数,以允许软件在两个或更多个不同子向量尺寸之间选择。例如,这可有助于为重映射来自具有不同非可缩放向量长度的两个或更多个不同的非可缩放向量ISA的码提供支持。

[0028] 然而,在一个示例中,每个子向量可具有与用于给定向量的向量长度无关的架构定义固定尺寸的子向量长度。这可简化架构且避免软件指定子向量长度的任何需要。代替地,在可缩放向量ISA中定义的子向量支持指令的架构定义中,子向量长度可能是固定的。

[0029] 架构定义固定尺寸可对应于规定用于根据预定非可缩放向量ISA处理的向量指令的架构定义最大向量长度。例如,预定非可缩放向量ISA可能是由英国剑桥的Arm® Limited提供的“先进SIMD”架构(亦称为Neon™架构)。

[0030] 例如,子向量长度的架构定义固定尺寸可为128个位。这对于与定义128个位的最大向量长度的先进SIMD架构的兼容性是有用的。

[0031] 实施标定其它非可缩放向量ISA的子向量支持指令也是可行的,在该情形中,子向量长度可取决于所标定的特定ISA而变化。

[0032] 再者,以下是可行的:实施子向量支持指令,而不意图标定任何特定非可缩放向量ISA,但仅是为了选择给定固定子向量长度,而不带有任何试图仿真在特定非可缩放向量ISA中使用的长度的目的。即使未标定特定的非可缩放向量ISA,但以下仍是有用的:定义具有架构定义固定尺寸的子向量长度的子向量,以使软件开发者和编译器可利用取决于该固定尺寸的编译时间知识的软件性能优化。

[0033] 子向量支持指令可支持可变元素尺寸,使得每个子向量的每个向量元素具有从由可缩放向量ISA所支持的两个或更多个不同尺寸选择的可变元素尺寸。子向量长度可与针对每个子向量内的每个向量元素而使用哪个元素尺寸无关。因此,无论所选择的元素尺寸是较大元素尺寸或较小元素尺寸,可使用相同的子向量长度。向量元素的数目定义每个子向量可对应于子向量长度与所选择元素尺寸之间的比率。所选择元素尺寸可通过包括子向量支持指令的指令序列的软件指定参数来定义,并且因此在编译时间下是已知的。因此,由于子向量长度和元素尺寸都可在编译时间下是已知的,因此每个子向量提供的向量元素的数目在编译时间下也可能是已知的,但整体上给定向量中的向量元素的总数目在编译时间下可能是未知的,这是由于:根据可缩放向量ISA,给定向量的整体向量长度在编译时间下是未知的。

[0034] 在子向量粒度下执行的操作可有所变化。可将若干个不同子向量支持指令定义为可缩放向量ISA的部分,以便取决于程序设计师或编译器的选择而在子向量粒度下执行若干个不同操作。

[0035] 在一些示例中,对于至少一个子向量支持指令,在子向量粒度下执行的操作是针对每个子向量而对该子向量内的向量元素执行的操作,而与其它子向量中的元素无关。因此,这可允许在单一子向量支持指令中执行操作,该操作可在非可缩放向量ISA中已使用若干个单独的向量指令(各在具有在编译时间下已知的非可缩放向量长度的相应向量上操作)来执行(例如,所述单独的向量指令可对应于以非可缩放向量ISA的码所编写的回圈的不同迭代)。

[0036] 在一些示例中,对于至少一个子向量支持指令,在子向量粒度下执行的操作是针对子向量内的每个元素位置而对多个子向量中的每个子向量内的该元素位置处的相应向量元素执行的操作。这一指令可用以,例如,复制将已使用单一回圈迭代内的向量指令序列来执行的处理,其中该向量指令将已组合在经指定用于该向量指令序列的若干个向量操作数内的对应元素位置中的数据值。

[0037] 在一些示例中,对于至少一个子向量支持指令,在子向量粒度下执行的操作是用以设置预测值的所选择预测位的操作或是取决于该预测值的该所选择预测位而执行操作的操作,其中该所选择预测位是对应于向量的子向量尺寸部分的预测位。这可与可缩放向量ISA的可在小于子向量长度的个别向量元素的粒度下设置或解译预测的许多指令不同。这类子向量支持预测设置或预测相依指令可用于允许在某些情况下选择性地屏蔽整个子向量的处理,例如,在非可缩放向量ISA的对应码将已对具有在编译时间下已知的特定向量长度的向量执行的向量化回圈的整个迭代屏蔽掉的状况下。

[0038] 在一些示例中,可缩放向量ISA可支持子向量支持置换指令的至少一个变体。响应于子向量支持置换指令,指令解码器控制处理电路以针对向量结果的每个子向量将子向量设置成一个或多个向量元素的置换(permutation),该一个或多个向量元素是从至少一个向量操作数的对应定位子向量内的向量元素中选择的。子向量支持置换指令可能无法基于在向量操作数中的一个内的不同非对应子向量位置的向量元素的位而设置向量结果的给定子向量的向量元素。通过在子向量粒度下而非整体上在横跨给定向量的整个向量长度下执行置换,这可允许子向量支持置换指令的行为反映非可缩放向量ISA中定义的假设用于置换的已知向量长度的若干个单独的非可缩放置换指令的行为,同时仍使针对给定指令所执行的这类子向量粒度置换的数目可基于根据可缩放向量ISA在硬件中所支持的所实施向量长度而缩放。

[0039] 在一些示例中,可缩放向量ISA可支持子向量支持缩减指令的至少一个变体。响应于子向量支持缩减指令,指令解码器可控制处理电路在子向量粒度下执行至少一个缩减操作,每个缩减操作用以将操作数向量的多个向量元素缩减成结果内的单一数据值。由于不同子向量中的向量元素可能无法彼此组合,因此在个别子向量的粒度下执行这一缩减操作时,可对横跨整个向量长度执行的对应缩减操作提供不同结果。这类缩减可用以仿真在非可缩放向量化码的序列中可能已通过在给定回圈迭代或一系列回圈迭代内的多个指令的序列而实施的处理,该一系列回圈迭代各包括指令的单一示例以将向量操作数的每个元素与追踪任何先前回圈迭代中的类似组合的结果的累加器值的对应元素组合。

[0040] 子向量支持缩减指令的不同变体是可能的,该变体在子向量粒度下执行缩减的方式有所不同。

[0041] 例如,以子向量内(intra-sub-vector)的子向量支持缩减指令来说,对于每个缩

减操作,多个向量元素包括在操作数向量的对应子向量内的相应向量元素。将这一指令包括在ISA中可用以允许软件开发者使用该指令在编译时间下具有未知向量长度的可缩放架构中仿真横跨单一向量操作数的所有向量元素执行缩减的非可缩放码的行为。可将每个子向量粒度缩减的结果放置在结果值的不同子向量中。替代地,子向量内的子向量支持缩减指令的变体可将每个子向量粒度缩减的结果放置在结果值的一个或多个子向量的相应向量元素中。

[0042] 在另一变体中,以子向量间(inter-sub-vector)的子向量支持缩减指令来说,对于每个缩减操作,多个向量元素包括在操作数向量的多个子向量内的对应元素位置处的向量元素。对于此类型的指令,经缩减到单一结果的向量元素是以子向量长度的间隔所选择的向量元素。将此指令包括在ISA中可用以允许软件开发者使用该指令在编译时间下具有未知向量长度的可缩放架构中仿真横跨在若干个连续回圈迭代中经处理的向量内的相同元素位置处的向量元素执行缩减的非可缩放码的行为。

[0043] 可缩放向量ISA可支持这些类型的子向量支持缩减指令中的任一个或多个子向量支持缩减指令。

[0044] 在一些示例中,可缩放向量ISA可支持子向量支持加载/存储指令的至少一个变体。响应于子向量支持加载/存储指令,指令解码器可控制处理电路执行加载/存储操作,以在子向量粒度下于存储器系统与至少一个向量寄存器之间转移一个或多个子向量。这可用以仿真非可缩放向量ISA的将已对具有已知向量长度的向量执行对应加载/存储操作的加载/存储指令的行为。

[0045] 子向量支持加载/存储指令可能是与预测值关联的预测指令(predicated instruction)。响应于子向量支持加载/存储指令,指令解码器可控制处理电路以基于在子向量粒度下从预测值选择的预测位而控制执行或屏蔽一个或多个子向量的每次转移。这可与可缩放向量ISA的可使用在用于向量的向量元素的元素尺寸(该元素尺寸小于子向量长度)的粒度下所选择的预测来横跨整个(可缩放)向量长度执行加载/存储操作的其它加载/存储指令的行为不同。

[0046] 在一些示例中,可缩放向量ISA可支持子向量支持递增/递减指令的至少一个变体。响应于子向量支持递增/递减指令,指令解码器可控制处理电路以基于向量中有多少个子向量尺寸部分是通过在子向量粒度下从预测值选择的该预测值的位指示为有效而递增或递减操作数值。这可有助于回圈控制,使得由软件使用以决定是否仍需要执行至少一个进一步回圈迭代的回圈控制变数可根据在回圈的最末迭代中处理的子向量的数目递增或递减(由于在回圈迭代中处理的子向量的数目在编译时间下可能是未知的,因此提供促成推断所处理的子向量的数目并因此将该数目用以更新回圈控制变数的指令可能是有用的)。

[0047] 由子向量支持递增/递减指令使用以确定如何更新操作数值的预测值可能是下列中的一个:由该子向量支持递增/递减指令指定作为预测操作数的预测值;和由所述子向量支持递增/递减指令指定的预测模式识别符所表示的预测值,该预测模式识别符在子向量粒度下指定预测位的预定模式。

[0048] 在一些示例中,可缩放向量ISA可支持子向量支持预测设置指令的至少一个变体。响应于子向量支持预测设置指令,指令解码器可控制处理电路执行预测设置操作以在子向

量粒度下设置预测值的位,以指示向量中有哪些子向量是有效的。这一指令可用以控制哪些子向量是由其它子向量支持预测指令所处理。经设置成特定值(例如,0)的预测值的位可导致对应子向量的处理被屏蔽。这一预测设置指令可能是与可缩放向量ISA的其它预测设置指令不同的指令,该指令可在可小于子向量长度的向量元素尺寸的粒度下设置预测值。

[0049] 用于子向量支持预测设置指令的预测设置操作可包括基于下列中的一个来设置预测值:预测模式识别符,其指定要在子向量粒度下施加的预测位的预定模式;和子向量粒度比较操作,其是基于第一操作数与第二操作数的比较。使用此方法时,预测的新值并未在子向量支持预测设置指令中明确地指定,但可根据可针对不同向量长度而缩放的某个一般模式来定义。这是有用的原因在于:由于可缩放向量长度的缘故,待设置的预测位的数目在编译时间下可能是未知的。

[0050] 于上文描述子向量支持指令的各种示例。应理解可缩放向量ISA的任何给定具体实施不需要支持所有这些类型指令。可实施在本申请中描述的指令中的任一个或多个指令。

[0051] 上文讨论的技术可实施在数据处理设备内,该数据处理设备具有针对实施上文讨论的指令解码器和处理电路而提供的硬件电路。

[0052] 然而,相同技术还可实施在计算机程序内,该计算机程序是在主机数据处理设备上执行以提供用于目标码的执行的指令执行环境。即使主机数据处理设备本身不支持所述架构,这一计算机程序也可控制主机数据处理设备以模拟将提供在实际上支持根据可缩放向量指令架构的目标码的硬件设备上的架构环境。计算机程序可具有仿真上文讨论的指令解码电路的功能的指令解码程序逻辑。例如,指令解码程序逻辑响应于目标码的给定指令而以主机数据处理设备的原生指令集来产生对应码序列,以控制主机数据处理设备执行经解码指令的对应功能。指令解码程序逻辑包括子向量支持指令解码程序逻辑,用以如上文讨论地解码子向量支持指令,以控制主机数据处理设备在子向量粒度下针对给定向量执行操作。例如,当针对一个指令集架构所编写的传统码是在支持不同指令集架构的主机处理器上执行时,这一模拟程序可能是有用的。并且,由于软件在模拟执行环境上的执行可使软件测试得以与支持新架构的硬件装置的进行中开发平行,模拟可允许针对较新版本的指令集架构的软件开发在支持该新架构版本的处理硬件就绪之前开始。模拟程序可存储在存储介质上,该存储介质可为非暂态存储介质。

[0053] 现在参照图式描述特定示例。应理解权利要求书不限于这些特定示例。

[0054] 图1示意地例示了数据处理设备2的示例。数据处理设备具有包括若干个管线级的处理管线4。在此示例中,管线级包括用于从指令高速缓存8提取指令的提取级6;用于解码经提取程序指令以产生待由管线的其余级处理的微操作(micro-operation)(经解码指令)的解码级10(指令解码电路的示例);用于检查微操作所需的操作数在寄存器14中是否可用且在一旦用于给定微操作的所需操作数可用时发布用于执行的微操作的发布级12;用于通过处理读自寄存器14的操作数以产生结果值来执行对应于微操作的数据处理操作的执行级16(处理电路的示例);和用于将处理的结果写回到寄存器14的写回级18。应了解这仅是可能的管线布置的一个示例,并且其它系统可能具有额外级或不同的级配置。例如,在乱序处理器中,可包括寄存器重命名级,以用于将由程序指令或微操作指定的架构寄存器映射到识别寄存器14中的实体寄存器的实体寄存器识别符。在一些示例中,在由解码级10解码

的程序指令与由执行级所处理的对应微操作之间可有一对一关系。在程序指令与微操作之间也可能有一对多或多对一关系,使得,例如,单一程序指令可分成两个或更多个微操作,或者两个或更多个程序指令可融合以作为单一微操作来处理。

[0055] 执行级16包括若干个处理单元,以用于执行不同类别的处理操作。例如,执行单元可包括标量处理单元20(例如,包括用于对读自寄存器14的标量操作数执行算术或逻辑运算的标量算术/逻辑单元(ALU)20);用于对包括多个向量元素的向量执行向量操作的向量处理单元22;和用于执行加载/存储操作以存取存储器系统8、30、32、34中的数据的加载/存储单元28。可提供在执行级的处理单元的其它示例可包括用于执行涉及以浮点格式表示的值的操作的浮点单元,或用于处理分支指令的分支单元。

[0056] 寄存器14包括用于存储标量值的标量寄存器25、用于存储向量值的向量寄存器26和用于存储预测值的预测寄存器27。在处理向量指令时,预测值27可由向量处理单元22使用,其中在给定预测寄存器中的预测值指示存储在向量寄存器26中的对应向量操作数有哪些向量元素是有效向量元素或无效向量元素(其中对应于无效数据元素的操作可受抑制,或可不影响响应于向量指令而由向量处理单元22产生的结果值)。

[0057] 存储器管理单元(MMU)36基于在存储于存储器系统中的页表结构中所定义的地址映射而控制虚拟地址(其是由来自提取电路6的指令提取或来自加载/存储单元28的加载/存储请求来指定)与识别存储器系统中的位置的实体地址之间的地址转换。页表结构还可定义存储器属性,该存储器属性可指定用于存取地址空间的对应页的存取权限,例如,指定地址空间的区域是只读或是可读/可写的、指定允许存取该区域的特权级和/或指定其管理地址空间的对应区域可如何存取的其它性质。来自页表结构的条目可高速缓存在转换后备缓冲器(TLB)38中,该转换后备缓冲器是由用于高速缓存页表条目或其它信息以用于加速对来自示出在存储器中的页表结构的页表条目的存取的MMU 36维持的高速缓存。

[0058] 在此示例中,存储器系统包括一级数据高速缓存30、一级指令高速缓存8、共享二级高速缓存32和主系统存储器34。将理解这仅是可能的存储器阶层的一个示例,并可提供高速缓存的其它布置。示出在执行级16中的处理单元20至28的具体类型仅是一个示例,并且其它具体实施可具有不同组的处理单元或可包括相同类型的处理单元的多个示例,使得可平行地处理多个相同类型的微操作。应理解图1仅是可能的处理器管线布置的一些组件的简化表示,并且处理器可包括为了简洁起见而未例示的许多其它元件。

[0059] 处理管线4支持可缩放向量ISA,其意味着ISA的向量指令可受处理,而无需在编译时间(编译器编译所执行指令的时间)下知道将用于执行所述向量指令的向量长度为何。这使支持不同最大向量长度的各种不同处理设备得以执行相同的软件码,以避免在产生适用于横跨一系列处理平台的执行的软件码时造成软件开发者的负担。这意味着给定处理设备的硬件设计者具有取决于设计者偏好的性能/电力需求而选择所实施的最大向量长度的自由(追求较高性能的系统可选择比追求优选能源效率的系统更长的最大向量长度)。

[0060] 图2例示了可缩放向量ISA的架构状态的示例。ISA将若干个可缩放向量寄存器(例如,32个向量寄存器,Z0至Z31)定义为向量寄存器26,该若干个可缩放向量寄存器具有针对给定硬件具体实施的可最多是特定架构定义最大长度的特定单位尺寸的任何倍数的具体实施选择向量长度。例如,在此示例中,单位尺寸是128个位,并且架构支持在128个位与2048个位之间的任何128个位的倍数的最大向量长度(即,给定处理器的向量长度是LEN*

128个位,其中1≤LEN≤16)。举例来说,这与寄存器可具有128个位的固定架构定义向量长度的非可缩放向量架构不同(见图2中标记为V0至V31的寄存器)。图2所示的示例是基于可缩放向量ISA是由Arm®Limited提供的“可缩放向量延伸(Scalable Vector Extension,SVE)”,以及非可缩放向量ISA是由Arm®Limited提供的“先进SIMD”架构(Neon™)。即使根据可缩放向量ISA支持大于128个位的向量长度,指令解码器10和处理电路16还可出于回溯兼容性的原因而支持非可缩放向量ISA,并且非可缩放向量ISA的指令可参考固定长度(例如,128个位)的寄存器V0至V31;实际上,这些寄存器可使用可为可缩放向量ISA的指令存取的较长寄存器的一部分来表示,使得可缩放向量寄存器Z0至Z31和非可缩放向量寄存器V0至V31可共享硬件存储电路。

[0061] 当然,其它示例可基于不同的可缩放和非可缩放架构,并且因此可用于可缩放向量长度的选择的尺寸范围和经指定以用于非可缩放向量长度的固定尺寸可与图2所示者不同。

[0062] 除了向量寄存器26以外,提供若干个预测寄存器27(标记为P0至P15),以用于存储用以选择性地屏蔽对使用向量寄存器26所提供的向量操作数的向量元素执行的操作的预测值。当使用硬件中所支持的最大向量长度时(此最大向量长度是指由特定硬件具体实施使用的所选择尺寸LEN*128,而不是由用于任何硬件具体实施的架构所允许的最大向量长度(例如,2048个位)),预测寄存器27在根据由ISA所支持的最小向量元素尺寸定义时可于向量寄存器26中每向量元素便具有一个位。例如,如果最小元素尺寸是8个位,则对于LEN*128位向量寄存器的示例来说,每个预测寄存器可具有尺寸LEN*16个位。图3示出在逐元素(element-by-element)的基础上施加到向量的预测的示例。指令可定义可变元素尺寸,例如,图3示出可如何将256位向量逻辑地划分成64位或32位元素(其它元素尺寸也可是可能的,例如,16或8位元素)。在使用最小元素尺寸时,指定用于存储在向量寄存器26中的给定向量的预测寄存器27指定足以能够为每个向量元素指定单独预测位的若干个预测位,但在较大的元素尺寸下,并非所有预测位都需要读取,并且代替地,预测位可在对应于元素尺寸的间隔下读取。例如,对于64位元素,可每八个预测位读取一个,如图3的顶部示例所示,而对于32位元素,可每四个预测位读取一个(在8位元素尺寸下,每个预测位都可被读取)。当将一个预测位设置成1时,则将对应元素视为是有效的,而具有设置成0的对应预测位的向量元素则视为是无效的。可基于向量操作的结果将结果向量的对应于无效向量元素的元素屏蔽掉而不更新(预测指令可包括将经屏蔽掉的结果元素设置成0的归零预测变体,或可包括在执行向量指令之前将经屏蔽掉的结果元素设置成与目的地寄存器内的对应元素相同的值的合并预测变体)。

[0063] 如图2所示,寄存器14还可包括一些控制寄存器29(标记为ZCR),所述控制寄存器可被更高特权软件用来限制可由在较低特权状态下执行的软件使用的最大向量长度。例如,为了节省电力,可限制给定软件,使得其无法利用硬件中所支持的全向量长度。然而,即使更高特权软件施加对于向量长度的限制,用于应用程序软件的向量长度在编译时间下仍是未知的,这是由于无法得知在特定处理器中所实施的实际向量长度将大于或小于在控制寄存器29中所定义的限制(对于具有比在控制寄存器29中所定义的限制小的最大向量长度的具体实施,将使用比该限制所指示者小的向量长度)。

[0064] 当考虑到规模从相对小能源效率微控制器到服务器和其它高性能计算系统的广

泛范围的需求时,由于在可用于编码ISA的指令的固定编码空间内为处理器设计者可能要求的每个不同向量长度产生不同指令是不可行的,因此可缩放向量ISA的向量长度未知性质是有用的。通过在编译时间下不具有已知的固定向量长度,可使用相同ISA来满足多个市场,而不需要软件开发者针对每个性能/电力/面积点来量身定制码的工作量。

[0065] 为达成可缩放向量ISA的可缩放性质,可缩放向量ISA的向量指令的功能性是参考指示所使用的向量长度的参数(例如,如图2所示的LEN)来定义于架构中(当考虑到在硬件中所支持的最大向量长度和使用控制寄存器29的任何软件定义限制时),其中该参数LEN在编译时间下是未知的。因此,在不同系统上执行相同的向量指令可产生不同结果(其在所产生的向量元素的数目上一般会有所变化,其子集在不同平台上可具有相同结果值,但与实施较小向量长度的平台相比,实施较大向量长度的平台通常可产生额外的向量元素)。预测可用以控制在指令的给定示例中产生哪些元素,并且可基于向量长度未知原理而设置,例如通过使用比较指令以自动产生用于特定回圈迭代的预测的值或施加可缩放到不同向量长度的某个一般定义预测模式。某些指令可更新回圈控制参数(例如,元素计数值)以追踪目前为止已处理多少个向量元素,使得整体上在横跨一回圈的情况中,具有较宽或较窄向量长度两者的具体实施最终可达成相同结果,但性能等级则不相同,这是因为具有较宽向量长度的具体实施可能比具有较窄向量长度的具体实施需要更少的回圈迭代。

[0066] 图4示出使用不同向量长度来呈现相同的指令序列可如何在不同硬件具体实施上以不同方式处理的工作示例。在此示例中,该码示例是实行双精度浮点操作‘daxpy’(双精度Ax加y)以为二个输入值阵列x[i]、y[i]计算y[i]＝a*x[i]+y[i],其中0≤i<n。这可使用如下的来自可缩放向量ISA的一组指令来实施于向量化回圈中:

[0067]

```
1 // x0 = &x[0], x1 = &y[0], x2 = &a, x3 = &n
2 daxpy_:
3    ldrsw x3, [x3]                          // x3=*n
4    mov x4, #0                              // x4=i=0
5    whilelt p0.d, x4, x3                    // p0=while(i++<n)
6    ld1rd z0.d, p0/z, [x2]                  // p0:z0=broadcast(*a)
7 .loop:
8    ld1d z1.d, p0/z, [x0, x4, lsl #3]       // p0:z1=x[i]
9    ld1d z2.d, p0/z, [x1, x4, lsl #3]       // p0:z2=y[i]
10   fmla z2.d, p0/m, z1.d, z0.d             // p0?z2+=x[i]*a
11   st1d z2.d, p0, [x1, x4, lsl #3]         // p0?y[i]=z2
12   incd x4                                 // i+=(VL/64)
13.latch:
14   whilelt p0.d, x4, x3                    // p0=while(i++<n)
15   b.first .loop                           // more to do?
16 ret
```

[0068] 图4示出针对一个示例使用128位向量长度和针对另一示例使用256位向量长度的这些指令的处理。变量a和阵列x[]和y[]是存储在存储器中的数据。寄存器x0至x4是标量寄存器。寄存器p0是预测寄存器。寄存器z0至z2是向量寄存器,其在第一示例中是128位寄存器,并且在第二示例中是256位寄存器(因此,相较于128位示例,预测寄存器p0在256位示

例中是较长的寄存器)。实线轮廓框指示由给定指令所处理或设置的有效元素,而虚线轮廓框指示由预测所屏蔽的无效元素或完全不用于给定指令的寄存器。应注意到,示出于图4中的指令编号指示由处理电路16执行的经解码指令序列的程序顺序(其与上文所示的经编译码的编号不同,因为一些指令可执行多次,并且在上述示例中经编号的分支点标记在图4中并未编号)。然而,图4中示出与上文所示相同的指令序列。

[0069] 在128位和256位示例两者中,向量指令是针对可适配在对应向量长度内的向量元素的数目而缩放。在此示例中,所使用的元素尺寸是64个位,使得在128位示例中,每向量有2个向量元素,并且在256位示例中,每向量有4个向量元素。因此,在128位示例中仅使用两个p0的预测位,并且在256位示例中仅使用四个p0的预测位(实际上,寄存器p0可包括更大数目的预测位,以使得更小的向量元素尺寸受到支持)。

[0070] whilelt、incd和b.first指令是根据所支持的向量长度来设置预测和更新回圈计数值且取决于回圈计数值的比较而有条件地分支回到回圈的开始的回圈控制指令。其它指令是根据所支持的向量长度来处理若干个向量元素的加载/存储指令或向量处理指令。

[0071] 递增指令incd基于在对应回圈迭代中处理的向量元素的数目而递增回圈计数器(例如,VL/ES,其中VL是向量长度,并且ES是向量元素尺寸)。预测设置指令whilelt使用回圈计数器(i,在指令序列中由寄存器x4表示)与终止限制(n,由寄存器x3表示,其是通过加载指令从存储器在上述码示例的第3行加载)之间的比较,以取决于回圈计数器的各种递增版本是否小于(lt)限制值而设置预测。对于递增回圈计数器仍小于限制值的所述元素,将预测设置为真,而一旦递增回圈计数器达到终止限制时,则将预测设置为伪(在此示例中,这在256位示例中发生于指令2所处理的第四元素上,但对于128位示例中的2元素向量,这限制在第一回合(pass)时尚未达到且将仅由经执行以确定仍有另一向量元素要处理(见图4的128位示例所示的在经解码程序顺序下的第9指令)的后续预测设置指令(示出在上述码示例中的第14行的whilelt指令)来达到,因此分支指令(在经解码程序顺序下的指令10)返回回圈以进行另一回合。在下一回合时,whilelt指令(在经解码程序顺序下的第16指令)现在确定所有预测都是“伪”,并且因此分支指令此时不采取分支,因而回圈结束。相反地,对于256位示例,仅需要一回合便可将每个向量寄存器中的第四元素的效果屏蔽掉。

[0072] 应注意到由指令序列产生的结果在128位和256位示例中最终都相同(存储到存储器以用于阵列y[]的数据具有相同值3、50、41、32(从最高有效地址到最低有效地址),但128位示例在两个回圈迭代中才获得其结果,而256位示例仅需要一个回圈迭代。因此,在128位示例中,上述码示例中在第8至15行的指令需要解码且执行两次(见图4中的128位示例的在经解码程序顺序下编号的指令4至17)。

[0073] 虽然可缩放向量ISA对于促成开发支持一系列不同硬件具体实施的平台独立码可能是非常有用的,然而许多使用中的软件程序已特别针对支持固定长度向量单元的非可缩放向量ISA来优化,例如针对与Arm®Limited的Neon™非可缩放向量ISA兼容的系统所提供的128位向量单元。对于回溯兼容性,支持可缩放向量ISA的系统还可支持非可缩放向量ISA的指令,以确保这类传统软件仍可执行。然而,这意味着:即使硬件支持更大的向量长度,非可缩放软件仍无法从硬件中可产生的额外性能获益。

[0074] 使用可缩放向量ISA的指令来重新开发以非可缩放向量ISA所编写的程序码可为所要的,因为这可开启更大的性能机会,进而利用在可缩放向量ISA的许多硬件具体实施上

可用的较长向量长度。然而,如果尝试将针对非可缩放向量ISA所编写的程序(其中向量长度在编译时间下是已知的)重编写成针对可缩放向量ISA所编写的程序码(其中向量长度在编译时间下是未知的),则这可能需要程序设计师或编译器编写者的可观工作量。由于呈非可缩放ISA形式的程序码常可包括取决于在编译时间下已知的固定向量长度(例如,128个位)的知识而选择的特定优化,因此非可缩放向量ISA的指令可能无法以简单方式映射到可缩放向量指令,并且如果无法得知接下来在编译时间下将使用的向量为何,则这可能会防止一些这类优化的使用,其在一些情形中可完全防止呈可缩放向量化码形式的向量化回圈的使用。

[0075] 例如,下列C语言码可实施部分总和缩减(例如,这可能是来自数字信号处理应用程序的操作):

```
for (int biquad = 0; sample < num_samples; ++sample) {
    tmp = acc;
    acc += (((int32_t) coeff[biquad].b1 * state[biquad].x1 +
        (int32_t) coeff[biquad].b2 * state[biquad].x2) >> 14) +
        (((int32_t) coeff[biquad].a1 * state[biquad].y1 +
        (int32_t) coeff[biquad].a2 * state[biquad].y2) >> 14);
    state[biquad].x2 = state[biquad].x1;
    state[biquad].x1 = tmp;
    state[biquad].y2 = state[biquad].y1;
    state[biquad].y1 = acc;
}
```

[0076]

[0077] 在上述码中,每个迭代中的乘加(multiply-add)操作的结果在acc中累加,并存储在state[biquad].y1中。在经过向量化之后,用以存储state[biquad].y1的值的向量的每个元素应存储累加的部分总和(元素0应在第0次迭代中存储acc的值,元素1应在最初两个迭代之后存储acc的所得值,依此类推)。

[0078] 在非可缩放向量ISA上,此部分总和缩减可通过将四个元素中的每个元素广播到新向量寄存器中并使用mla(乘累加,multiply-accumulate)指令来部分累加这四个向量寄存器而达成,如下文所示:

[0079] …

[0080] dup v17.4h,v16.h[0]

[0081] dup v18.4h,v16.h[1]

[0082] dup v19.4h,v16.h[2]

[0083] dup v20.4h,v16.h[3]

[0084] add v 6.4h,v6.4h,v17.4h

[0085] mla v 6.4h,v18.4h,v29.4h

[0086] mla v 6.4h,v19.4h,v30.4h

[0087] mla v 6.4h,v20.4h,v31.4h

[0088] …

[0089] 此处,第一复制(dup)指令将向量寄存器v17的所有向量元素设置成等于向量寄存器v16的元素0,第二dup指令将向量寄存器v18的所有向量元素设置成等于向量寄存器v16

的元素1,依此类推。寄存器v16每向量元素有一个dup。由于非可缩放向量ISA中的向量长度在编译时间下是已知的,则有鉴于向量元素的尺寸,编译器可找出要包括多少个dup指令(在此示例中是四个,其是基于每128位向量有4个32位元素)。

[0090] 然而,在可缩放向量ISA中,此部分总和缩减无法以相同方式执行,这是由于在编译时间下的向量长度是未知的,并且因此编译器不知道应该包括多少个dup指令。这可能阻碍以C语言码定义的标量回圈的成功向量化,并且可能强制可缩放向量码采用标量回圈,使得向量化的益处无法实现。

[0091] 这仅是可能依赖向量长度的编译时间知识的软件优化的一个示例。其它示例可包括:回圈展开,其通过将某一个数目的原始回圈迭代映射到各包括较大数目的指令的较小数目的回圈迭代而减少需要执行的回圈控制指令的数目,其中"经展开"回圈的一个迭代对应于原始回圈的多个迭代;和软件管线化,其中编译器使回圈的指令的执行重新排序,使得稍后回圈迭代的一些指令可在来自较早回圈迭代的指令之前执行。

[0092] 图5示意地例示可帮忙此问题的技术的示例。具有(在编译时间下是未知的)可缩放向量长度的向量被逻辑地视为包括可缩放数目的子向量,每个子向量具有相等子向量长度。该子向量长度在编译时间下是已知的。例如,每个子向量可具有对应于非可缩放向量架构(例如,Neon™)中的固定向量长度的固定尺寸(例如,128个位),或可单纯是在可缩放向量ISA中定义的任意固定尺寸(不论与现有非可缩放向量ISA有任何对应)。替代地,在可缩放向量ISA的一些具体实施中,软件可有从两个或更多个选项中指定子向量长度的能力,使得子向量长度在编译时间下是已知的(经软件定义),但可针对利用此子向量方法的指令的不同示例而有所变化。由于子向量长度在编译时间下是已知的,但整体向量长度并非已知的,因此适配在向量内的子向量的总数目在编译时间下也是未知的,并且可取决于针对执行经设计用以支持子向量方法的指令的特定硬件具体实施而选择的特定向量长度而可缩放。每个子向量具有可变数目的向量元素,并且每个向量元素的尺寸可能是可变的且由软件从若干个选项(例如,8、16、32、或64个位)中选择。因此,虽然每子向量的向量元素的尺寸和数目在编译时间下可能是已知的,但在整个向量中的数目效应元素可能是可变的且在编译时间下是未知的。子向量长度可与由硬件使用的整体向量长度和针对给定指令而指定的向量元素尺寸两者都无关。

[0093] 使用此方法时,由于子向量长度是已知的,因此将用于与非可缩放向量ISA一起使用而编译的码(假设固定向量长度)转换成用于与可缩放向量ISA一起使用而编译的码(使用假设已知子向量长度的子向量的子向量支持指令,但允许可缩放整体向量长度)变得更加简单。此外,即使针对支持子向量支持指令的可缩放向量ISA而直接进行编译(不从非可缩放向量化码开始),子向量支持指令可用以允许施加软件性能改进技术(例如上文讨论的那些),否则所述软件性能改进技术将不可能用于在可缩放长度的向量中在逐元素或整体向量的粒度下操作而不是在固定尺寸的子向量粒度下操作的可缩放向量指令。

[0094] 因此,如以下示例中所讨论,并且可定义控制处理电路16(例如,向量处理单元22和/或加载/存储单元28)在子向量的粒度下而不是在个别元素的粒度或整体向量的粒度下执行操作的子向量支持指令的数目)。在子向量的粒度下执行的操作可响应于执行子向量支持指令的单一示例而平行地、循序地、部分平行地且部分循序地、或以管线化方式来执行(因此,不需要使用设置在执行子向量支持指令的相应示例之间的预测值将向量分割成子

向量,其中每个子向量经由子向量支持指令在单独回合中受处理)。

[0095] 提供对应于可能期望对向量操作数执行的所有向量操作的子向量支持指令并非必需的。许多操作(例如,加或乘)可在逐元素粒度下施加,并且因此即使在施加到经设计用以支持图5所示的向量的向量方法(vector-of-vectors approach)的操作数时也可提供正确结果。因此,对于某些类型的处理操作,子向量粒度指令可能是不可行的。然而,对于一些类型的处理操作(例如一般会横跨整个向量所实施的置换或缩减),定义在子向量的粒度下施加对应操作的子向量支持指令以提供不同处理结果可能是有用的。类似地,实施某些类型的在子向量粒度下操作的加载/存储指令和回圈控制或预测设置指令(例如,通过在子向量的粒度下而不是在个别元素的粒度下设置或读取预测位)可能是有用的。通过将这类指令包括在可缩放向量ISA中,这可使重新开发先前针对非可缩放向量ISA所优化的程序码以使用可缩放向量ISA变得更加直接,进而在使用由可缩放向量ISA所支持的较大向量长度的较高端处理器具体实施上执行时,使码得以达成优选性能。

[0096] 图6示出例示了子向量支持指令的处理的流程图。在步骤100,支持可缩放向量ISA的指令解码器10解码正在执行的程序的下一个指令。在步骤102,指令解码器10确定所解码指令是否是子向量支持指令。若否,则在步骤104,指令解码器10控制处理电路16执行如由其它类型的指令所指示的处理操作。如果由指令解码器10解码的指令是子向量支持指令,则在步骤106,指令解码器10控制处理电路16在子向量粒度下针对视为包括两个或更多个子向量的给定向量执行操作,该两个或更多个子向量各包括某一个数目的向量元素,其中每个子向量具有相等子向量长度。该子向量长度在编译时间下是已知的。然而,整体向量长度在编译时间下是未知的,因此由指令所处理的子向量的数目在编译时间下是未知的。

[0097] 图7至图11示出用于将置换操作施加在向量操作数的每个子向量内的子向量支持置换指令的各种示例。在每个情形中,用于指令的(多个)输入操作数和结果是被视为经逻辑地划分成若干个子向量的向量,该子向量各具有在架构中是固定的或否则在编译时间下是已知的子向量长度(例如,128个位)。向量操作数和结果的整体长度是可缩放的且在编译时间下是未知的。为方便解释起见,下文讨论的所有示例示出具有32位元素的示例,使得每子向量有四个向量元素,但应理解其它示例可使用不同的元素尺寸。对于每个置换指令,将结果的给定子向量内的所有元素设置成从一个或多个操作数向量的对应子向量选择的位的置换。取决于从操作数向量的其它子向量中的元素选择的位而设置结果的给定子向量中的给定元素是不可能的,与结果的该给定子向量相比,该元素是在一不同的相对子向量位置处。因此,将对应置换操作在子向量粒度下执行多次。

[0098] 图7示出复制置换指令DUPQ的第一示例,其中施加在结果Zd的给定子向量中的置换是要将操作数向量Zn的对应子向量的经选择元素复制到结果的给定子向量的向量元素中的每个向量元素。将复制哪个元素是由定义相对于子向量的开始的元素位置的立即索引值#imm来指示。例如,在图7中,立即值具有1的值,该值指示操作数Zn的每个子向量的元素1将会被复制到结果Zd的对应子向量中的所有向量元素位置。因此,将值A1复制到Zd的前四个元素位置,将值B1复制到次四个元素位置,将值C1复制到次四个元素位置,依此类推(因为在此示例中每个子向量包括四个向量元素)。使用置换操作所处理的子向量的数目将取决于如可缩放向量ISA所允许的由硬件使用的特定向量长度。然而,由于将被复制的给定向量元素的示例的数目在编译时间下是已知的(例如,在此情形中为4),因此仍可使用码优化

(例如上文所示的用于部分总和缩减的该一者),以使其更易于采用可缩放向量ISA,同时仍允许依赖在编译时间下的(子)向量长度的知识的码优化。

[0099] 图7的底部部分示出如果在整体向量的粒度下施加时的对应复制置换,如常规向量指令所可能预期。在此情形中,立即索引值将定义相对于向量操作数Zn的开始的向量元素位置,并且在向量操作数Zn的此位置处的向量元素将会被复制到结果向量Zd的每一个元素。这清楚地示出使用置换指令的子向量支持变体所达成的结果在相较于整体向量置换指令的情况下的差异。

[0100] 图8示出子向量支持置换指令EXTQ的另一示例。在此示例中,施加到每个子向量的置换是提取置换,用以将结果Zdn'的每个子向量的下部分设置成第一向量操作数Zdn的对应子向量的较高位,并且将每个子向量的上部分设置成第二向量操作数Zm的对应子向量的较低位。在此示例中,编码是破坏性的,使得结果被写入到用以提供第一向量操作数的相同寄存器Zdn。其它示例可使用建构性编码,其定义另一向量寄存器以提供与用以存储结果的目的地寄存器不同的第一向量操作数。立即索引值定义从两个操作数向量提取的部分的尺寸(例如,索引可定义相对于子向量的开始的位置,用于待复制到结果的对应子向量的较低位的第一操作数的上部分是在该位置提取,并且结果的对应子向量的剩余位可使用从第二操作数的最低有效端选择的位来填充)。在此特定示例中,立即值在字节粒度(以8位为单位)下选择位位置,使得将给定结果子向量的位[127-(imm*8):0]设置成等于第一操作数Zdn中的对应子向量的位[127:imm*8],并且将给定结果子向量的位[127:128-(imm*8)]设置成等于第二操作数Zm的位[(imm*8)-1:0]。应理解其它示例可定义立即值以在8位以外的其它单位的递增下选择。此置换是在子向量粒度下分别施加,但其中每个子向量置换使用表示从两个操作数的对应子向量提取的部分之间的边界的相同立即值。同样地,图8示出在子向量粒度下施加此操作与整体向量置换有多不同。

[0101] 图9示出如果表查找置换是在逐子向量(sub-vector by sub-vector)的基础上施加的子向量支持置换指令TBLQ的另一示例。在此示例中,第一向量操作数Zm定义一组索引值,该组索引值指示第二向量操作数Zn的给定子向量有哪些元素要被复制到结果向量Zd的对应子向量内的对应元素位置。索引值是相对于对应子向量的开始而不是相对于整个向量来定义。因此,对于第二子向量(对应于第二操作数Zn中的值B0、B1、B2、B3),元素索引3、3、1、2针对结果中的对应子向量的对应向量元素而选择值B3、B3、B1、B2(B3是放置在子向量的最低有效元素,并且B2是放置在最高有效元素中)。如果特定索引值具有大于在给定子向量中的最大向量元素索引的值,则将零写入到结果向量Zd内的对应位置(例如,见用于第三子向量中的最高有效向量元素位置的索引值6的示例,其大于在给定子向量中的4个元素(标记为0至3)的最高有效索引3,因此将0写入到结果Zd中的对应位置)。同样地,每子向量置换的此方法不同于整体向量置换,其索引值将相对于整体向量来定义,并且因此将整体向量置换施加到图9所示的两个相同的操作数Zm、Zn会将不同结果提供给在子向量粒度下执行的操作,如图9的下部分所示。

[0102] 图10示出子向量支持置换指令ZIPQ1的另一示例,其中在此情形中,施加在给定子向量中的置换是要将来自两个向量操作数Zn、Zm的对应子向量的下半部的元素交错在结果向量Zd的给定子向量内。此置换是针对子向量中的每个子向量来执行,并且同样地将不同结果提供给类似置换是在整体向量粒度下执行的情形,如图10的下部分所示。虽然图10示

出将来自每个子向量的下半部的元素交错的示例,但还可将对应指令ZIPQ2另外定义成将来自每个子向量的上半部的元素交错。

[0103] 图11示出子向量支持置换指令UZPQ1的另一示例,其中在此情形中,施加在给定子向量中的置换是要将两个向量操作数Zn、Zm的对应子向量的偶数编号向量元素在结果向量Zd的对应子向量内串接。同样地,这产生具有与如果类似置换是在整体向量的粒度下施加时所会达成的不同顺序的元素的结果。此指令的替代版本UZPQ2可串接奇数编号元素,而不是偶数编号元素。

[0104] 图12示出在子向量粒度下实行减缩操作的子向量间的缩减指令的示例,每个缩减操作用以将操作数向量的多个元素缩减成结果向量内的单一数据值。此示例示出预测指令,其中指令指定与操作数向量Zn关联的预测值Pg以指示哪些元素被屏蔽。经屏蔽元素不会促成缩减结果。对于此示例,缩减是横跨每个子向量内的对应编号元素来执行,并且在此示例中施加的缩减运算符是加法。因此,将结果向量Zd的元素0设置成在向量操作数Zn的子向量中的每个子向量内的位置0处的任何未屏蔽元素的总和,将结果向量Zd的元素1设置成在向量操作数Zn的子向量中的每个子向量内的位置1处的任何未屏蔽元素的总和,其它元素位置则依此类推(如果在每个子向量内的给定位置处的所有元素都被屏蔽,则将结果的对应元素设置成0)。由于缩减操作使元素的总数目减少,因此可将结果向量Zd的未填入缩减结果的剩余元素填入零。

[0105] 虽然图12示出具有加法缩减的示例,但可针对其它操作执行在子向量粒度下的类似缩减操作,例如,如与AND、异或(EOR)、浮点加法(FADD)、浮点最大值(FMAX,其确定一组浮点数的最大值)、浮点最小值(FMIN,其确定一组浮点数的最小值)、或(OR)、有正负号最大值(SMAX,其确定一组有正负号整数的最大值)、有正负号最小值(SMIN,其确定一组有正负号整数的最小值)、无正负号最大值(UMAX,其确定一组无正负号整数的最大值)和无正负号最小值(UMIN,其确定一组无正负号整数的最小值)。

[0106] 如图13所示,还可能提供子向量内的缩减指令,其在每个子向量内执行缩减进而将输入操作数Zm的给定子向量的所有元素缩减成结果向量Zd'的单一数据值。该单一元素可经正负号延伸或零延伸,以填充结果向量Zd'的对应子向量。如图13所示,给定子向量内的缩减操作还可取决于从第二向量操作数Zd的对应子向量中的给定元素位置(例如,元素位置0)提取的元素。同样地,预测可基于定义Zm操作数的有效或无效元素的预测值Pg而施加(可始终将取自Zd的额外元素视为是有效的,此元素可用作为追踪一系列先前缩减的结果的累加器值,因此使用如图13所示的破坏性编码可能是有用的,其中将结果写到用以提供该累加器值的相同寄存器Zd)。

[0107] 无论缩减是如图12所示地横跨每个子向量的对应元素来实施,或者缩减是如图13所示地对单一子向量内的元素来实施,此缩减是在子向量粒度下执行的操作,与以下条件所会达成的结果不同:如果对应操作是在全向量粒度下执行(将向量的所有元素缩减成单一数据值),或如果相同的缩减操作是使用包括在回圈或指令序列中的逐元素(element-wise)向量加法指令(或用于其它缩减操作的类似逐位指令)来实施,以将一组向量内的对应位置处的元素相加或以其它方式缩减。基于子向量的方法可用以简化非可缩放向量化码(其可能已假设固定向量长度)到可缩放向量化码的映射,其中子向量对应于该固定向量长度,并且整体向量长度可根据硬件设计者的设计选择而缩放。

[0108] 图14示出可帮助支持向量的向量法(vector of vectors approach)的子向量支持加载/存储指令的示例。图14示出连续加载/存储指令,其中从存储器加载到至少一个向量寄存器或从至少一个向量寄存器存储到存储器的数据块对应于在存储器地址空间中的连续数据块。虽然图14示出加载/存储单一向量寄存器的加载/存储指令,但可提供支持多个向量寄存器的加载/存储的其它加载/存储指令。与其它形式的向量加载/存储指令不同,子向量支持向量加载存储将子向量长度(而不是子向量内的个别向量元素)视为向量加载/存储操作的基本单位。因此,用以指示向量寄存器的哪些部分将会被加载或存储的预测值Pg是在子向量粒度下而不是在个别元素的粒度下施加。对应于零的预测位的子向量可被屏蔽,使得对于加载指令,不将目的地寄存器Zt的对应子向量设置成从存储器加载的对应子向量的值,并且对于存储指令,不将来源寄存器Zt的对应子向量内的数据写入到存储器中的对应寻址位置。

[0109] 类似地,图15示出子向量支持加载/存储指令的集中分散形式,其能够将来自存储器中的非连续地址块的若干个子向量加载或存储到向量寄存器Zt,或从向量寄存器Zt加载或存储该若干个子向量。在此示例中,向量寄存器Zm提供用以确定对应于给定子向量的地址块的索引值。每个索引指示具有对应于子向量长度的尺寸的数据块的倍数。同样地,预测是在子向量粒度下施加。

[0110] 应理解图14和图15所示的寻址模式仅为一个示例,并且其它示例可使用不同技术来确定待加载到向量寄存器或从向量寄存器存储的子向量的地址。例如,对于其它集中分散指令,可将用以计算用于每个子向量的加载/存储操作的地址的向量操作数用作基底地址的向量而不是偏移的向量。一般来说,子向量支持加载/存储指令可是在子向量的粒度下操作的任何加载/存储指令。

[0111] 图16例示了可用以在子向量粒度下设置对应于给定向量的预测的预测设置指令的示例。在此示例中,预测设置指令是在如果'Rn'+i与'Rm'的比较为真的情况下将对应于子向量i的预测位Pi设置成'真'(1)的比较指令,其中'Rn'是存储在第一标量寄存器Rn中的值,并且'Rm'是存储在第二标量寄存器Rm中的值。可提供指令的对应于不同比较条件的不同变体,例如LO(无正负号低于)、LS(无正负号小于或相同)、LT(有正负号小于)、LE(有正负号小于或等于)、HI(无正负号高于)、HS(无正负号高于或相同)、GT(有正负号大于)、或GE(有正负号大于或等于)。此指令类似于在图4的可缩放码示例中示出的whilelt指令,但是在子向量粒度下而不是在向量元素粒度下设置预测。这可用于仿真码的行为,其在非可缩放向量化示例中会屏蔽掉整个向量的作用。

[0112] 图17示出子向量支持预测设置指令的第二示例,其中同样地在子向量粒度下设置预测值,但在此示例中,指令指定将要在子向量粒度下施加的特定预定义模式(例如,待取决于要实施的模式而设置/清除的预测位是对应于每个子向量的开始的所述预测位)。这可能与可在个别向量元素的粒度下施加模式的对应预测设置指令不同。例如,预测模式可为将指定数目的预测位设置成有效,在每N个预测中设置一个有效者(其中N是例如2、3、4等的值),或在考虑硬件的所实施向量长度时将对应于与预测值适配的2的最大幂数的若干个预测位设置成有效。

[0113] 图18例示了指定识别预定预测模式的模式识别符且在子向量粒度下施加时控制处理电路16以对应于由该经指定预测模式所指示的有效子向量数目的数目来递增标量操

作数Xdn的子向量支持递增指令的示例。可提供对应指令以代替地通过预测模式所指示的有效子向量的数目来递减操作数。可采用对应方式将预测模式定义成针对图17的指令而定义的模式。此指令可用于控制回圈计数变量的递增或递减,所述回圈计数变量用以追踪是否仍需要继续进一步回圈迭代,或者由于所需的子向量都已受处理而是否可能终止回圈。

[0114] 上文描述的所有指令可有助于使将针对固定长度向量架构所编写的码调适成可缩放向量架构变得更容易。应理解并非所有这些指令均需要在给定具体实施中实施。此外,类似的子向量粒度指令可针对其它操作来定义。

[0115] 总地来说,为使从非可缩放向量长度规定架构(例如Neon™)转变成可缩放向量长度未知架构(例如SVE)的软件开发者可以更直接地转变,上述示例加入子向量(例如,四字组(128位)尺寸的)元素,并将每个子向量视为可缩放架构中的元素。在如此进行的同时,形成向量中的向量(vector-in-vector)风格,以使用可缩放架构的向量长度未知风格将非可缩放架构的每个固定长度向量向量化。这允许以粗略的1对1指令映射将非可缩放码映射成可缩放码,使得向量化可利用可缩放架构中所允许的较长且具弹性的向量长度,并且仍保留针对非可缩放架构引入的依赖向量长度在编译时间下已知的假设的码优化。

[0116] 图19例示了可使用的模拟器具体实施。虽然稍早所述的实施方案以用于操作支持所关注技术的特定处理硬件的设备和方法来实施本发明,但也可能根据本文所述的实施方案提供指令执行环境,其是通过使用计算机程序实施。这类计算机程序常称为模拟器,因为其提供硬件架构的基于软件的具体实施。模拟器计算机程序的种类包括仿真器、虚拟机、模型和二进制转换器(包括动态二进制转换器)。一般来说,模拟器具体实施可在可选地运行主机操作系统320、支持模拟器程序310的主机处理器330上运行。在一些布置中,在硬件与所提供的指令执行环境和/或相同的主机处理器上提供的多个相异指令执行环境之间可有多层模拟。历史上,已需要强大的处理器来提供模拟器具体实施,其以合理速度执行,但这种方法在某些情况下可能是有正当理由的,例如当因为兼容性或再使用原因而需要运行另一处理器本体的码时。例如,模拟器具体实施可提供具有不为主机处理器硬件所支持的额外功能性的指令执行环境,或提供一般与不同的硬件架构相关联的指令执行环境。模拟的综述是在“Some Efficient Architecture Simulation Techniques”中给出,Robert Bedichek,Winter 1990USENIX Conference,第53至63页。

[0117] 在先前已参照特定硬件架构或特征来描述实施方案的情况下,在模拟实施方案中,可通过合适的软件架构或特征提供等效功能。例如,可在模拟实施方案中将特定电路实施为计算机程序逻辑。类似地,存储器硬件(例如,寄存器或高速缓存)可在模拟实施方案中经实施作为存储在主机处理器330的主机存储器(例如,存储器或寄存器)中的软件数据结构。在先前描述的实施方案中提及的硬件元件中的一个或多个硬件元件存在于主机硬件(例如,主机处理器330)上的布置中,一些模拟实施方案可(在适当处)利用主机硬件。

[0118] 模拟器程序310可存储在计算机可读存储介质(其可为非暂态介质)上,并提供程序接口(指令执行环境)给目标码300(其可包括应用程序、操作系统和超管理器),该程序接口与通过模拟器程序310模型化的硬件架构的程序接口相同。因此,目标码300的程序指令可使用模拟器程序310而从指令执行环境内执行,使得实际上不具有上文讨论的设备2的硬件特征(例如,支持如上文讨论的子向量支持指令的指令解码器10和处理电路16)的主机计算机330可仿真所述特征。

[0119] 因此,模拟器程序310可具有用于解码目标码300的指令和将这些指令映射到主机设备330的原生指令集中的对应指令集的指令解码程序逻辑312。指令解码程序逻辑312包括用于解码上文描述的子向量支持指令的子向量支持指令解码程序逻辑313。寄存器仿真程序逻辑314将由目标码请求的寄存器存取映射成对维护在主机设备330的主机硬件上的对应数据结构的存取,例如通过存取主机设备330的寄存器或存储器中的数据。存储器管理程序逻辑316采用对应于如上述硬件实施实施方案所描述的MMU 36的方式来实施地址转换、页表遍历和存取权限检查,以模拟目标码300对模拟地址空间的存取。提供存储器地址空间模拟程序逻辑318,以将基于使用由目标程序码300的软件维护的页表信息的地址转换由存储器管理程序逻辑316获得的模拟实体地址映射成用以存取主机处理器330的主机存储器的主机虚拟地址。这些主机虚拟地址可本身使用由主机所支持的标准地址转换机制来转换成主机实体地址(将主机虚拟地址转换成主机实体地址是在受到模拟器程序310控制的范围之外)。

[0120] 在本申请中,用语“被配置为……”是用以意指设备的元件具有能够实行该经定义操作的配置。在此上下文中,“配置”意指硬件或软件的互连的布置或方式。例如,该设备可具有专用硬件,其提供经定义的操作,或者处理器或其它处理装置可经程序化以执行所述功能。“被配置为”并不意味着设备元件需要以任何方式改变以提供所定义的操作。

[0121] 虽然本文已参照附图详细地描述本发明的说明性实施方案,但应了解本发明不限于该精确实施方案,并且所属技术领域的技术人员可于其中实行各种变化与修改,而不脱离如随附权利要求书所定义的本发明的范围。

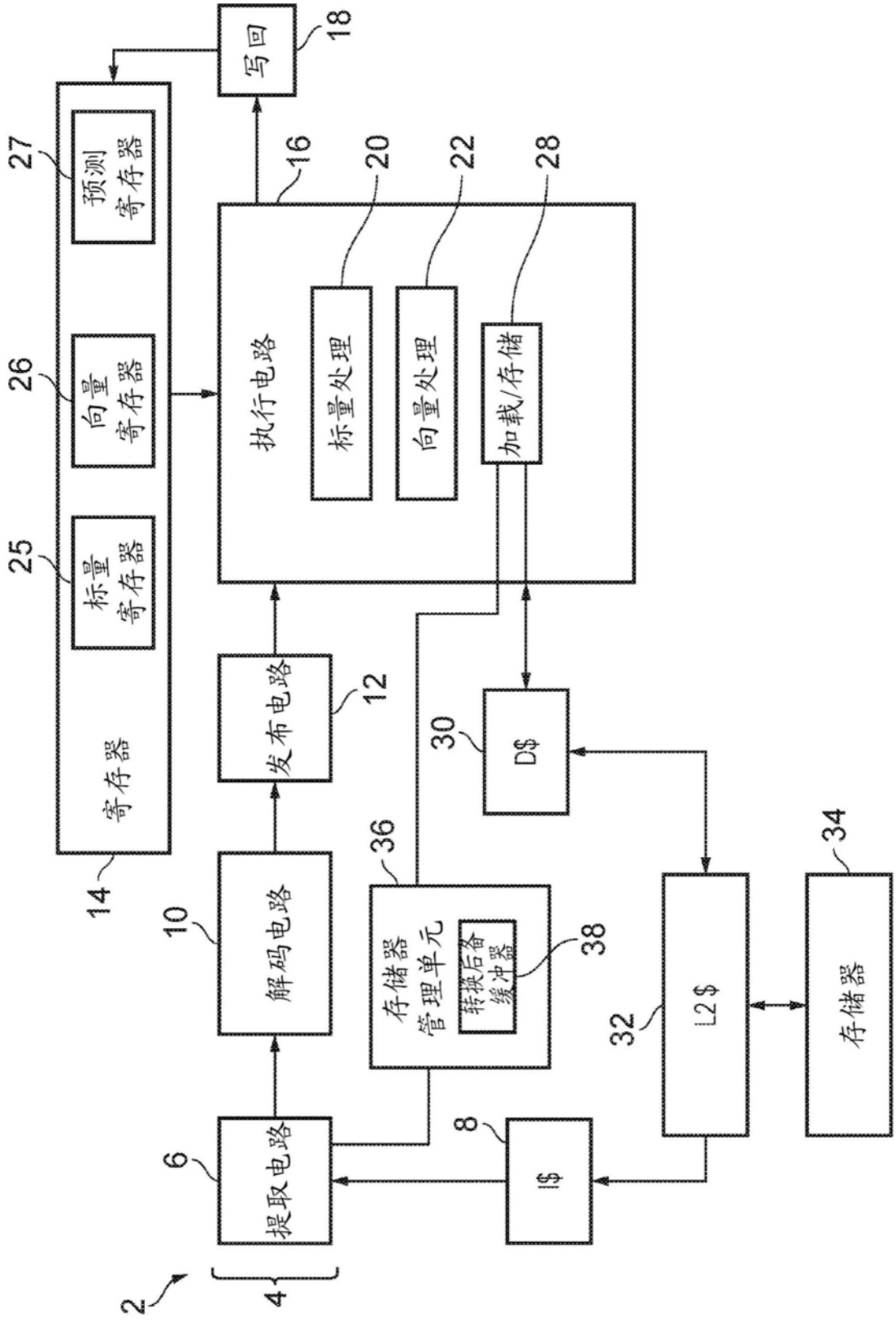

写回
18
预测寄存器
27
向量寄存器
26
标量寄存器
25
寄存器
14
执行电路
16
标量处理
20
向量处理
22
加载/存储
28
发布电路
12
D$
30
解码电路
10
存储器管理单元
36
转换后备缓冲器
38
L2 $
32
存储器
34
提取电路
6
I$
8
2
4

图1

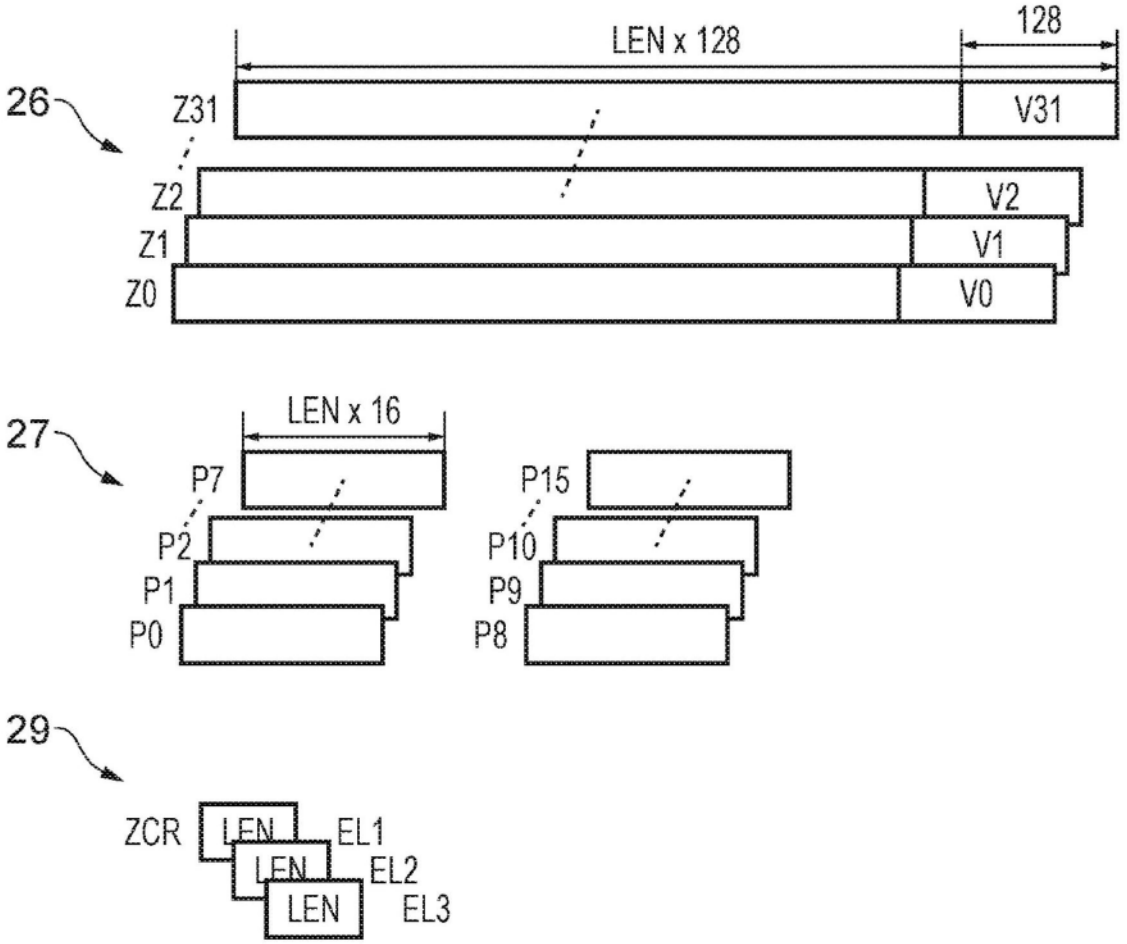

LEN x 128
128
26
Z31
V31
Z2
V2
Z1
V1
Z0
V0
LEN x 16
27
P7
P15
P2
P10
P1
P9
P0
P8
29
ZCR
LEN
EL1
LEN
EL2
LEN
EL3

图2

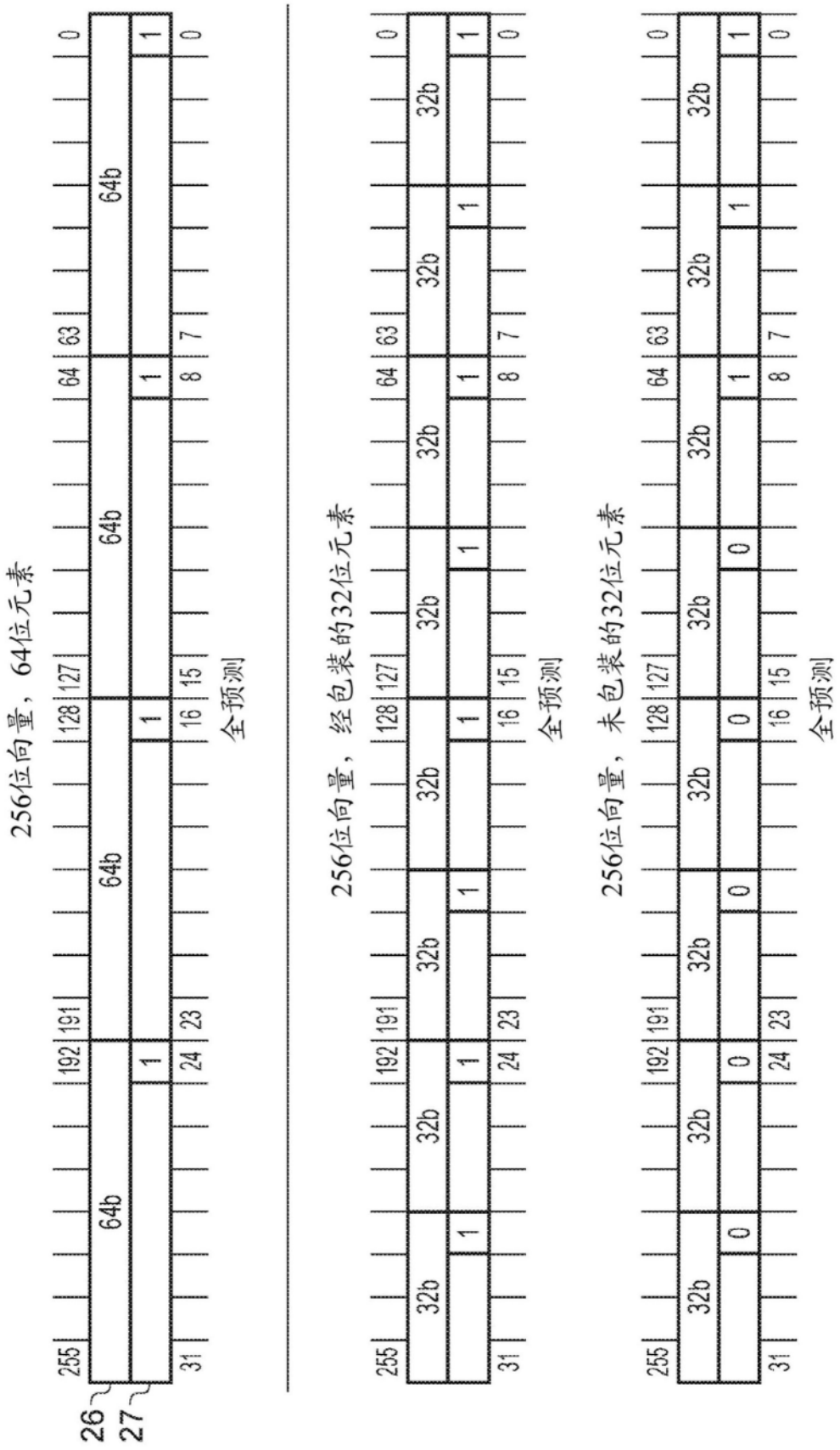

256位向量，64位元素
64b
1
0
全预测
256位向量，经包装的32位元素
32b
256位向量，未包装的32位元素
26
27

图3

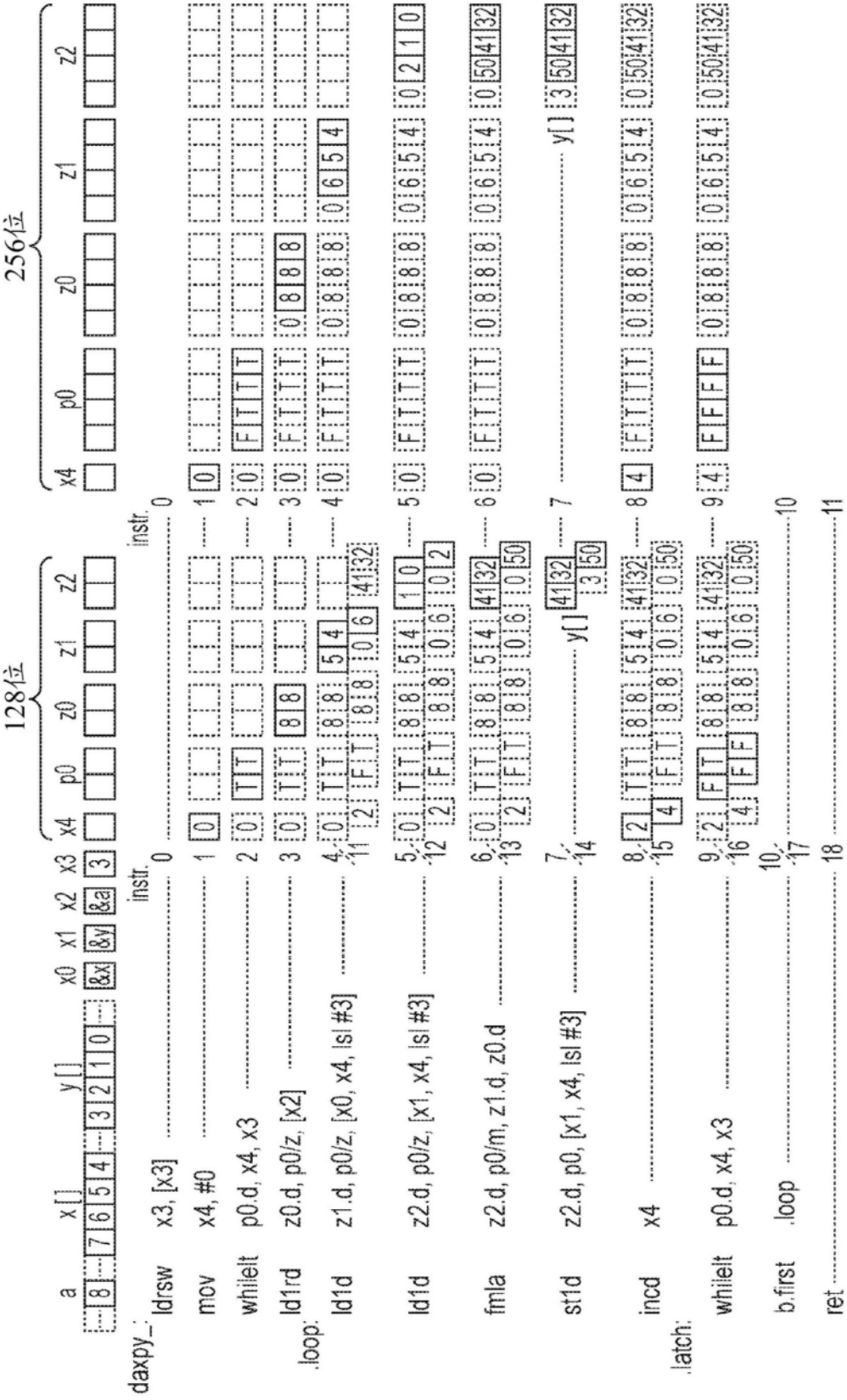

256位
128位
daxpy_:
.loop:
.latch:
ldrsw x3, [x3]
mov x4, #0
whilelt p0.d, x4, x3
ld1rd z0.d, p0/z, [x2]
ld1d z1.d, p0/z, [x0, x4, lsl #3]
ld1d z2.d, p0/z, [x1, x4, lsl #3]
fmla z2.d, p0/m, z1.d, z0.d
st1d z2.d, p0, [x1, x4, lsl #3]
incd x4
whilelt p0.d, x4, x3
b.first .loop
ret

图4

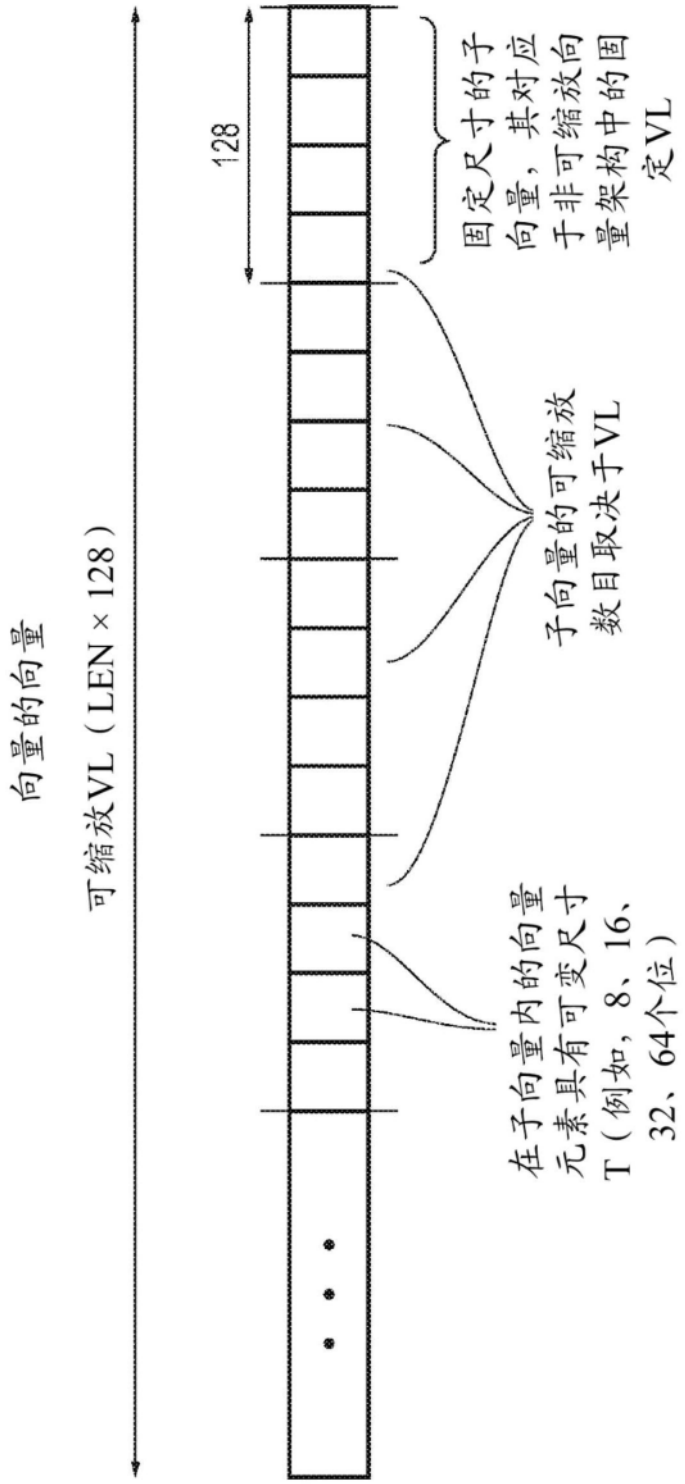

向量的向量
可缩放VL（LEN × 128）
128
固定尺寸的子向量，其对应于非可缩放向量架构中的固定VL
子向量的可缩放数目取决于VL
在子向量内的向量元素具有可变尺寸T（例如，8、16、32、64个位）

图5

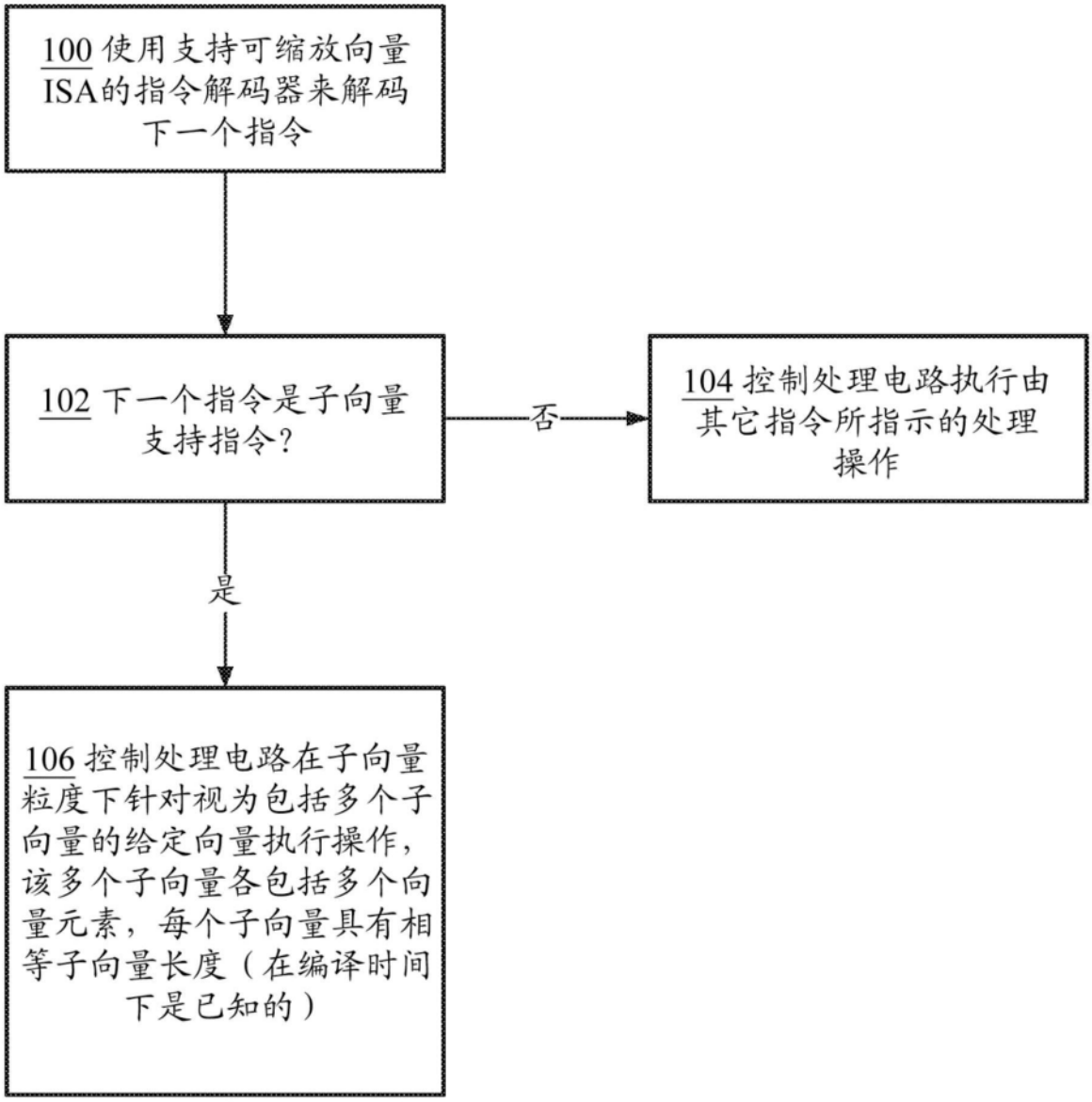

100 使用支持可缩放向量ISA的指令解码器来解码下一个指令
102 下一个指令是子向量支持指令?
否
104 控制处理电路执行由其它指令所指示的处理操作
是
106 控制处理电路在子向量粒度下针对视为包括多个子向量的给定向量执行操作，该多个子向量各包括多个向量元素，每个子向量具有相等子向量长度（在编译时间下是已知的）

图6

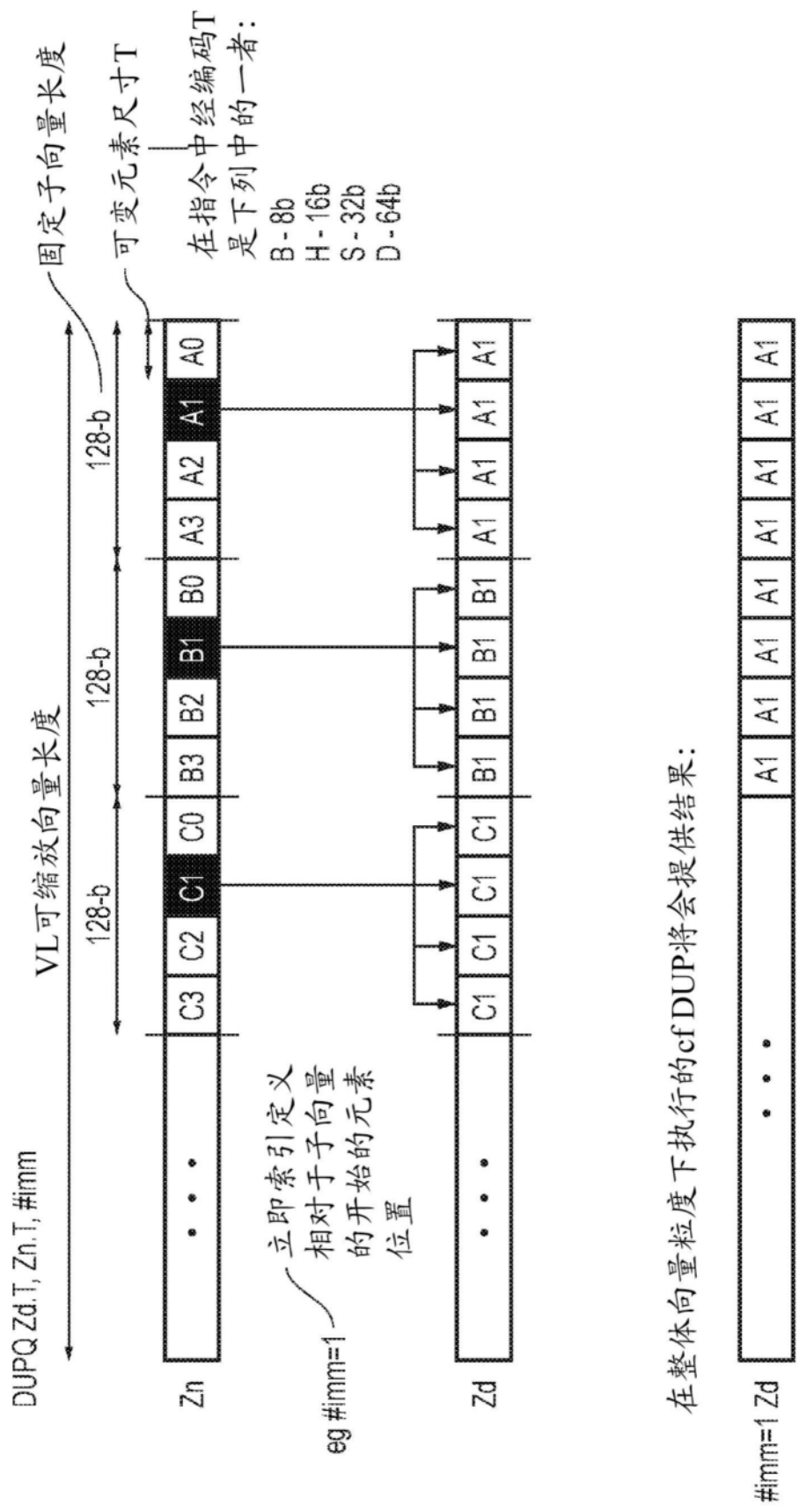

DUPQ Zd.T, Zn.T, #imm
VL可缩放向量长度
固定子向量长度
可变元素尺寸T
在指令中经编码T是下列中的一者：
B - 8b
H - 16b
S - 32b
D - 64b
128-b
128-b
128-b
Zn
C3 C2 C1 C0 B3 B2 B1 B0 A3 A2 A1 A0
eg #imm=1
立即索引定义相对于子向量的开始的元素位置
Zd
C1 C1 C1 C1 B1 B1 B1 B1 A1 A1 A1 A1
在整体向量粒度下执行的cf DUP将会提供结果：
#imm=1 Zd
A1 A1 A1 A1 A1 A1 A1 A1

图7

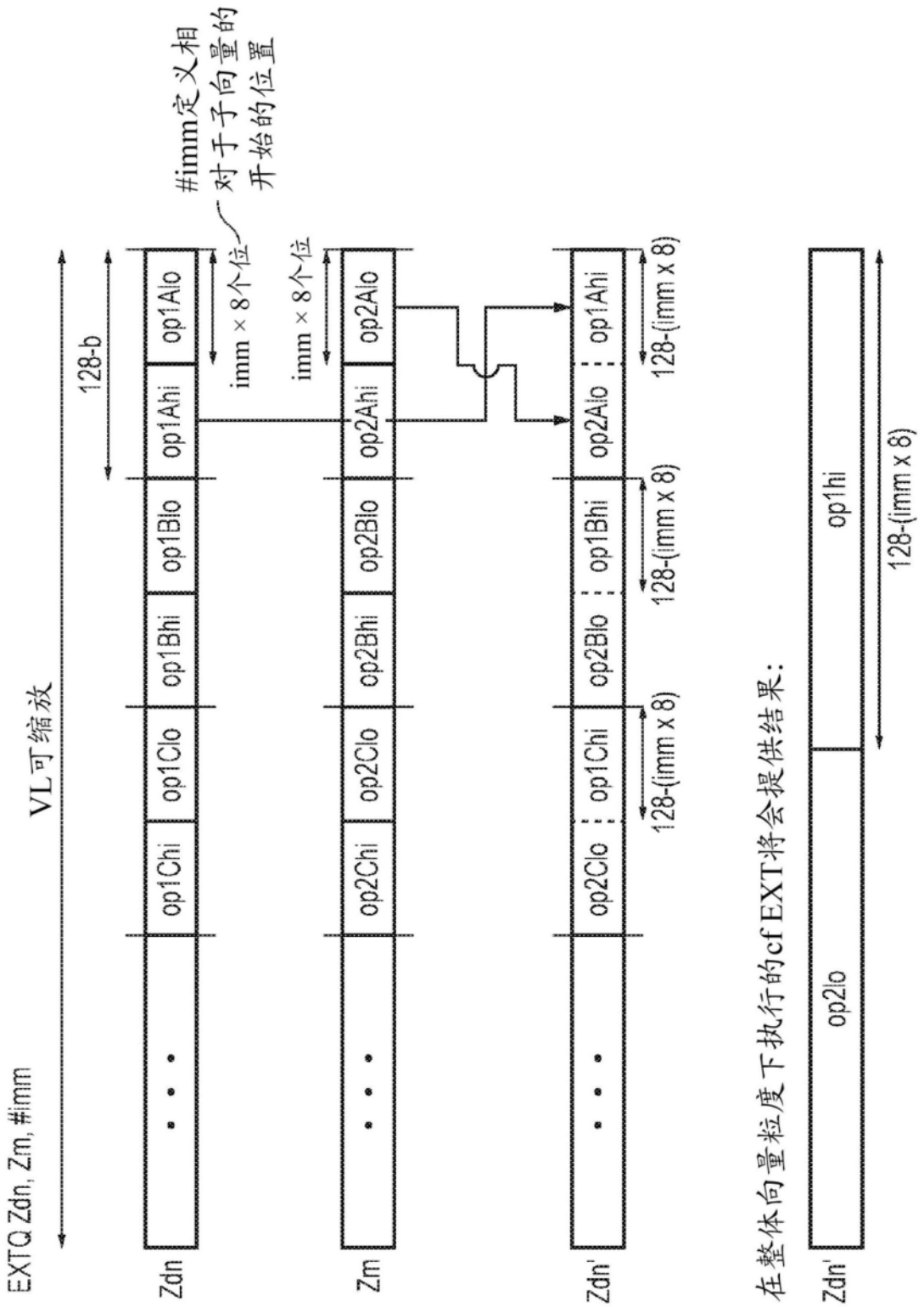

EXTQ Zdn, Zm, #imm
VL可缩放
128-b
#imm定义相对于子向量的开始的位置
imm × 8个位
imm × 8个位
Zdn
Zm
Zdn'
op1Chi
op1Clo
op1Bhi
op1Blo
op1Ahi
op1Alo
op2Chi
op2Clo
op2Bhi
op2Blo
op2Ahi
op2Alo
op2Clo
op1Chi
op2Blo
op1Bhi
op2Alo
op1Ahi
128-(imm x 8)
128-(imm x 8)
128-(imm x 8)
在整体向量粒度下执行的cf EXT将会提供结果:
Zdn'
op2lo
op1hi
128-(imm x 8)

图8

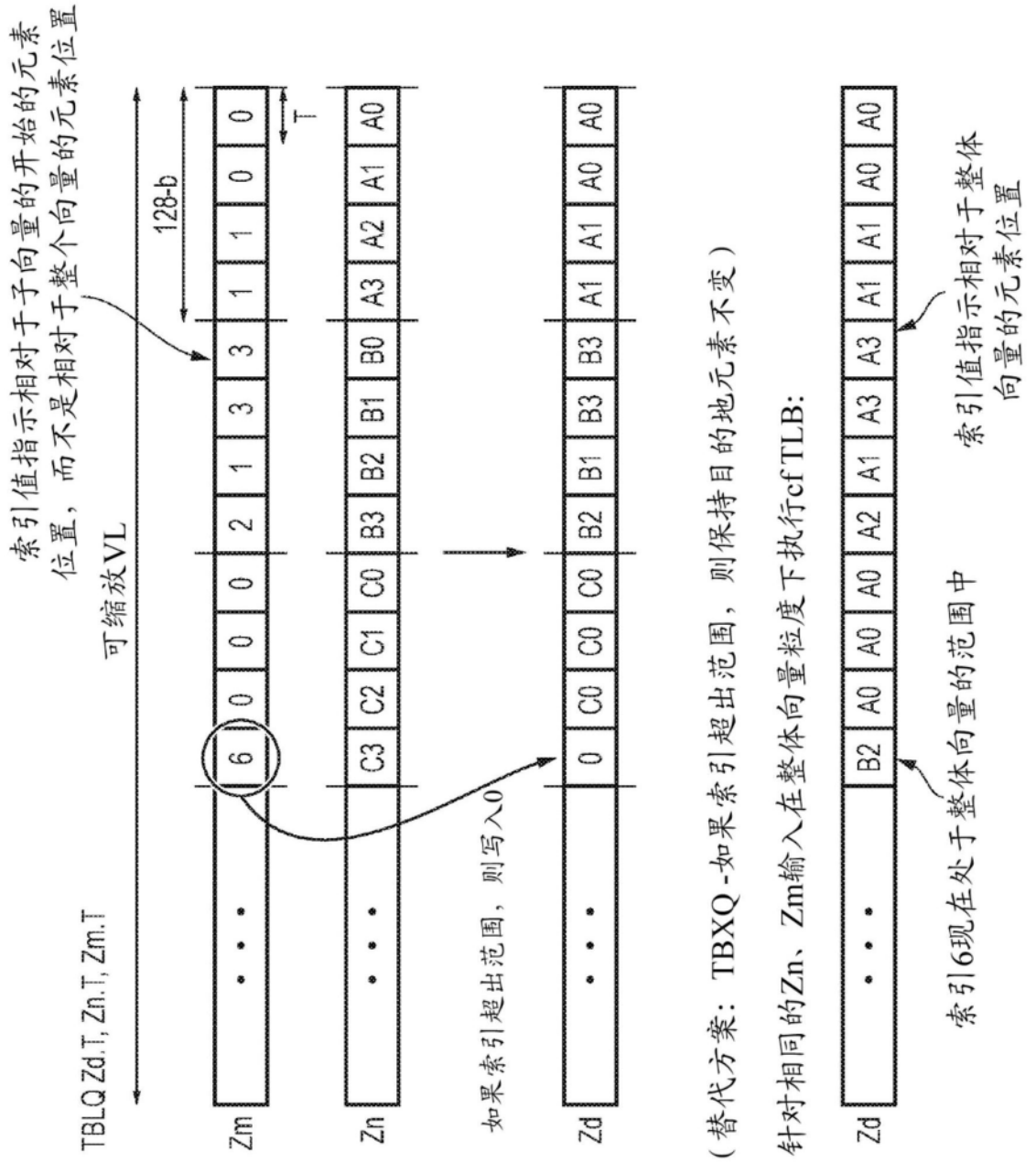

索引值指示相对于子向量的开始的元素位置，而不是相对于整个向量的元素位置
可缩放VL
TBLQ Zd.T, Zn.T, Zm.T
128-b
T
Zm
Zn
Zd
如果索引超出范围，则写入0
（替代方案：TBXQ-如果索引超出范围，则保持目的地元素不变）
针对相同的Zn、Zm输入在整体向量粒度下执行cf TLB:
索引6现在处于整体向量的范围中
索引值指示相对于整体向量的元素位置

图9

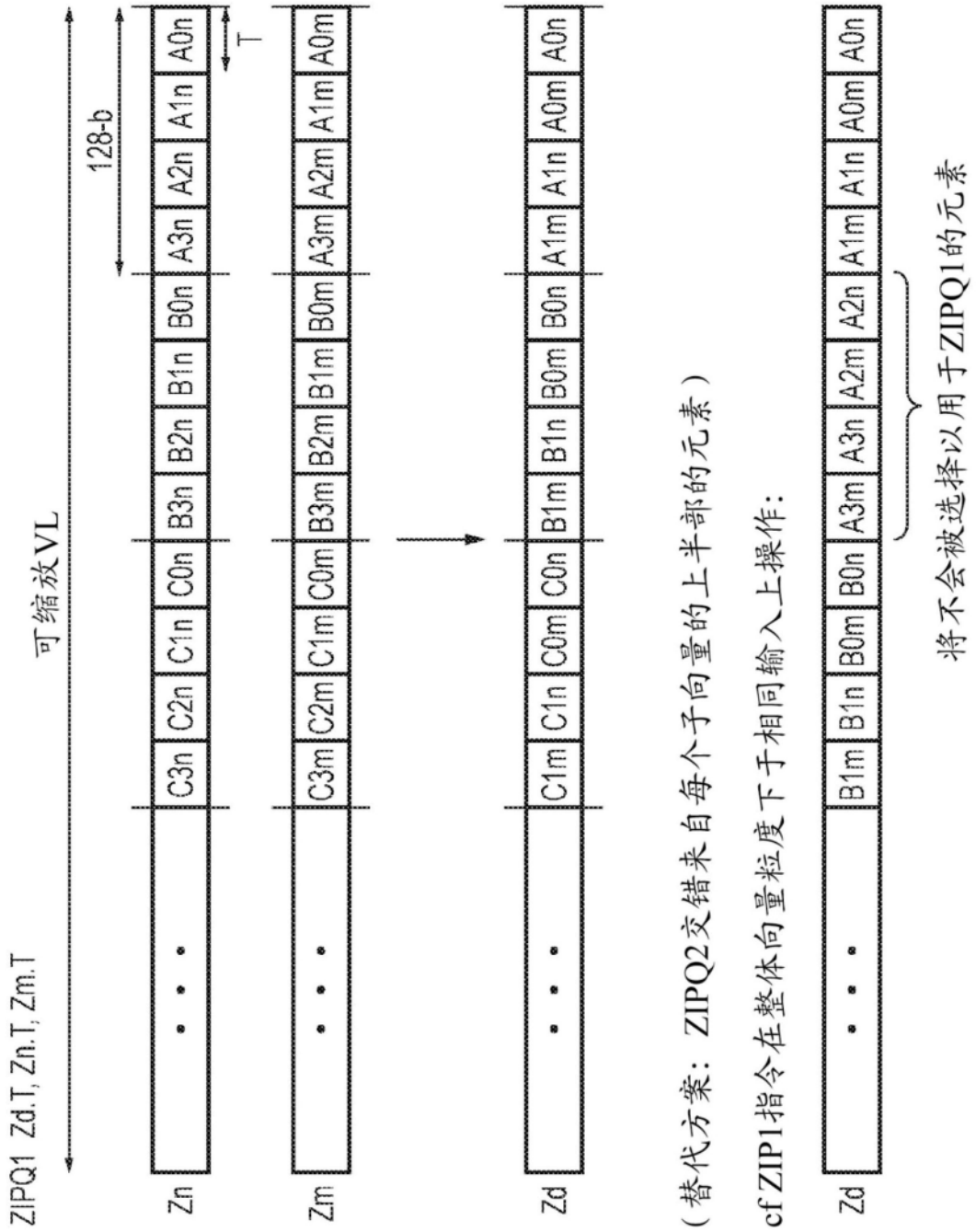

ZIPQ1 Zd.T, Zn.T, Zm.T
可缩放VL
128-b
T
Zn
Zm
Zd
C3n C2n C1n C0n B3n B2n B1n B0n A3n A2n A1n A0n
C3m C2m C1m C0m B3m B2m B1m B0m A3m A2m A1m A0m
C1m C1n C0m C0n B1m B1n B0m B0n A1m A1n A0m A0n
（替代方案：ZIPQ2交错来自每个子向量的上半部的元素）
cf ZIP1指令在整体向量粒度下于相同输入上操作：
Zd
B1m B1n B0m B0n A3m A3n A2m A2n A1m A1n A0m A0n
将不会被选择以用于ZIPQ1的元素

图10

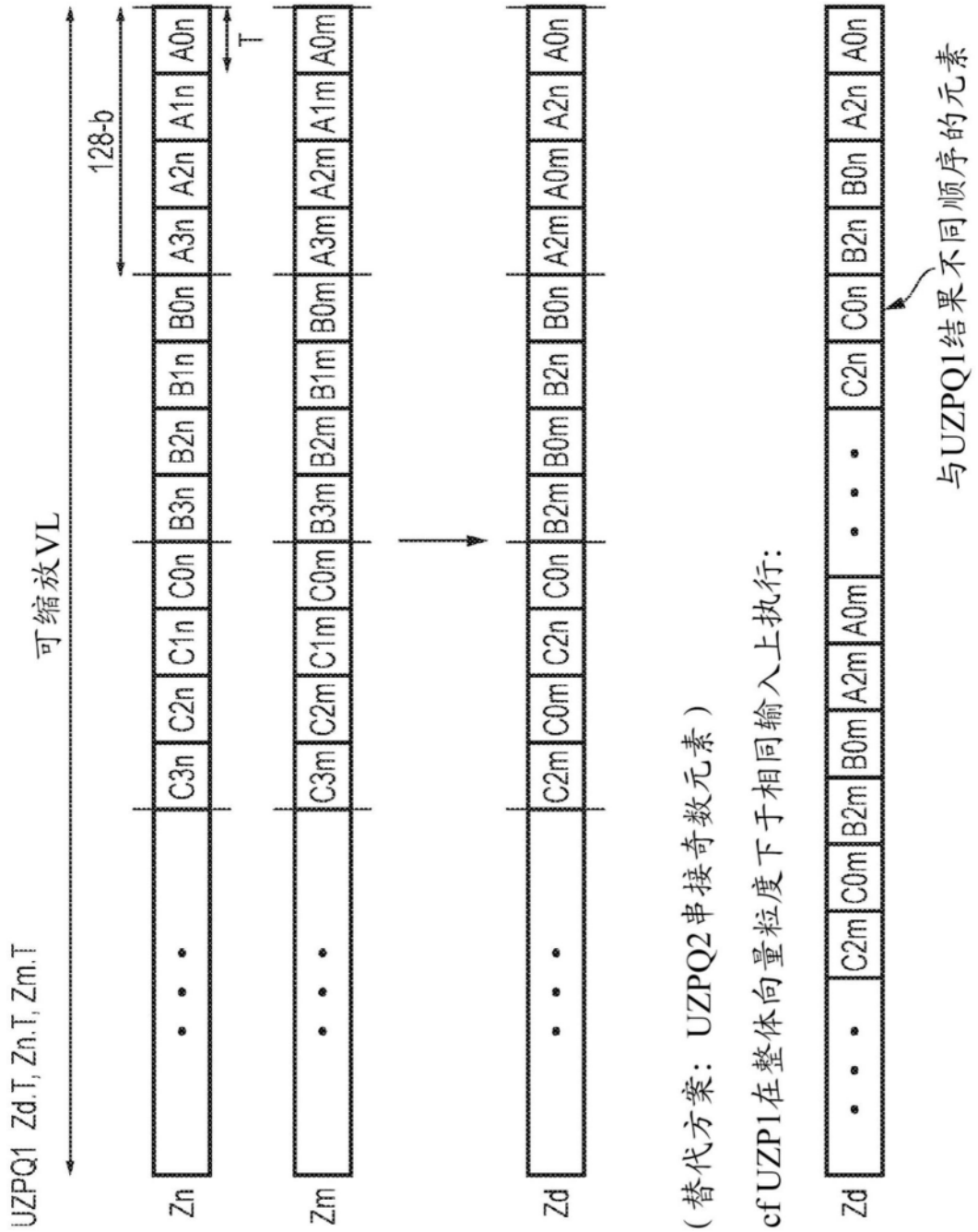

UZPQ1 Zd.T, Zn.T, Zm.T
可缩放VL
128-b
T
Zn
Zm
Zd
C3n C2n C1n C0n B3n B2n B1n B0n A3n A2n A1n A0n
C3m C2m C1m C0m B3m B2m B1m B0m A3m A2m A1m A0m
C2m C0m C2n C0n B2m B0m B2n B0n A2m A0m A2n A0n
（替代方案：UZPQ2串接奇数元素）
cf UZP1在整体向量粒度下于相同输入上执行:
C2m C0m B2m B0m A2m A0m C2n C0n B2n B0n A2n A0n
与UZPQ1结果不同顺序的元素

图11

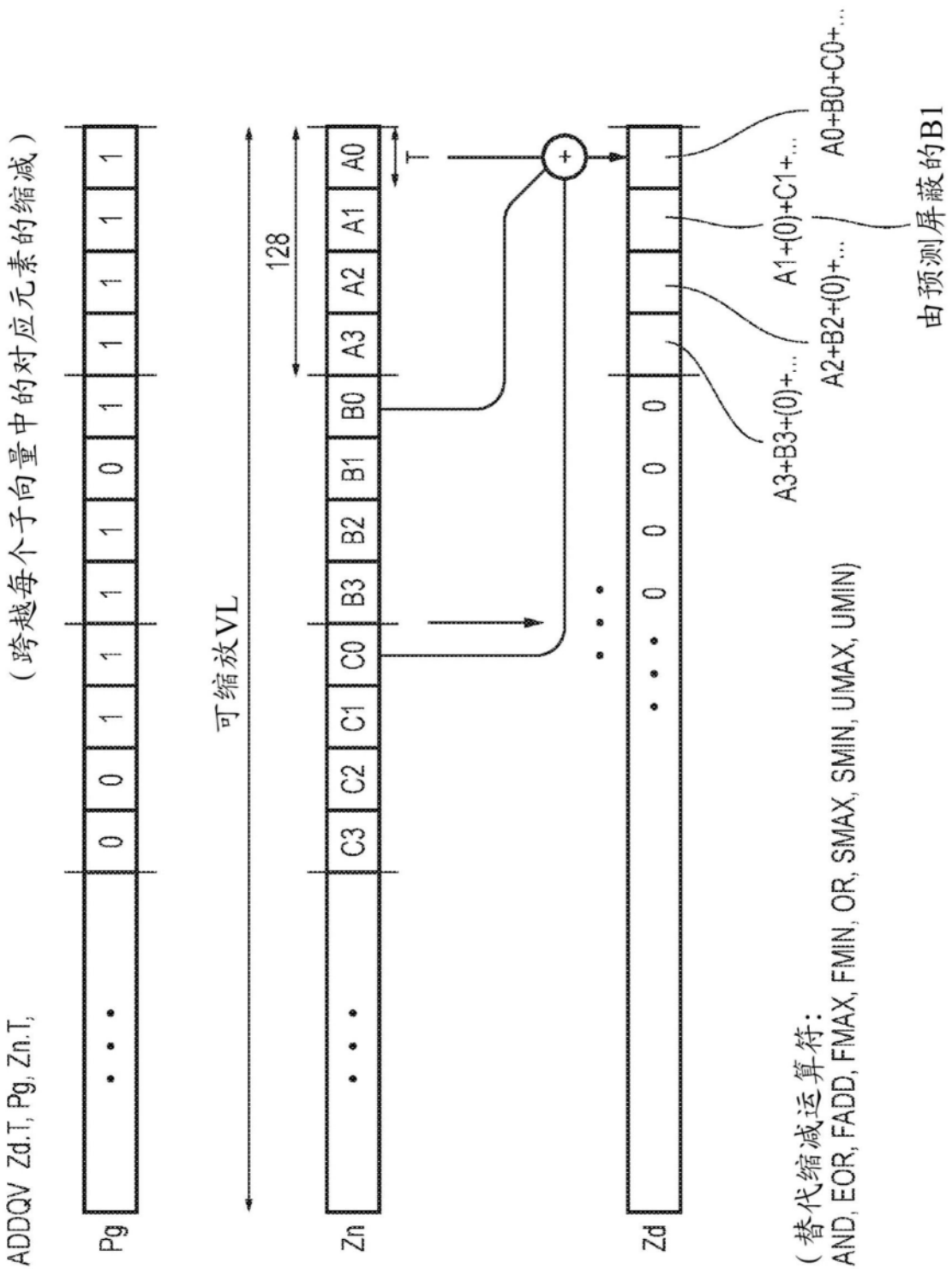

ADDQV Zd.T, Pg, Zn.T,
（跨越每个子向量中的对应元素的缩减）
Pg
Zn
Zd
可缩放VL
128
T
C3 C2 C1 C0 B3 B2 B1 B0 A3 A2 A1 A0
A3+B3+(0)+...
A2+B2+(0)+...
A1+(0)+C1+...
A0+B0+C0+...
由预测屏蔽的B1
（替代缩减运算符：
AND, EOR, FADD, FMAX, FMIN, OR, SMAX, SMIN, UMAX, UMIN）

图12

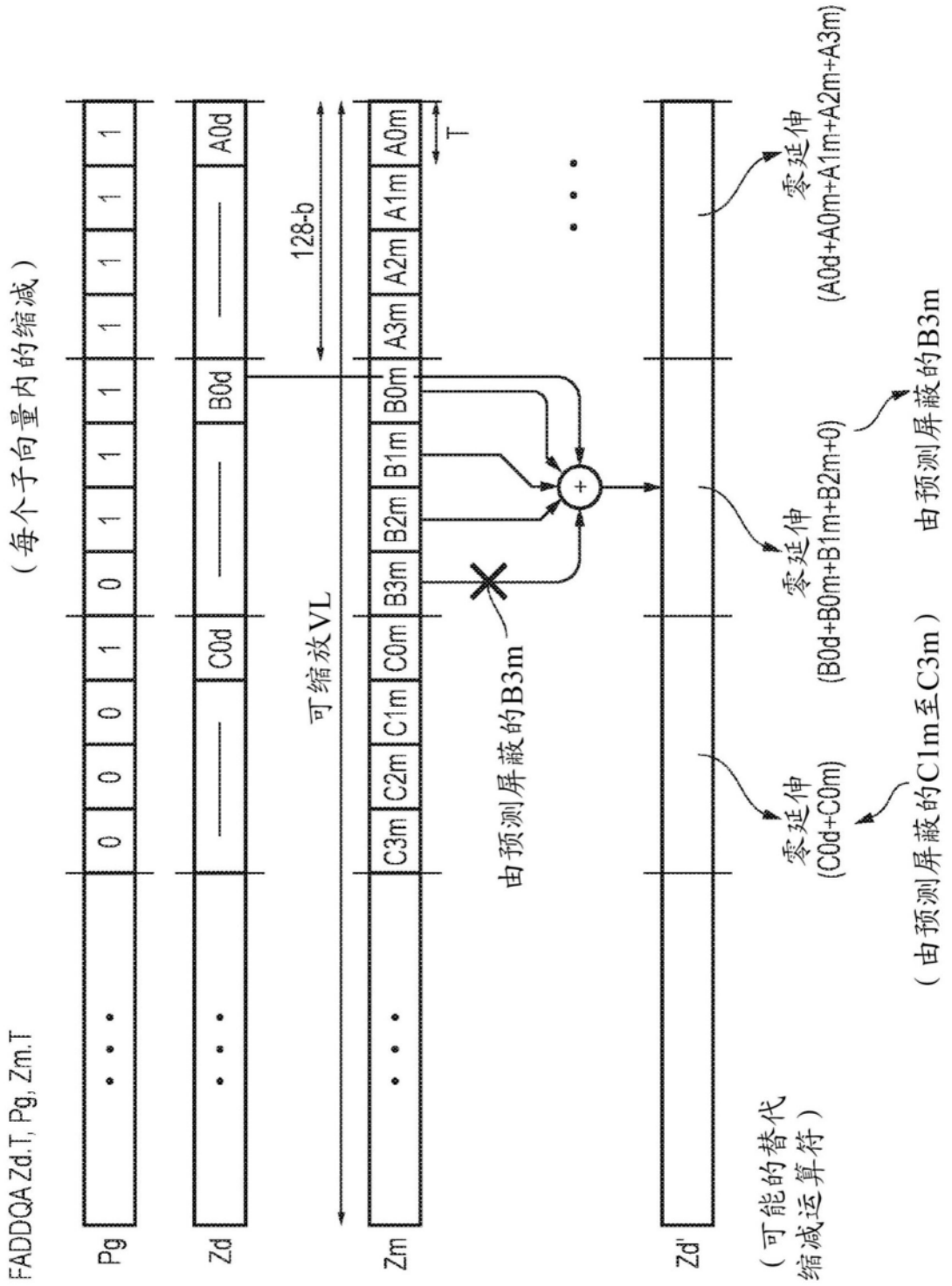

FADDQA Zd.T, Pg, Zm.T
（每个子向量内的缩减）
Pg
Zd
Zm
Zd'
128-b
可缩放VL
T
A0d
B0d
C0d
A0m
A1m
A2m
A3m
B0m
B1m
B2m
B3m
C0m
C1m
C2m
C3m
由预测屏蔽的B3m
零延伸
(A0d+A0m+A1m+A2m+A3m)
零延伸
(B0d+B0m+B1m+B2m+0)
由预测屏蔽的B3m
零延伸
(C0d+C0m)
（由预测屏蔽的C1m至C3m）
（可能的替代缩减运算符）

图13

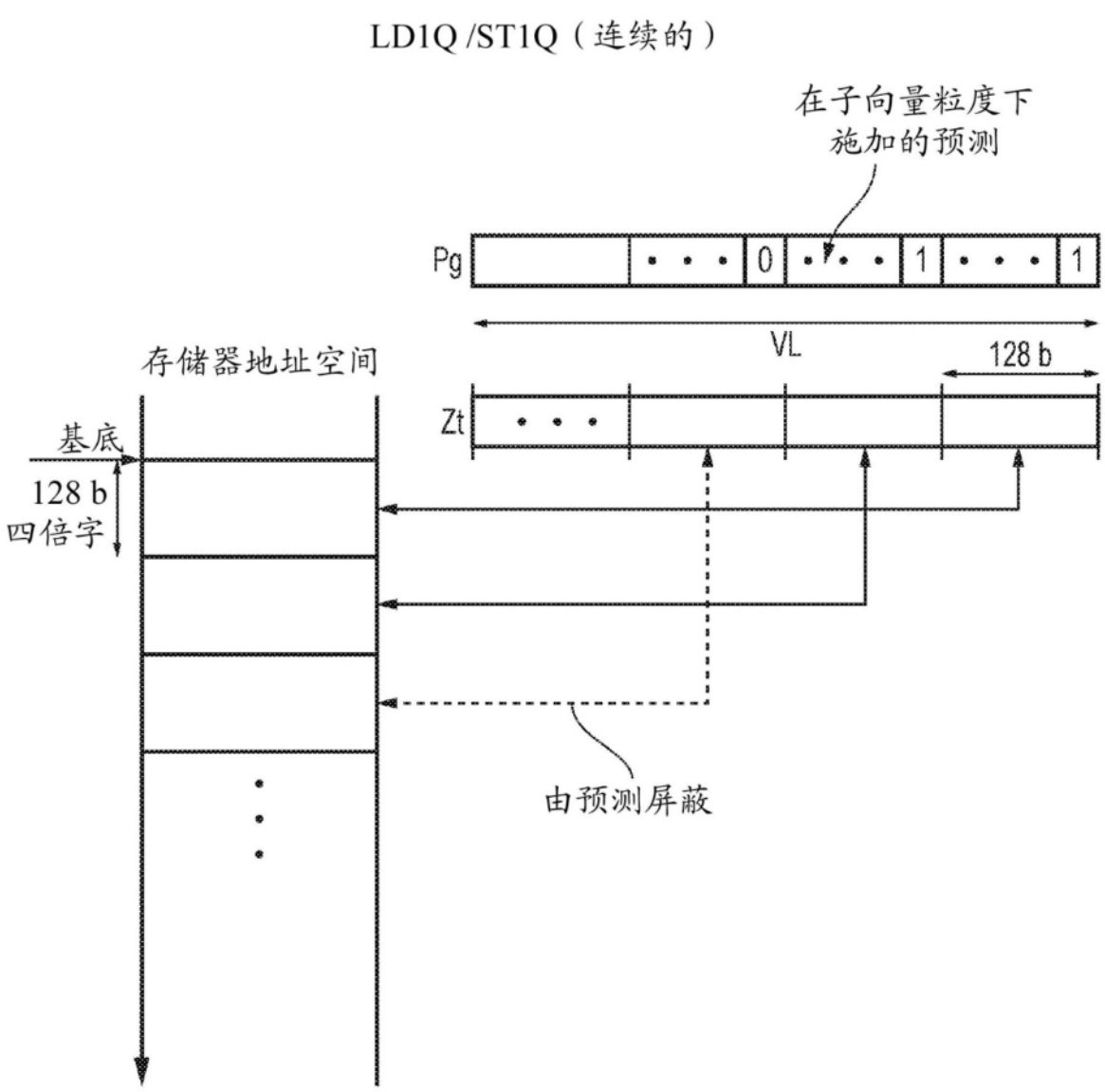

LD1Q /ST1Q（连续的）
在子向量粒度下
施加的预测
Pg
0
1
1
VL
128 b
存储器地址空间
Zt
基底
128 b
四倍字
由预测屏蔽

图14

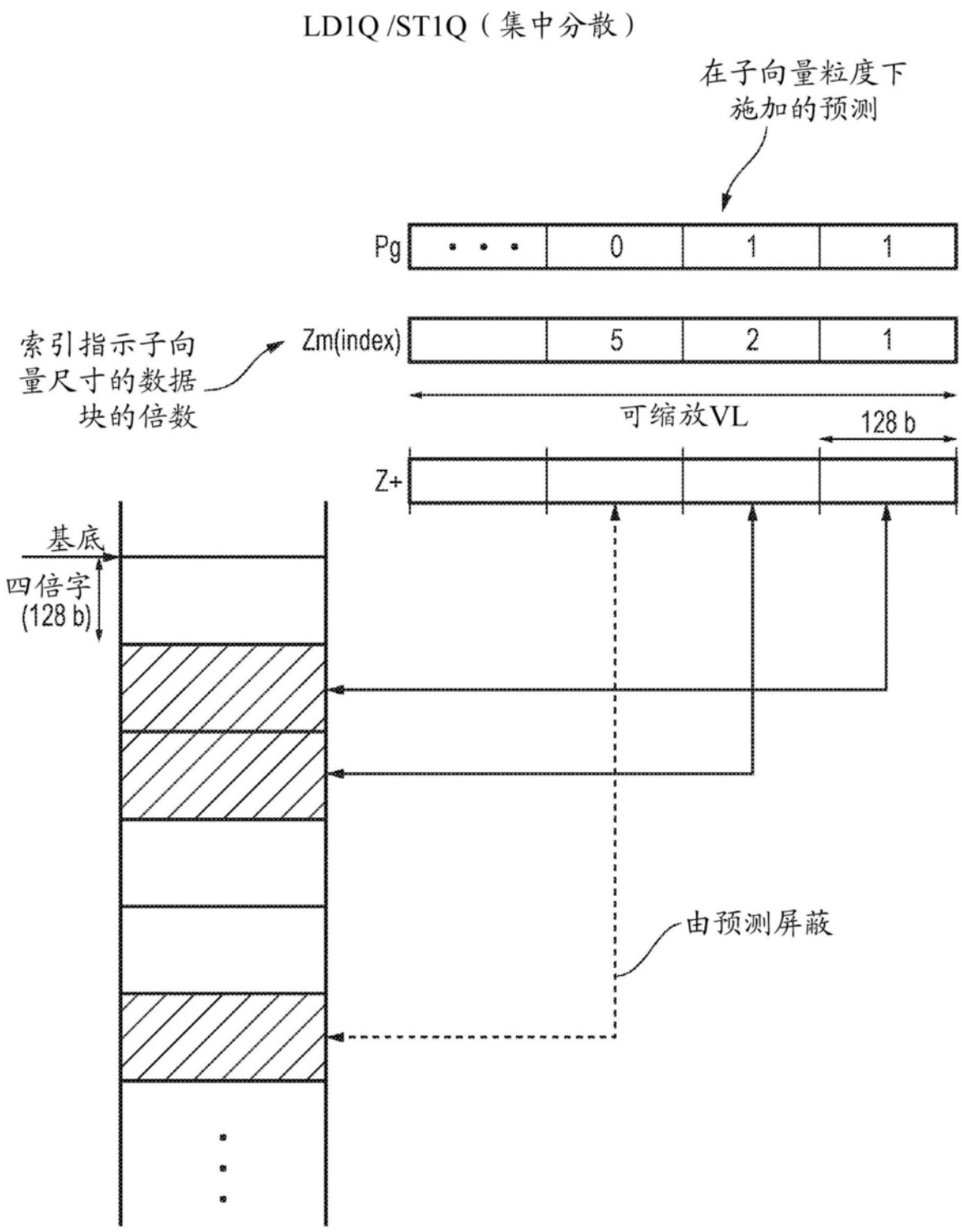
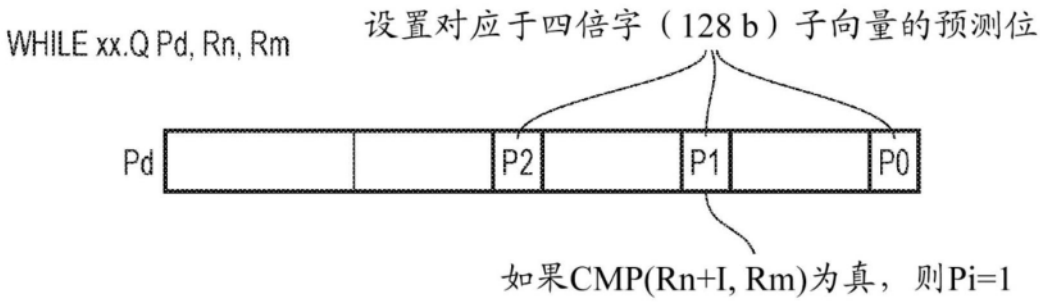

LD1Q /ST1Q（集中分散）
在子向量粒度下施加的预测
Pg
0
1
1
索引指示子向量尺寸的数据块的倍数
Zm(index)
5
2
1
可缩放VL
128 b
Z+
基底
四倍字(128 b)
由预测屏蔽

图15

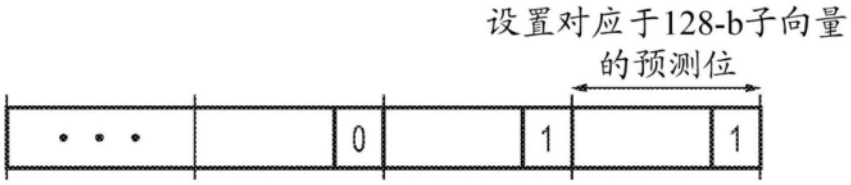


xx指示比较条件，例如：

-LO	-HI
-LS	-HS
-LT	-GT
-LE	-GE

图16

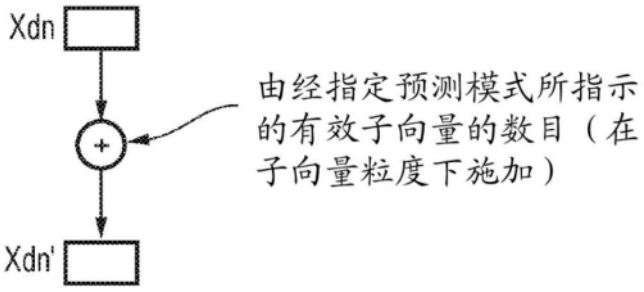
PTRUES Pd.Q，模式



模式定义要在子向量粒度下施加的预测模式

图17

INCQ Xdn，模式



（替代方案：DECQ递减操作数）

图18

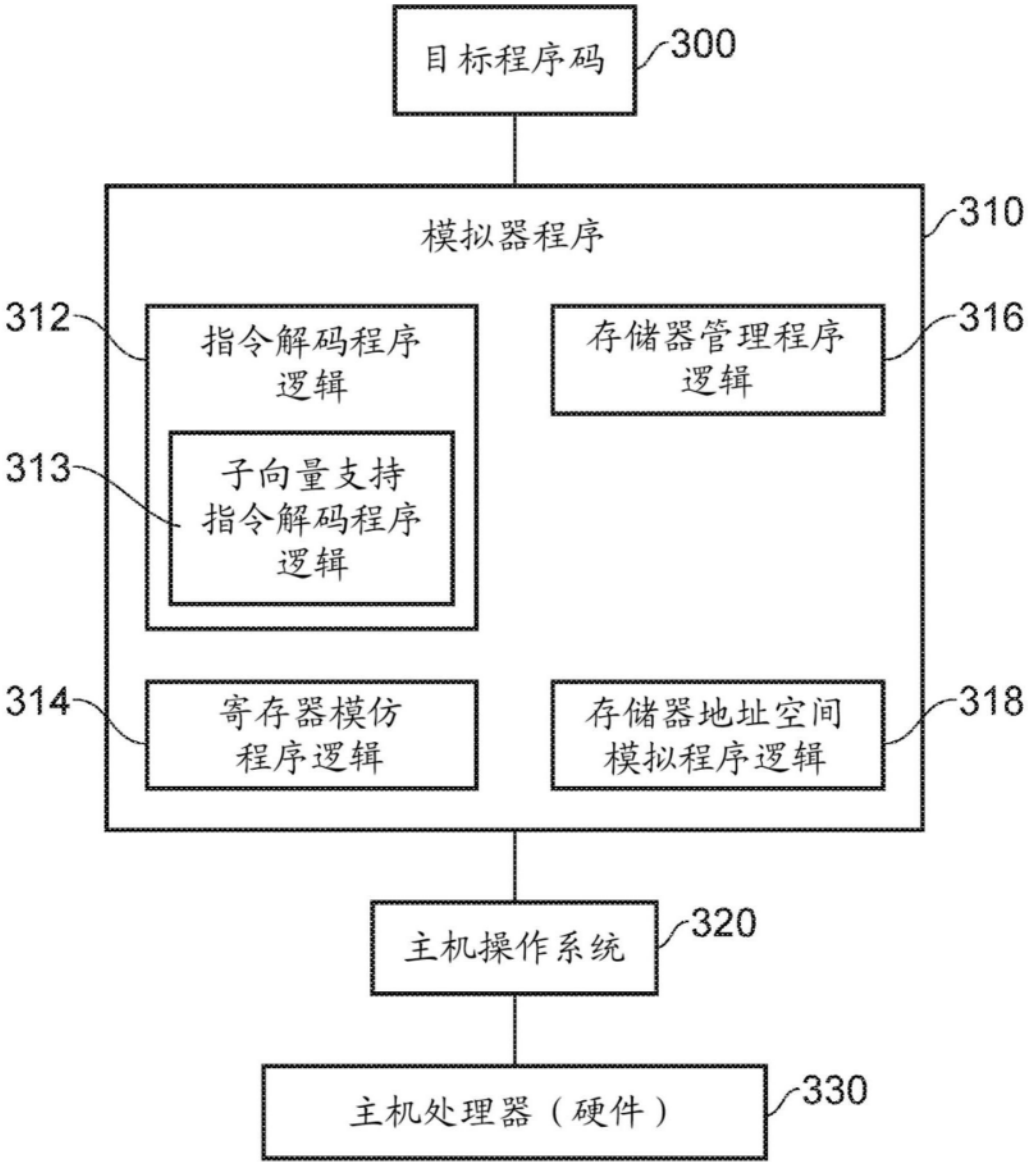

目标程序码
300
模拟器程序
310
312
指令解码程序
逻辑
313
子向量支持
指令解码程序
逻辑
316
存储器管理程序
逻辑
314
寄存器模仿
程序逻辑
318
存储器地址空间
模拟程序逻辑
主机操作系统
320
主机处理器（硬件）
330

图19