(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 104011663 B
(45)授权公告日 2018.01.26

(21)申请号 201180075791.9

(22)申请日 2011.12.22

(65)同一申请的已公布的文献号
申请公布号 CN 104011663 A

(43)申请公布日 2014.08.27

(85)PCT国际申请进入国家阶段日
2014.06.20

(86)PCT国际申请的申请数据
PCT/US2011/067035 2011.12.22

(87)PCT国际申请的公布数据
WO2013/095575 EN 2013.06.27

(73)专利权人 英特尔公司
地址 美国加利福尼亚州

(72)发明人 E·乌尔德-阿迈德-瓦尔
M·B·吉尔卡尔 R·C·凡伦天
S·赛尔 J·考博尔圣阿德里安

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

(51)Int.Cl.
G06F 9/30(2006.01)
G06F 9/305(2006.01)

(56)对比文件
US 2004/0030863 A1,2004.02.12,
CN 1072788 A,1993.06.02,
US 2002/0112147 A1,2002.08.15,
US 2003/0093648 A1,2003.05.15,

审查员 舒瀚

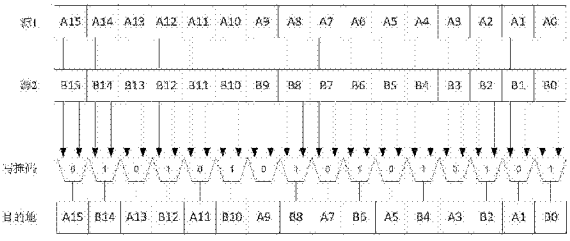
权利要求书2页 说明书16页 附图19页

(54)发明名称

掩码寄存器上的广播操作

(57)摘要

描述了在计算机处理器中执行掩码广播指令的系统、设备和方法的实施例。在一些实施例中，掩码广播指令的执行导致根据广播大小源操作数的数据元素广播到目的地操作数的目的地寄存器。



CN 104011663 B

1.一种用于在计算机处理器中执行掩码广播指令的方法,包括:

获取所述掩码广播指令,其中所述掩码广播指令包括目的地操作数、源操作数、第二源操作数和广播大小;

解码所获取的掩码广播指令;以及

执行经解码的掩码广播指令以组合所述源操作数的数据元素和所述第二源操作数的第二数据元素,以及将结果数据元素广播至由所述目的地操作数标识的目的地寄存器,其中目的地寄存器是掩码寄存器。

2.如权利要求1所述的方法,其特征在于,所述数据元素是源操作数中数据的最低有效位。

3.如权利要求1所述的方法,其特征在于,所述广播大小从掩码寄存器指令的名称导出。

4.如权利要求3所述的方法,其特征在于,所述广播大小从包括以下各项构成的组中选出:8位、16位、32位和64位。

5.如权利要求1所述的方法,其特征在于,所述源是512位寄存器。

6.如权利要求1所述的方法,其特征在于,并行地完成所述广播。

7.如权利要求1所述的方法,其特征在于,执行广播还包括将所述源的数据元素与另一个源的另一个数据元组组合成结果,并且将所述结果广播至目的地寄存器。

8.一种处理器,包括:

解码单元,用于解码包括目的地操作数、第一源操作数、第二源操作数和广播大小的掩码广播指令;以及执行单元,用于组合所述第一源操作数的数据元素和所述第二源操作数的第二数据元素作为广播数据,并且对于由所述目的地操作数标识的目的地操作数的每个目的地位置,将此广播数据存储到所述目的地位置,其中目的地操作数是掩码寄存器。

9.如权利要求8所述的处理器,其特征在于,所述处理器还用于对于每个目的地位置,将所述广播数据与第二源操作数的另一个数据元素组合。

10.如权利要求9所述的处理器,其特征在于,所述组合为AND操作。

11.如权利要求9所述的处理器,其特征在于,所述第二操作数是512位寄存器。

12.如权利要求9所述的处理器,其特征在于,所述组合并行地完成。

13.如权利要求8所述的处理器,其特征在于,所述目的地操作数是16位掩码寄存器。

14.如权利要求8所述的处理器,其特征在于,所述数据元素是所述源操作数中数据的最低有效位。

15.如权利要求8所述的处理器,其特征在于,所述广播大小从掩码寄存器指令的名称导出。

16.如权利要求15所述的处理器,其特征在于,所述广播大小从包括以下各项的组选出:8位、16位、32位和64位。

17.如权利要求9所述的处理器,其特征在于,所述第二源操作数是512位寄存器。

18.一种处理器,包括:

硬件解码器,用于解码掩码广播指令,其中所述掩码广播指令包括写目的地操作数、源操作数、第二源操作数和广播大小;以及

执行逻辑单元,用于组合所述源操作数的数据元素和所述第二源操作数的第二数据元

素，以及将结果数据元素广播至由所述目的地操作数标识的目的地寄存器，其中目的地操作数是掩码寄存器。

19.如权利要求18所述的处理器，其特征在于，还包括：

源寄存器，用于存储所述数据元素；以及

目的地寄存器，用于存储所广播的数据元素。

20.如权利要求18所述的处理器，其特征在于，所述数据元素是所述源操作数中数据的最低有效位。

21.如权利要求18所述的处理器，其特征在于，所述广播大小是以下之一：8位、16位、32位和64位，并且所述源操作数在掩码寄存器中。

22.一种处理器，包括：

解码器，用于解码掩码广播指令，其中所述掩码广播指令指示第一掩码寄存器、目的地掩码寄存器和广播大小；以及

执行单元，用于执行所述掩码广播指令以将所述第一掩码寄存器的位广播至具有所述广播大小的所述目的地掩码寄存器的多个位。

23.如权利要求22所述的处理器，所述位包括所述第一掩码寄存器中的最低有效位。

24.一种处理器，包括：

解码器，用于解码掩码广播指令，其中所述掩码广播指令指示第一掩码寄存器、第二掩码寄存器、目的地掩码寄存器和广播大小；以及

执行单元，用于执行所述掩码广播指令以将所述第一掩码寄存器的单个位与具有所述广播大小的所述第二掩码寄存器的多个位中的每一个组合，并将结果位存储在所述目的地掩码寄存器中，其中所述目的地掩码寄存器的大小基于所述广播大小。

25.如权利要求24所述的处理器，所述单个位包括所述第一掩码寄存器中的最低有效位。

26.一种或多种其上存储有指令的计算机可读介质，所述指令当由计算机处理器执行时使所述处理器执行如权利要求1至7中任一项所述的方法。

27.一种设备，包括用于执行如权利要求1至7中任一项所述的方法的装置。

掩码寄存器上的广播操作

发明领域

[0001] 本发明的领域一般涉及计算机处理器架构，更具体而言，涉及当执行时导致特定结果的指令。

[0002] 背景

[0003] 基于控制流信息合并来自矢量源的数据是基于矢量的架构的常见问题。例如，为了将以下代码矢量化，需要：1)生成指示a[i]>0是否为真的布尔矢量的方式以及2)基于布尔矢量从两个源(A[i]或B[i])选择任意值并将内容写入不同目的地(C[i])的方式。

[0004] For(i＝0;i<N;i++)

[0005] {

[0006] C[i]＝(a[i]>0?A[i]:B[i];

[0007] }

[0008] 为了使用掩码数据a[i]，用作为数组a[]的一部分的掩码数据填充一个或多个掩码寄存器。如果掩码数据用于从不同数组(诸如A[]和B[])选择数据，则掩码数据也被称为写掩码。

附图说明

[0009] 本发明是作为示例说明的，而不仅限制于各个附图的图形，在附图中，类似的参考编号表示类似的元件，其中：

[0010] 图1示出利用写掩码的示例。

[0011] 图2AB示出掩码广播指令的执行的示例。

[0012] 图3AB示出掩码广播指令的伪代码的示例。

[0013] 图4示出处理器中使用掩码广播指令的实施例。

[0014] 图5示出处理掩码广播指令的方法的实施例。

[0015] 图6示出处理掩码广播指令的方法的实施例。

[0016] 图7A、7B和7C是示出根据本发明的实施例的示例性专用矢量友好指令格式的框图。

[0017] 图8是根据本发明的一个实施例的寄存器架构的方框图。

[0018] 图9A是示出根据本发明的实施例的示例性有序流水线以及示例性寄存器重命名的无序发布/执行流水线的框图。

[0019] 图9B是示出根据本发明的实施例的有序架构核的示例性实施例以及包括在处理器中的示例性寄存器重命名的无序发布/执行架构核的框图。

[0020] 图10A和10B是根据本发明的实施例示出示例性无序架构的框图。

[0021] 图11是根据本发明的实施例示出具有一个以上的核的处理器的框图。

[0022] 图12示出根据本发明一个实施例的系统的框图。

[0023] 图13示出根据本发明的实施例的第二系统的框图。

[0024] 图14是根据本发明的实施例的第三系统的框图。

[0025] 图15是根据本发明的实施例的SoC的框图。

[0026] 图16是根据本发明的实施例的对比使用软件指令变换器将源指令集中的二进制指令变换成目标指令集中的二进制指令的框图。

具体实施方式

[0027] 在下面的描述中，阐述了很多具体细节。然而，应当理解，本发明的各实施例可以在不具有这些具体细节的情况下得到实施。在其他实例中，公知的电路、结构和技术未被详细示出以免混淆对本描述的理解。

[0028] 在说明书中对“一个实施例”、“一实施例”、“示例实施例”等的引用指示所描述的实施例可以包括特定特征、结构或特性，但并不一定每个实施例都需要包括该特定特征、结构或特性。此外，这样的短语不一定是指同一个实施例。此外，当结合一个影响例描述特定特征、结构或特性时，认为在本领域技术人员学识范围内，可以与其他影响例一起影响这样的特征、结构或特性，无论是否对此明确描述。

[0029] 指令集，或指令集架构(ISA)是涉及编程的计算机架构的一部分，并可以包括本机数据类型、指令、寄存器架构、寻址模式、存储器架构，中断和异常处理，以及外部输入和输出(I/O)。在本文中术语指令一般指宏指令——即被提供给处理器(或指令转换器，该指令转换器(例如使用静态二进制翻译、包括动态编译的动态二进制翻译)翻译、变形、仿真，或以其他方式将指令转换成要由处理器处理的一个或多个指令)的指令)以用于执行的指令——而不是微指令或微操作(micro-op)——它们是处理器的解码器解码宏指令的结果。

[0030] ISA与微架构不同，微架构是实现指令集的处理器的内部设计。带有不同的微架构的处理器可以共享共同的指令集。例如，INTEL® 奔腾四(Pentium4)处理器、Intel® 酷睿(Core™)处理器、以及来自加利福尼亚州桑尼威尔(Sunnyvale)的超微半导体有限公司(Advanced Micro Devices,Inc.)的诸多处理器执行几乎相同版本的x86指令集(在更新的版本中加入了一些扩展)，但具有不同的内部设计。例如，ISA的相同寄存器架构在不同的微架构中可使用已知的技术以不同方法来实现，包括专用物理寄存器、使用寄存器重命名机制(诸如，使用寄存器别名表RAT、重排序缓冲器ROB、以及隐退寄存器组；使用多映射和寄存器池)的一个或多个动态分配物理寄存器。除非另作说明，短语寄存器架构、寄存器组，以及寄存器在本文中被用来指代对软件/编程器以及指令指定寄存器的方式可见的东西。在需要特殊性的情况下，形容词逻辑、架构，或软件可见的将用于表示寄存器架构中的寄存器/文件，而不同的形容词将用于指定给定微型架构中的寄存器(例如，物理寄存器、重新排序缓冲器、退役寄存器、寄存器池)。

[0031] 指令集包括一个或多个指令格式。给定指令格式定义各个字段(位的数量、位的位置)以指定要执行的操作(操作码)以及对其要执行该操作的操作码等。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，给定指令格式的指令模板可被定义为具有指令格式的字段(所包括的字段通常在相同的阶中，但是至少一些字段具有不同的位位置，因为包括更少的字段)的不同子集，和/或被定义为具有不同解释的给定字段。由此，ISA的每一指令使用给定指令格式(并且如果定义，则在该指令格式的指令模板的给定一个中)来表达，并且包括用于指定操作和操作码的字段。例如，示例性ADD指令具有专用操作码以及包括指定该操作码的操作码字段和选择操作数的操作数字段(源1/目的地以及源2)的指令

格式，并且该ADD指令在指令流中的出现将具有选择专用操作数的操作数字段中的专用内容。

[0032] 科学、金融、自动矢量化的通用，RMS(识别、挖掘以及合成)，以及可视和多媒体应用程序(例如，2D/3D图形、图像处理、视频压缩/解压缩、语音识别算法和音频操纵)常常需要对大量的数据项执行相同操作(被称为“数据并行性”)。单指令多数据(SIMD)是指使处理器对多个数据项执行操作的一种指令。SIMD技术特别适于能够在逻辑上将寄存器中的位分割为若干个固定大小的数据元素的处理器，每一个元素都表示单独的值。例如，256位寄存器中的位可以被指定为四个单独的64位打包的数据元素(四字(Q)大小的数据元素)，八个单独的32位打包的数据元素(双字(D)大小的数据元素)，十六单独的16位打包的数据元素(一字(W)大小的数据元素)，或三十二个单独的8位数据元素(字节(B)大小的数据元素)来被操作的源操作数。这种类型的数据被称为打包的数据类型或矢量数据类型，这种数据类型的操作数被称为打包的数据操作数或矢量操作数。换句话说，打包数据项或矢量指的是打包数据元素的序列，并且打包数据操作数或矢量操作数是SIMD指令(也称为打包数据指令或矢量指令)的源操作数或目的地操作数。

[0033] 作为示例，一种类型的SIMD指令指定要以垂直方式对两个源矢量操作数执行的单个矢量运算，以利用相同数量的数据元素，以相同数据元素顺序，生成相同大小的目的地矢量操作数(也称为结果矢量操作数)。源矢量操作数中的数据元素被称为源数据元素，而目的地矢量操作数中的数据元素被称为目的地或结果数据元素。这些源矢量操作数是相同大小，并包含相同宽度的数据元素，如此，它们包含相同数量的数据元素。两个源矢量操作数中的相同位位置中的源数据元素形成数据元素对(也称为相对应的数据元素；即，每个源操作数的数据元素位置0中的数据元素相对应，每个源操作数的数据元素位置1中的数据元素相对应，等等)。由该SIMD指令所指定的操作分别地对这些源数据元素对中的每一对执行，以生成匹配的数量的结果数据元素，如此，每一对源数据元素都具有对应的结果数据元素。由于操作是垂直的并且由于结果矢量操作数大小相同，具有相同数量的数据元素，并且结果数据元素与源矢量操作数以相同数据元素顺序来存储，因此，结果数据元素与源矢量操作数中的它们的对应的源数据元素对处于结果矢量操作数的相同位位置。除此示例性类型的SIMD指令之外，还有各种其他类型的SIMD指令(例如，只有一个或具有两个以上的源矢量操作数的；以水平方式操作的；生成不同大小的结果矢量操作数的，具有不同大小的数据元素的，和/或具有不同的数据元素顺序的)。应该理解，术语目的地矢量操作数(或目的地操作数)被定义为执行由指令所指定的操作的直接结果，包括将该目的地操作数存储在某一位置(寄存器或在由该指令所指定的存储器地址)，以便它可以作为源操作数由另一指令访问(由另一指令指定该同一个位置)。

[0034] 诸如由具有包括x86、MMX™、流式SIMD扩展(SSE)、SSE2、SSE3、SSE4.1以及SSE4.2指令的指令集的Intel® Core™处理器使用的技术之类的SIMD技术，在应用程序性能方面实现了大大的改善。已经发布和/或公布了涉及高级矢量扩展(AVX)(AVX1和AVX2)且使用矢量扩展(VEX)编码方案的附加SIMD扩展集(例如，参见2011年10月的Intel® 64和IA-32架构软件开发手册，并且参见2011年6月的Intel®高级矢量扩展编程参考)。

[0035] 掩码广播

[0036] 以下是一般称为“掩码广播”的指令的实施例以及在包括背景技术中描述的各种不同领域中有益的可用于执行这一指令的系统、架构指令格式等的实施例。掩码广播指令的执行高效地处理具有掩码数据的掩码寄存器的加载。在一个实施例中，当掩码数据用于选择矢量寄存器的源数据时，掩码数据还被称为写掩码。换言之，掩码广播指令的执行导致处理器执行将数据从任一源或多个源广播到掩码寄存器。在一些实施例中，源中的至少一个是寄存器，诸如128位、256位、512位矢量寄存器等。在一些实施例中，源操作数中的至少一个是与开始存储器位置相关联的数据元素的集合。另外，在一些实施例中，一个或两个源的数据元素在任何掩码广播之前经过数据变换，诸如混合、广播、转换等(在本文中将讨论示例)。在另一个实施例中，目的地是寄存器，诸如8位掩码寄存器、16位掩码寄存器、32位掩码寄存器、64位掩码寄存器等。在一个实施例中，kbroadcast(k广播)指令可以是VEX类型的指令。

[0037] 该指令的示例性格式是“KBROADCAST{B/W/D/Q}k1,k2/存储器{k3}”，其中操作数k1是目的地掩码寄存器，k2/存储器是第一源，而k3是与第一源进行AND(与)操作的任选的其它源。在一个实施例中，KBROADCAST{B/W/D/Q}使用第一源并将第一源的内容中的一些或全部广播到目的地掩码寄存器。在一个实施例中，KBROADCAST{B/W/D/Q}使用源的最低有效位来广播至掩码寄存器。在另一个实施例中，第一源的内容的一些或全部与第二源的内容进行AND操作。此外，KBROADCAST{B/W/D/Q}将数据广播到目的地掩码寄存器中的连续位集合。广播的位的数量基于指令名的后缀。例如，在一个实施例中，对于512位示例寄存器上的结果掩码寄存器，“B”表示数据的六十四个位被广播，“W”表示数据的三十二个位(字)被广播，“D”表示数据的十六个位(双字)被广播，“Q”表示数据的八个位(四字)被广播。在一些实施例中，目的地写掩码也具有不同大小(8位、32位等)。KBROADCAST是指令的操作码。典型地，在指令中明确地定义每个操作数。可在指令的“前缀”中定义数据元素的大小，诸如通过使用类似稍后描述的“W”的数据粒度的指示。在大多数实施例中，W将指示每个数据元素是32位或64位。如果数据元素是32位大小，且源是512位大小，则每个源有十六(16)个数据元素。

[0038] 在图1中示出如何使用写掩码的示例。在该示例中，有两个源，每个源具有16个数据元素。在大多数情况下，这些源之一是寄存器(对于该示例，源1被视为512位寄存器，诸如具有16个32位数据元素的ZMM寄存器，然而，可使用其它数据元素和寄存器大小，诸如XMM和YMM寄存器和16位或64位数据元素)。其它(任选的)源是寄存器或存储器位置(在该图中源2是其它源)。如果第二源是存储器位置，则在大多数实施例中，在源的任意广播之前，将其置于临时寄存器中。另外，存储器位置的数据元素在置于临时寄存器中之前可经历数据变换。所示的掩码模式是0x5555。

[0039] 在该示例中，对于具有值“1”的写掩码的每个位位置，它是第二源(源2)的相应数据元素应被写入目的地寄存器的相应数据元素位置的指示。因此，源2的第一、第三、第五等位位置(B0、B2、B4等)被写入目的地的第一、第三、第五等数据元素位置。在写掩码具有“0”值的情况下，第一源的数据元素被写入目的地的对应数据元素位置。当然，取决于实现，可反转“1”和“0”的使用。另外，尽管该图和以上的描述将相应的第一位置视为最低有效位置，但在一些实施例中，第一位置是最高有效位置。

[0040] 图2A示出使用一个源的掩码广播指令的执行的示例。在图2A中，源200的内容被广

播到写掩码202。在一个实施例中，最低有效位从源200广播到每个写掩码。例如且在一个实施例中，源200的最低有效位被广播到写掩码202的最低有效位。作为另一个示例且在另一个实施例中，源200的最低有效位被广播到整个写掩码202。写入写掩码的位数量基于指令的后缀(例如，8、16、32、64位等)。例如且在一个实施例中，源200的最低有效位A0被广播到写掩码202的前八个位。

[0041] 图2B示出使用两个源的掩码广播指令的执行的示例。在图2B中，源252的内容与源254的内容进行AND操作，并且被广播到写掩码256。在一个实施例中，一个源的同一内容与其它源的不同内容进行AND操作。例如且在一个实施例中，源252的最低有效位与源254的不同内容进行AND操作。在该实施例中，这种AND操作的结果被存储到写掩码256的相应位置。例如且在一个实施例中，源252的最低有效位A0与源254的前八个位(例如，B7、B6、B5、B4、B3、B2、B1和B0)中的每一个进行AND操作。这些AND操作的结果被写入写掩码256的相应位。

[0042] 在代码序列中使用的k广播指令的示例如下：

```
for(i)
{
  bool useAlpha = bitTable[i];
  for(j)
  {
    ...
    if( useAlpha )
    {
      C[i][j] = Alpha[i][j] - Beta[i][j];
    }
    else
    {
      C[i][j] = Beta[i][j];
    }
  }
}
```

[0043]

[0044]

[0045] 在以上的代码中，标量布尔值useAlpha确定数组Alpha是否用于i行的所有元素。使用kbroadcast(k广播)指令，编译器可将useAlpha广播到掩码寄存器(即k1)。if语句归结为源Alpha和Beta在写掩码k1下作减法到C以及在k1的倒数下从Beta到C的移动。如果在“if”或“else”部分有另一个if条件(即，if B[i][j]>0)，则编译器可使用两个源k广播来合并useAlpha和B[i][j]>0掩码。

[0046] 图3A和3B示出掩码广播指令的不同实施例的伪代码的示例。在图3A中，伪代码302

示出来自一个源的掩码广播。在图3B中，伪代码352示出来自两个源的掩码广播，对这两个源进行AND以使其合并在一起。

[0047] 图4示出处理器中使用掩码广播指令的实施例。在401获取具有目的地操作数、两个源操作数、偏移(如果有的话)以及写掩码的掩码广播指令。在一些实施例中，目的地操作数是16位寄存器(诸如稍后详细描述的“k”掩码寄存器)。源操作数中的至少一个可以是存储器源操作数。在其它实施例中，一个源可以是掩码寄存器，而另一个源可以是存储器，或者两个源均可以是掩码寄存器。

[0048] 在403解码掩码广播指令。取决于指令的格式，在该阶段可解释各种数据，诸如如果有数据变换，则写入和检索哪些寄存器、访问哪些存储器地址等。

[0049] 在405检索/读取源操作数值。如果两个源是寄存器，则读取这些寄存器。如果源操作数之一或两者是存储器操作数，则检索与操作数相关联的数据元素。在一些实施例中，来自存储器的数据元素被存储在临时寄存器中。

[0050] 如果要执行任何数据元素变换(诸如上转换、广播、混合等，这些稍后将详细描述)，则可在407执行。例如，可将来自存储器的16位数据元素上转换成32位数据元素，或者可将数据元素从一个模式混合成另一个(例如，XYZWXYZW XYZW…XYZW至XXXXXXXX YYYYYYYY ZZZZZZZZZZWWWWWWWW)。

[0051] 在409，由执行资源执行掩码广播指令(或者操作包括这一指令，诸如微操作)。该执行导致数据从一个或多个源广播至目的地掩码寄存器。例如，在掩码寄存器的连续位集合上广播源操作数的数据元素的最低有效位。作为另一个示例，一个源的最低有效位与来自另一个源的数据进行AND操作，其中AND操作的结果被存储到掩码寄存器中的相应位置中。在图2AB中示出这一掩码广播的示例。

[0052] 在411将掩码广播的结果数据元素存储到目的地寄存器中。而且，在图2AB中示出其示例。尽管分别地示出了409和411，但是在一些实施例中，它们是作为指令的执行的一部分一起执行的。

[0053] 尽管以上已经示出一种类型的执行环境，但它易于修改以符合其它环境，诸如以下详细描述的有序和无序环境。

[0054] 图5示出处理掩码广播指令的方法的实施例。在此实施例中，假设早先已经执行操作401-407中的某些，如果不是全部，然而，没有示出它们，以便不使下面呈现的细节模糊。例如，没有示出获取和解码，也没有示出操作数(源和目的地)检索。

[0055] 在501，接收第一源数据、任选的第二源数据和目的地数据大小。例如，从第一源操作数接收第一源数据的第一源数据元素。在一个实施例中，第一源数据元素是存储在第一源操作数中的第一源数据元素的最低有效位。作为另一个示例，从第二源操作数接收任选的第二源数据。在一些实施例中，从对应的指令操作数接收目的地大小。在另一个实施例中，目的地大小基于指令名称是固定的。在该实施例中，指令名称的前缀确定目的地大小。例如，在一个实施例中，对于512位示例寄存器上的结果掩码寄存器，“B”表示数据的六十四个位被广播，“W”表示数据的三十二个位(字)被广播，“D”表示数据的十六个位(双字)被广播，“Q”表示数据的八个位(四字)被广播。”

[0056] 在503-511，执行循环以将数据广播到掩码寄存器。在505，将广播数据设定为第一源数据。例如，第一源数据的数据元素的最低有效位是广播数据。尽管在一个实施例中，贯

穿循环，第一源数据是相同的，但在替换实施例中，在环执行期间第一源数据可改变。在507，如果使用第二源数据，则将对应的第二源数据与广播数据进行AND操作。例如，如图2B所示，源252的内容与源254的内容进行AND操作，并且被广播到掩码寄存器256。如果不使用第二源，则在507不执行操作。在509，将广播数据复制到相应的目的地位置。例如，如图2A所述，将源202的内容复制到适当的目的地位置204。在511，循环结束。

[0057] 图6示出处理掩码广播指令的方法的实施例。在该实施例中，假设在601之前，已经执行操作401-407中的一些而非全部。在601，确定目的地位位置中的每一个的值需要两个源的组合。

[0058] 如果掩码广播值来自一个源，则在603，对于写掩码的每个目的地位位置，将相应的值存储在该目的地位位置。例如，如以上图2A所述，将源的最低有效位存储在写掩码的相应位位置。如果掩码广播值是源的组合，则在605，对于写掩码的每个目的地位位置，对相应的源值进行AND操作以合并在一起并且将结果值存储在该目的地位位置。例如，源252的最低有效位A0与源254的前八个位进行AND操作，其中结果值被写入写掩码256的相应位位置，如以上图2B所述。在一些实施例中，并行地执行603和605。

[0059] 尽管图5和6已经讨论了基于来自第一源的单个位的掩码广播，但可预想其它实施例(使用位模式的多于单个广播的掩码广播)。另外，应当清楚地理解可使用其它类型的掩码广播。将掩码广播作为单个指令的优点在于程序将具有较小的二进制，该二进制具有指令高速缓存暗示。例如且在一个实施例中，在执行期间，在流水线上对于获取、解码、执行资源而言具有较小压力。结果，该程序可能执行得更快。

[0060] 示例性指令格式

[0061] 本文中所描述的指令的实施例可以不同的格式体现。另外，在下文中详述示例性系统、架构、以及流水线。指令的实施例可在这些系统、架构、以及流水线上执行，但是不限于详述的系统、架构、以及流水线。

[0062] VEX指令格式

[0063] VEX编码允许指令具有两个以上操作数，并且允许SIMD矢量寄存器比128位长。VEX前缀的使用提供了三个操作数(或者更多)句法。例如，先前的两个操作数指令执行改写源操作数的操作(诸如A＝A+B)。VEX前缀的使用使操作数执行非破坏性操作，诸如A＝B+C。

[0064] 图7A示出示例性AVX指令格式，包括VEX前缀702、实操作码字段730、MoD R/M字节740、SIB字节750、位移字段762、以及IMM8772。图7B示出来自图7A的哪些字段构成完整操作码字段774和基础操作字段742。图7C示出来自图7A的哪些字段构成寄存器索引字段744。

[0065] VEX前缀(字节0-2)702以三字节形式进行编码。第一字节是格式字段740(VEX字节0，位[7:0])，该格式字段1140包含明确的C4字节值(用于区分C4指令格式的唯一值)。第二-第三字节(VEX字节1-2)包括提供专用能力的大量位字段。具体地，REX字段705(VEX字节1，位[7-5])由VEX.R位字段(VEX字节1，位[7]-R)、VEX.X位字段(VEX字节1，位[6]-X)以及VEX.B位字段(VEX字节1，位[5]-B)组成。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以及bbb)进行编码，由此Rrrr、Xxxx以及Bbbb可通过增加VEX.R、VEX.X以及VEX.B来形成。操作码映射字段715(VEX字节1，位[4:0]-mmmmm)包括对隐含的领先操作码字节进行编码的内容。W字段764(VEX字节2，位[7]-W)由记号VEX.W表示，并且取决于该指令提供了不同的功能。VEX.vvvv720(VEX字节2，位[6:3]-vvvv)的作用可包括

如下:1)VEX.vvvv对以颠倒(1(多个)补码)的形式指定第一源寄存器操作数进行编码,且对具有两个或两个以上源操作数的指令有效;2)VEX.vvvv针对特定矢量位移对以1(多个)补码的形式指定的目的地寄存器操作数进行编码;或者3)VEX.vvvv不对任何操作数进行编码,保留该字段,并且应当包含1111b。如果VEX.L768大小的字段(VEX字节2,位[2]-L)=0,则它指示128位矢量;如果VEX.L=1,则它指示256位矢量。前缀编码字段725(VEX字节2,位[1:0]-pp)提供了用于基础操作字段的附加位。

[0066] 实操作码字段730(字节3)还被称为操作码字节。操作码的一部分在该字段中指定。

[0067] MOD R/M字段740(字节4)包括MOD字段742(位[7-6])、Reg字段744(位[5-3])、以及R/M字段746(位[2-0])。Reg字段744的作用可包括如下:对目的地寄存器操作数或源寄存器操作数(Rfff中的rrr)进行编码;或者被视为操作码扩展且不用于对任何指令操作数进行编码。R/M字段746的作用可包括如下:对参考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。

[0068] 缩放索引基址(SIB)-缩放字段750(字节5)的内容包括用于存储器地址生成的SS752(位[7-6])。先前已经针对寄存器索引Xxxx和Bbbb参考了SIB.xxx754(位[5-3])和SIB.bbb756(位[2-0])的内容。

[0069] 位移字段762和立即数字段(IMM8)772包含地址数据。

[0070] 示例性编码成VEX

[0071] 在以下的附件A中示出对于指令的示例性编码成VEX。

[0072] 示例性编码成具体的示例友好指令格式

[0073] 示例性寄存器架构

[0074] 图8是根据本发明的一个实施例的寄存器架构800的框图。在所示出的实施例中,有32个512位宽的矢量寄存器810;这些寄存器被引用为zmm0到zmm31。较低的16zmm寄存器的较低阶256个位覆盖在寄存器ymm0-16上。较低的16zmm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。

[0075] 写掩码寄存器815-在所示的实施例中,存在8个写掩码寄存器(k0至k7),每一写掩码寄存器的大小是64位。在替换实施例中,写掩码寄存器815的大小是16位。如先前所述的,在本发明的一个实施例中,矢量掩码寄存器k0无法用作写掩码;当正常可指示k0的编码用作写掩码时,它选择硬连线的写掩码0xFFFF,从而有效地停用该指令的写掩码。

[0076] 通用寄存器825——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寻址模式来寻址存储器操作数一起使用。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。

[0077] 标量浮点堆栈寄存器组(x87堆栈)845,在其上面混叠MMX打包整数平坦寄存器组850——在所示出的实施例中,x87堆栈是用于使用x87指令集扩展来对32/64/80位浮点数据执行标量浮点运算的八元素堆栈;而使用MMX寄存器来对64位打包整数数据执行操作,以及为在MMX和XMM寄存器之间执行的某些操作保存操作数。

[0078] 本发明的替换实施例可以使用较宽的或较窄的寄存器。另外,本发明的替换实施例可以使用多一些,少一些或不同的寄存器组和寄存器。

[0079] 示例性核架构、处理器和计算机架构

[0080] 处理器核可以用出于不同目的的不同方式在不同的处理器中实现。例如，这样的核的实现可以包括：1)旨在用于通用计算的通用有序核；2)预期用于通用计算的高性能通用无序核；3)主要预期用于图形和/或科学(吞吐量)计算的专用核。不同处理器的实现可包括：包括预期用于通用计算的一个或多个通用有序核和/或预期用于通用计算的一个或多个通用无序核的CPU；以及2)包括主要预期用于图形和/或科学(吞吐量)的一个或多个专用核的协处理器。这样的不同处理器导致不同的计算机系统架构，其可包括：1)在与CPU分开的芯片上的协处理器；2)在与CPU相同的封装中但分开的管芯上的协处理器；3)与CPU在相同管芯上的协处理器(在该情况下，这样的协处理器有时被称为诸如集成图形和/或科学(吞吐量)逻辑等专用逻辑，或被称为专用核)；以及4)可以将所描述的CPU(有时被称为应用核或应用处理器)、以上描述的协处理器和附加功能包括在同一管芯上的片上系统。接着描述示例性核架构，随后描述示例性处理器和计算机架构。

[0081] 示例性核架构

[0082] 有序和无序核框图

[0083] 图9A是示出根据本发明的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无序发布/执行流水线的框图。图9B是示出根据本发明的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无序发布/执行架构核的框图。图9A-10B中的实线框解说了有序流水线和有序核，而虚线框中的可选附加项解说了寄存器重命名的、无序发布/执行流水线和核。给定有序方面是无序方面的子集的情况下，无序方面将被描述。

[0084] 在图9A中，处理器流水线900包括提取级902、长度解码级904、解码级906、分配级908、重命名级910、调度(也称为分派或发布)级912、寄存器读/存储器读取级914、执行级916、写回/存储器写入级918、异常处理级922和提交级924。

[0085] 图9B示出了包括耦合到执行引擎单元950的前端单元930的处理器核990，且执行引擎单元和前端单元两者都耦合到存储器单元970。核990可以是精简指令集合计算(RISC)核、复杂指令集合计算(CISC)核、非常长的指令字(VLIW)核或混合或替代核类型。作为又一选项，核990可以是专用核，诸如例如网络或通信核、压缩引擎、协处理器核、通用计算图形处理器单元(GPGPU)核、或图形核等等。

[0086] 前端单元930包括耦合到指令高速缓存单元934的分支预测单元932，该指令高速缓存单元934被耦合到指令翻译后备缓冲器(TLB)936，该指令翻译后备缓冲器936被耦合到指令获取单元938，指令获取单元938被耦合到解码单元940。解码单元940(或解码器)可解码指令，并生成从原始指令解码出的、或以其他方式反映原始指令的、或从原始指令导出的一个或多个微操作、微代码进入点、微指令、其他指令、或其他控制信号作为输出。解码单元940可使用各种不同的机制来实现。合适的机制的示例包括但不限于查找表、硬件实现、可编程逻辑阵列(OLA)、微代码只读存储器(ROM)等。在一个实施例中，核990包括存储(例如，在解码单元940中或否则在前端单元930内的)某些宏指令的微代码的微代码ROM或其他介质。解码单元940耦合到执行引擎单元950中的重命名/分配器单元952。

[0087] 执行引擎单元950包括重命名/分配器单元952，该重命名/分配器单元952耦合至引退单元954和一个或多个调度器单元956的集合。调度器单元956表示任何数目的不同调度器，包括预留站、中央指令窗等。调度器单元956被耦合到物理寄存器组单元958。每个物

理寄存器组单元958表示一个或多个物理寄存器组，其中不同的物理寄存器组存储一种或多种不同的数据类型，诸如标量整数、标量浮点、打包整数、打包浮点、矢量整数、矢量浮点、状态(例如，作为要执行的下一指令的地址的指令指针)等。在一个实施例中，物理寄存器组单元958包括矢量寄存器单元、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矢量寄存器、矢量掩码寄存器、和通用寄存器。物理寄存器组单元958被引退单元954覆盖以示出可以用来实现寄存器重命名和无序执行的各种方式(例如，使用记录器缓冲器和引退寄存器组；使用将来的文件、历史缓冲器和引退寄存器组；使用寄存器图和寄存器池等等)。引退单元954和物理寄存器组单元958被耦合到执行群集960。执行群集960包括一个或多个执行单元962的集合和一个或多个存储器访问单元964的集合。执行单元962可以执行各种操作(例如，移位、加法、减法、乘法)，以及对各种类型的数据(例如，标量浮点、打包整数、打包浮点、矢量整型、矢量浮点)执行。尽管某些实施例可以包括专用于特定功能或功能集合的多个执行单元，但其他实施例可包括全部执行所有函数的仅一个执行单元或多个执行单元。调度器单元956、物理寄存器组单元958和执行群集960被示为可能有多个，因为某些实施例为某些类型的数据/操作(例如，标量整型流水线、标量浮点/打包整型/打包浮点/矢量整型/矢量浮点流水线，和/或各自具有其自己的调度器单元、物理寄存器单元和/或执行群集的存储器访问流水线——以及在分开的存储器访问流水线的情况下，实现其中仅该流水线的执行群集具有存储器访问单元964的某些实施例)创建分开的流水线。还应当理解，在分开的流水线被使用的情况下，这些流水线中的一个或多个可以为无序发布/执行，并且其余流水线可以为有序发布/执行。

[0088] 存储器访问单元964的集合被耦合到存储器单元970，该存储器单元970包括耦合到数据高速缓存单元974的数据TLB单元972，其中数据高速缓存单元974耦合到二级(L2)高速缓存单元976。在一个示例性实施例中，存储器存取单元964可包括加载单元、存储地址单元、以及存储数据单元，这些单元中的每一个耦合到存储器单元970中的数据TLB单元972。指令高速缓存单元934还耦合到存储器单元970中的第二级(L2)高速缓存单元976。L2高速缓存单元976被耦合到一个或多个其他级的高速缓存，并最终耦合到主存储器。

[0089] 作为示例，示例性寄存器重命名的、无序发布/执行核架构可以如下实现流水线900：1)指令获取938执行获取和长度解码级902和904；2)解码单元940执行解码级906；3)重命名/分配器单元952执行分配级908和重命名级910；4)调度器单元956执行调度级912；5)物理寄存器组单元958和存储器单元970执行寄存器读取/存储器读取级914；执行群集960执行执行级916；6)存储器单元970和物理寄存器组单元958执行写回/存储器写入级918；7)各单元可牵涉到异常处理级922；以及8)引退单元954和物理寄存器组单元958执行提交级924。

[0090] 核990可支持一个或多个指令集合(例如，x86指令集合(具有与较新版本一起添加的某些扩展)；加利福尼亚州桑尼维尔市的MIPS技术公司的MIPS指令集合；加利福尼州桑尼维尔市的ARM控股的ARM指令集合(具有诸如NEON等可选附加扩展))，其中包括本文中描述的各指令。在一个实施例中，核990包括支持打包数据指令集合扩展(例如，AVX1、AVX2等)的逻辑，由此允许被许多多媒体应用使用的操作将使用打包数据来执行。

[0091] 应当理解，核可支持多线程化(执行两个或更多个并行的操作或线程的集合)，并且可以按各种方式来完成该多线程化，此各种方式包括时分多线程化、同步多线程化(其中

单个物理核为物理核正同步多线程化的各线程中的每一个线程提供逻辑核)、或其组合(例如,时分提取和解码以及此后诸如用Intel® 超线程化技术来同步多线程化)。

[0092] 尽管在无序执行的上下文中描述了寄存器重命名,但应当理解,可以在有序架构中使用寄存器重命名。尽管所解说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934/974以及共享L2高速缓存单元976,但替换实施例可以具有用于指令和数据两者的单个内部高速缓存,诸如例如一级(L1)内部高速缓存或多个级别的内部缓存。在某些实施例中,该系统可包括内部高速缓存和在核和/或处理器外部的外部高速缓存的组合。或者,所有高速缓存都可以在核和/或处理器的外部。

[0093] 具体的示例性有序核架构

[0094] 图10A-B示出了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)。这些逻辑块通过高带宽的互连网络(例如,环形网络)与某些固定的功能逻辑、存储器I/O接口和其它必要的I/O逻辑通信,这依赖于应用。

[0095] 图10A是根据本发明的各实施例的单个处理器核连同它与管芯上互连网络1002的连接以及其二级(L2)高速缓存1004的本地子集的框图。在一个实施例中,指令解码器1000支持具有打包数据指令集合扩展的x86指令集。L1高速缓存1006允许对标量和矢量单元中的高速缓存存储器的低等待时间访问。尽管在一个实施例中(为了简化设计),标量单元1008和矢量单元1010使用分开的寄存器集合(分别为标量寄存器1012和矢量寄存器1014),并且在这些寄存器之间转移的数据被写入到存储器并随后从一级(L1)高速缓存1006读回,但是本发明的替换实施例可以使用不同的方法(例如使用单个寄存器集合或包括允许数据在这两个寄存器组之间传输而无需被写入和读回的通信路径)。

[0096] L2高速缓存的本地子集1004是全局L2高速缓存的一部分,该全局L2高速缓存被划分成多个分开的本地子集,即每个处理器核一个本地子集。每个处理器核具有到其自己的L2高速缓存1004的本地子集的直接访问路径。被处理器核读出的数据被存储在其L2高速缓存子集1004中,并且可以被快速访问,该访问与其他处理器核访问其自己的本地L2高速缓存子集并行。被处理器核写入的数据被存储在其子集的L2高速缓存子集1004中,并在必要的情况下从其它子集清除。环形网络确保共享数据的一致性。环形网络是双向的,以允许诸如处理器核、L2高速缓存和其它逻辑块之类的代理在芯片内彼此通信。每个环形数据路径为每个方向1012位宽。

[0097] 图10B是根据本发明的各实施例的图10A中的处理器核的一部分的展开图。图10B包括作为L1高速缓存1004的L1数据高速缓存1006A部分,以及关于矢量单元1010和矢量寄存器1014的更多细节。具体地说,矢量单元1010是16宽矢量处理单元(VPU)(见16宽ALU1028),该单元执行整型、单精度浮点以及双精度浮点指令中的一个或多个。该VPU通过混合单元1020支持对寄存器输入的混合、通过数值转换单元1022A-B支持数值转换,并通过复制单元1024支持对存储器输入的复制。写掩码寄存器1026允许断言所得的矢量写入。

[0098] 具有集成存储器控制器和图形器件的处理器

[0099] 图11是根据本发明的实施例的可具有一个以上核、可具有集成存储器控制器、并且可具有集成图形的处理器1100的方框图。图11中的实线框示出具有单一核1102A、系统代理1100、一组一个或多个总线控制器单元1116的处理器1100,而任选增加的虚线框示出具

有多个核1102A-N、系统代理单元1110中的一组一个或多个集成存储器控制器单元1114、以及专用逻辑1108的替换处理器1100。

[0100] 因此，处理器1100的不同实现可包括：1)CPU，其中专用逻辑1108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1102A-N是一个或多个通用核(例如，通用的有序核、通用的无序核、这两者的组合)；2)协处理器，其中核1102A-N是主要预期用于图形和/或科学(吞吐量)的大量专用核；以及3)协处理器，其中核1102A-N是大量通用有序核。因此，处理器1100可以是通用处理器、协处理器或专用处理器，诸如例如网络或通信处理器、压缩引擎、图形处理器、GPGPU(通用图形处理单元)、高吞吐量的集成众核(MIC)协处理器(包括30个或更多核)、或嵌入式处理器等。该处理器可以被实现在一个或多个芯片上。处理器1100可以是一个或多个衬底的一部分，和/或可以使用诸如例如BiCMOS、CMOS或NMOS等的多个加工技术中的任何一个技术将其实现在一个或多个衬底上。

[0101] 存储器层次结构包括在各核内的一个或多个级别的高速缓存、一个或多个共享高速缓存单元1106的集合、以及耦合至集成存储器控制器单元1114的集合的外部存储器(未示出)。该共享高速缓存单元1106的集合可以包括一个或多个中间级高速缓存，诸如二级(L2)、三级(L3)、四级(L4)或其他级别的高速缓存、末级高速缓存(LLC)、和/或其组合。尽管在一个实施例中，基于环的互连单元1112将集成图形逻辑1108、共享高速缓存单元1106的集合以及系统代理单元1110/集成存储器控制器单元1114互连，但替代实施例可使用任何数量的公知技术来将这些单元互连。在一个实施例中，在一个或多个高速缓存单元1106与核1102A-N之间维持一致性。

[0102] 在某些实施例中，核1102A-N中的一个或多个核能够多线程化。系统代理1110包括协调和操作核1102A-N的那些组件。系统代理单元1110可包括例如功率控制单元(PCU)和显示单元。PCU可以是或包括调整核1102A-N和集成图形逻辑1108的功率状态所需的逻辑和组件。显示单元用于驱动一个或多个外部连接的显示器。

[0103] 核1102A-N在架构指令集合方面可以是同构的或异构的；即，这些核1102A-N中的两个或更多个核可能能够执行相同的指令集合，而其他核可能能够执行该指令集合的仅仅子集或不同的指令集合。

[0104] 示例性计算机架构

[0105] 图12-15是示例性计算机架构的框图。本领域已知的对膝上型设备、台式机、手持PC、个人数字助理、工程工作站、服务器、网络设备、网络集线器、交换机、嵌入式处理器、数字信号处理器(DSP)、图形设备、视频游戏设备、机顶盒、微控制器、蜂窝电话、便携式媒体播放器、手持设备以及各种其他电子设备的其他系统设计和配置也是合适的。一般来说，能够纳入本文中所公开的处理器和/或其它执行逻辑的大量系统和电子设备一般都是合适的。

[0106] 现在参考图12，示出了根据本发明的一个实施例的系统1200的方框图。系统1200可以包括一个或多个处理器1210、1215，这些处理器耦合到控制器中枢1220。在一个实施例中，控制器中枢1220包括图形存储器控制器中枢(GMCH)1290和输入/输出中枢(IOH)1250(其可以在分开的芯片上)；GMCH1290包括存储器1240和协处理器1245耦合到的存储器和图形控制器；IOH1250将输入/输出(I/O)设备1260耦合到GMCH1290。替换地，存储器和图形控制器中的一个或两个在处理器(如本文中所描述的)内集成，存储器1240和协处理器1245直接耦合到处理器1210、以及单一芯片中的具有IOH1250的控制器中枢1220。

[0107] 附加处理器1215的可选性质用虚线表示在图12中。每一处理器1210、1215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1100的某一版本。

[0108] 存储器1240可以是例如动态随机存取存储器(DRAM)、相变化存储器(PCM)或这两者的组合。对于至少一个实施例，控制器中枢1220经由诸如前侧总线(FSB)之类的多点总线(multi-drop bus)、诸如快速通道互连(QPI)之类的点对点接口、或者类似的连接1295与处理器1210、1215进行通信。

[0109] 在一个实施例中，协处理器1245是专用处理器，诸如例如高吞吐量MIC处理器、网络或通信处理器、压缩引擎、图形处理器、GPGPU、或嵌入式处理器等等。在一个实施例中，控制器中枢1220可以包括集成图形加速计。

[0110] 在包括架构、微架构、热、功耗特性等的优点度量的范围方面，在物理资源1210、1215之间可存在各种差异。

[0111] 在一个实施例中，处理器1210执行控制一般类型的数据处理操作的指令。嵌入在这些指令中的可以是协处理器指令。处理器1210识别如具有应当由附连的协处理器1245执行的类型的这些协处理器指令。因此，处理器1210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1245。协处理器1245接受并执行所接收的协处理器指令。

[0112] 现在参考图13，示出了根据本发明的一个实施例的第一更具体的示例性系统1300的方框图。如图13所示，多处理器系统1300是点对点互连系统，并包括经由点对点互连1350耦合的第一处理器1370和第二处理器1380。处理器1370和1380中的每一个都可以是处理器1100的某一版本。在本发明的一个实施例中，处理器1370和1380分别是处理器1210和1215，而协处理器1338是协处理器1245。在另一实施例中，处理器1370和1380分别是处理器1210和协处理器1245。

[0113] 处理器1370和1380被示为分别包括集成存储器控制器(IMC)单元1372和1382。处理器1370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1376和1378；类似地，第二处理器1380包括点对点接口1386和1388。处理器1370、1380可以使用点对点(P-P)电路1378、1388经由P-P接口1350来交换信息。如图13所示，IMC1372和1382将各处理器耦合至相应的存储器，即存储器1332和存储器1334，这些存储器可以是本地附连至相应的处理器的主存储器的一部分。

[0114] 处理器1370、1380可各自经由使用点对点接口电路1390、1394、1386、1398的各个P-P接口1352、1354与芯片组1390交换信息。芯片组1390可以可选地经由高性能接口1339与协处理器1338交换信息。在一个实施例中，协处理器1338是专用处理器，诸如例如高吞吐量MIC处理器、网络或通信处理器、压缩引擎、图形处理器、GPGPU、或嵌入式处理器等等。

[0115] 共享高速缓存(未示出)可以被包括在任一处理器之内或被包括两个处理器外部但仍经由P-P互连与这些处理器连接，从而如果将某处理器置于低功率模式时，可将任一处理器或两个处理器的本地高速缓存信息存储在该共享高速缓存中。

[0116] 芯片组1390可经由接口1396耦合至第一总线1316。在一个实施例中，第一总线1316可以是外围部件互连(PCI)总线，或诸如PCI Express总线或其它第三代I/O互连总线之类的总线，但本发明的范围并不受此限制。

[0117] 如图13所示，各种I/O设备1314可以连同总线桥1318耦合到第一总线1316，总线桥

1318将第一总线1316耦合至第二总线1320。在一个实施例中，诸如协处理器、高吞吐量MIC处理器、GPGPU的处理器、加速计(诸如例如图形加速计或数字信号处理器(DSP)单元)、场可编程门阵列或任何其他处理器的一个或多个附加处理器1315被耦合到第一总线1316。在一个实施例中，第二总线1320可以是低引脚计数(LPC)总线。各种设备可以被耦合至第二总线1320，在一个实施例中这些设备包括例如键盘/鼠标1322、通信设备1327以及诸如可包括指令/代码和数据1328的盘驱动器或其它海量存储设备的存储单元1330。此外，音频I/O1324可以被耦合至第二总线1320。注意，其它架构是可能的。例如，取代图13的点对点架构，系统可以实现多站总线或其它这类架构。

[0118] 现在参考图14，示出了根据本发明的一个实施例的第二更具体的示例性系统1400的方框图。图13和14中的相似元件具有相似的附图标记，并且图13的特定方面已经从图14中省略以避免混淆图14的其他方面。

[0119] 图14示出处理器1370、1380可分别包括集成存储器和I/O控制逻辑(“CL”)1372和1382。因此，CL1372、1382包括集成存储器控制器单元并包括I/O控制逻辑。图14不仅解说了耦合至CL1372、1382的存储器1332、1334，而且还解说了同样耦合至控制逻辑1372、1382的I/O设备1414。传统I/O设备1415被耦合至芯片组1390。

[0120] 现在参考图15，示出了根据本发明的一个实施例的SoC1500的方框图。在图11中，相似的部件具有同样的附图标记。另外，虚线框是更先进的SoC的可选特征。在图15中，互连单元1502被耦合至：应用处理器1510，该应用处理器包括一个或多个核202A-N的集合以及共享高速缓存单元1106；系统代理单元1110；总线控制器单元1116；集成存储器控制器单元1114；一组或一个或多个协处理器1520，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1530；直接存储器存取(DMA)单元1532；以及用于耦合至一个或多个外部显示器的显示单元1540。在一个实施例中，协处理器1520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。

[0121] 本文公开的机制的各实施例可以被实现在硬件、软件、固件或这些实现方法的组合中。本发明的实施例可实现为在可编程系统上执行的计算机程序或程序代码，该可编程系统包括至少一个处理器、存储系统(包括易失性和非易失性存储器和/或存储元件)、至少一个输入设备以及至少一个输出设备。

[0122] 可将程序代码(诸如图13中解说的代码1330)应用于输入指令，以执行本文描述的各功能并生成输出信息。输出信息可以按已知方式被应用于一个或多个输出设备。为了本申请的目的，处理系统包括具有诸如例如数字信号处理器(DSP)、微控制器、专用集成电路(ASIC)或微处理器之类的处理器的任何系统。

[0123] 程序代码可以用高级程序化语言或面向对象的编程语言来实现，以便与处理系统通信。程序代码也可以在需要的情况下用汇编语言或机器语言来实现。事实上，本文中描述的机制不仅限于任何特定编程语言的范围。在任一情形下，语言可以是编译语言或解释语言。

[0124] 至少一个实施例的一个或多个方面可以通过存储在机器可读介质上的代表性的指令来实现，指令表示处理器内的各种逻辑，指令在由机器读取时使机器制造执行此处所描述的技术的逻辑。被称为“IP核”的这些表示可以被存储在有形的机器可读介质上，并被

提供给多个客户或生产设施以加载到实际制造该逻辑或处理器的制造机器中。

[0125] 这样的机器可读存储介质可以包括但不限于通过机器或设备制造或形成的物品的非瞬态、有形安排，其包括存储介质，诸如硬盘；任何其它类型的盘，包括软盘、光盘、紧致盘只读存储器(CD-ROM)、紧致盘可重写(CD-RW)的以及磁光盘；半导体器件，例如只读存储器(ROM)、诸如动态随机存取存储器(DRAM)和静态随机存取存储器(SRAM)的随机存取存储器(RAM)、可擦除可编程只读存储器(EPROM)、闪存、电可擦除可编程只读存储器(EEPROM)；相变化存储器(PCM)；磁卡或光卡；或适于存储电子指令的任何其它类型的介质。

[0126] 因此，本发明的各实施例还包括非瞬态、有形机器可读介质，该介质包含指令或包含设计数据，诸如硬件描述语言(HDL)，它定义本文中描述的结构、电路、装置、处理器和/或系统特性。这些实施例也被称为程序产品。

[0127] 仿真(包括二进制变换、代码变形等)

[0128] 在某些情况下，指令转换器可用来将指令从源指令集转换至目标指令集。例如，指令转换器可以变换(例如使用静态二进制变换、包括动态编译的动态二进制变换)、变形、仿真或以其它方式将指令转换成将由核来处理的一个或多个其它指令。指令转换器可以用软件、硬件、固件、或其组合实现。指令转换器可以在处理器上、在处理器外、或者部分在处理器上部分在处理器外。

[0129] 图16是根据本发明的实施例的对比使用软件指令变换器将源指令集中的二进制指令变换成目标指令集中的二进制指令的框图。在所示的实施例中，指令转换器是软件指令转换器，但作为替代该指令转换器可以用软件、固件、硬件或其各种组合来实现。图16示出了用高级语言1602的程序可以使用x86编译器1604来编译，以生成可以由具有至少一个x86指令集核1616的处理器原生执行的x86二进制代码1606。具有至少一个x86指令集核1616的处理器表示任何处理器，这些处理器能通过兼容地执行或以其他方式处理以下内容来执行与具有至少一个x86指令集核的英特尔处理器基本相同的功能：1)英特尔x86指令集核的指令集的本质部分，或2)被定向为在具有至少一个x86指令集核的英特尔处理器上运行的应用或其它程序的对象代码版本，以便取得与具有至少一个x86指令集核的英特尔处理器基本相同的结果。x86编译器1604表示用于生成x86二进制代码1606(例如，对象代码)的编译器，该二进制代码706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1616的处理器上执行。类似地，图16示出用高级语言1602的程序可以使用替代的指令集编译器1608来编译，以生成可以由不具有至少一个x86指令集核1614的处理器(例如具有执行加利福尼亚州桑尼维尔市的MIPS技术公司的MIPS指令集，和/或执行加利福尼亚州桑尼维尔市的ARM控股公司的ARM指令集的核的处理器)原生执行的替代指令集二进制代码1610。指令转换器1612被用来将x86二进制代码1606转换成可以由不具有x86指令集核1614的处理器原生执行的代码。该转换后的代码不大可能与替换性指令集二进制代码1610相同，因为能够这样做的指令转换器难以制造；然而，转换后的代码将完成一般操作并由来自替换性指令集的指令构成。因此，指令转换器1612通过仿真、模拟或任何其它过程来表示允许不具有x86指令集处理器或核的处理器或其它电子设备执行x86二进制代码1606的软件、固件、硬件或其组合。

[0130] 本文公开的矢量友好指令格式的指令的某些操作可由硬件组件执行，且可体现在机器可执行指令中，该指令用于导致或至少致使电路或其它硬件组件以执行该操作的指令

编程。电路可包括通用或专用处理器、或逻辑电路，这里仅给出几个示例。这些操作还可任选地由硬件和软件的组合执行。执行逻辑和/或处理器可包括响应于从机器指令导出的机器指令或一个或多个控制信号以存储指令指定的结果操作数的专用或特定电路或其它逻辑。例如，本文公开的指令的实施例可在图12-15的一个或多个系统中执行，且矢量友好指令格式的指令的实施例可存储在将在系统中执行的程序代码中。另外这些附图的处理元件可利用本文详细描述的详细描述的流水线和/或架构(例如有序和无序架构)之一。例如，有序架构的解码单元可解码指令、将经解码的指令传送到矢量或标量单元等。

[0131] 上述描述旨在说明本发明的优选实施例。根据上述讨论，还应当显而易见的是，在发展迅速且进一步的进展难以预见的此技术领域中，本领域技术人员可在安排和细节上对本发明进行修改，而不背离落在所附权利要求及其等价方案的范围内的本发明的原理。例如，方法的一个或多个操作可组合或进一步分开。

[0132] 可选实施例

[0133] 尽管已经描述了将本地执行矢量友好指令格式的实施例，但本发明的可选实施例可通过运行在执行不同指令集的处理器(例如，执行美国加利福亚州桑尼维尔的MIPS技术公司的MIPS指令集的处理器、执行加利福亚州桑尼维尔的ARM控股公司的ARM指令集的处理器)上的仿真层来执行矢量友好指令格式。同样，尽管附图中的流程图示出本发明的某些实施例的特定操作顺序，按应理解该顺序是示例性的(例如，可选实施例可按不同顺序执行操作、组合某些操作、使某些操作重叠等)。

[0134] 在以上描述中，为解释起见，阐明了众多具体细节以提供对本发明的实施例的透彻理解。然而，将对本领域技术人员明显的是，没有这些具体细节中的一些也可实践一个或多个其他实施例。提供所描述的具体实施例不是为了限制本发明而是为了说明本发明的实施例。本发明的范围不是由所提供的具体示例确定，而是仅由所附权利要求确定。

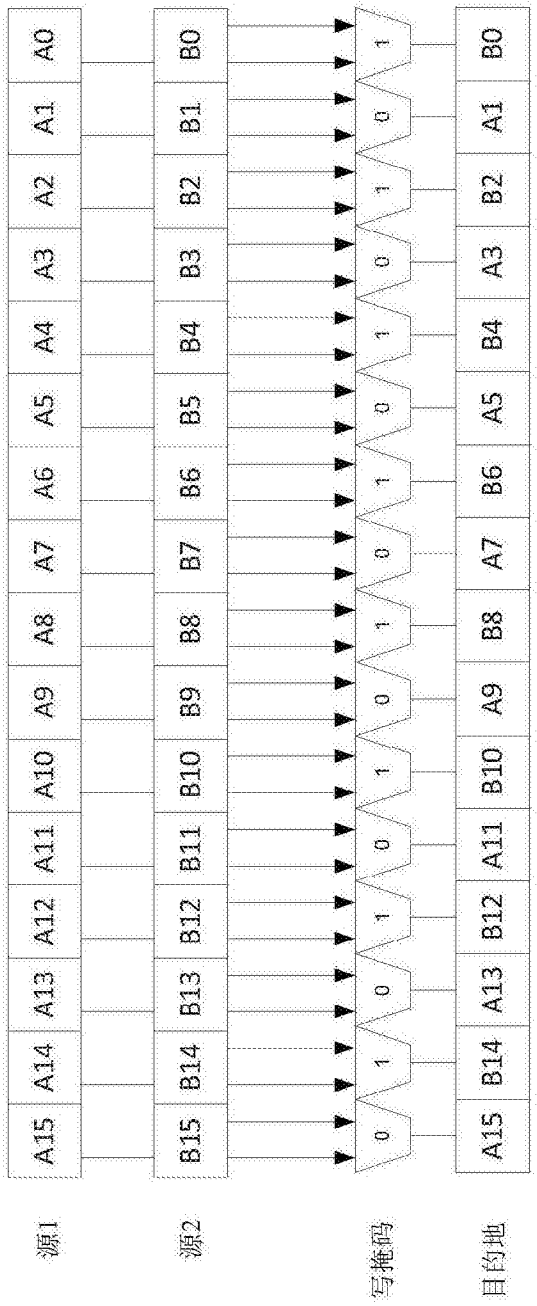

A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
A15 B14 A13 B12 A11 B10 A9 B8 A7 B6 A5 B4 A3 B2 A1 B0
源1
源2
写掩码
目的地

图1

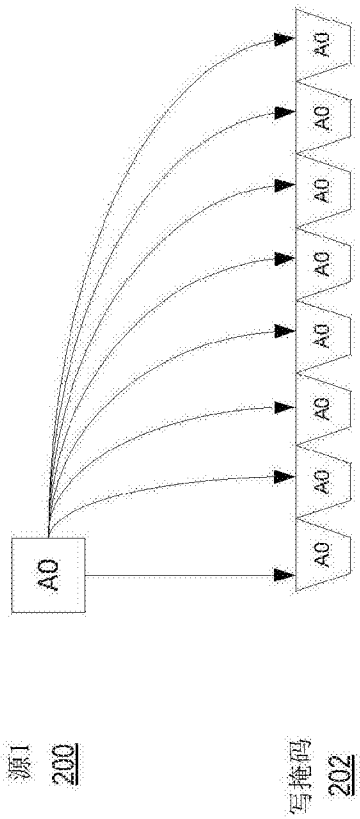

A0
A0
A0
A0
A0
A0
A0
A0
A0
源1
200
写掩码
202

图2A

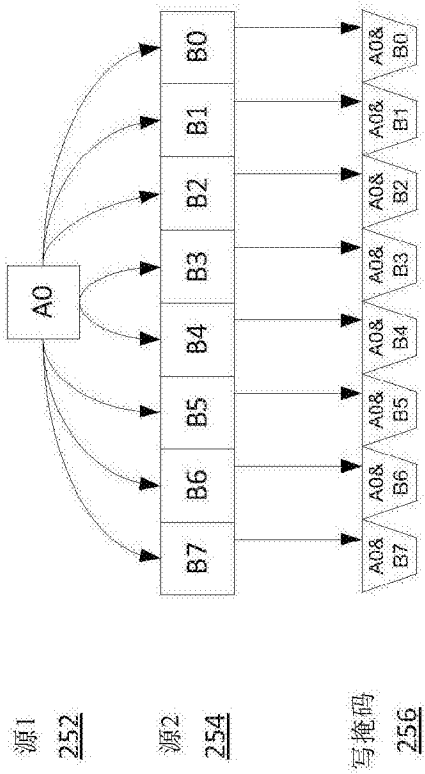

A0
B0
B1
B2
B3
B4
B5
B6
B7
A0&B0
A0&B1
A0&B2
A0&B3
A0&B4
A0&B5
A0&B6
A0&B7
源1
252
源2
254
写掩码
256

图2B

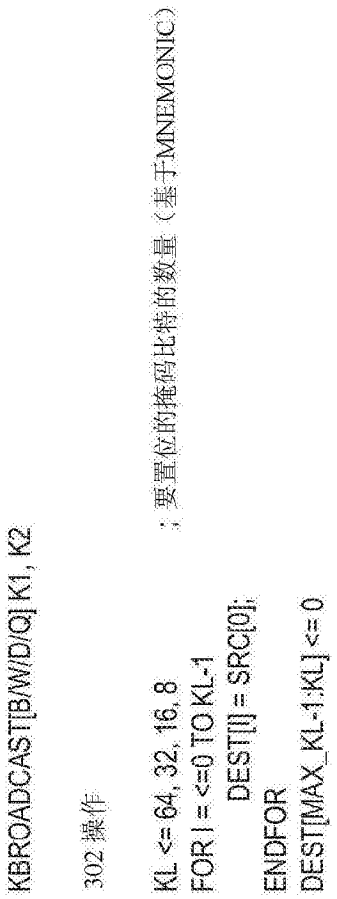

KBROADCAST[B/W/D/Q] K1, K2
302 操作
K1 <= 64, 32, 16, 8 ; 要置位的掩码比特的数量（基于MNEMONIC）
FOR I = <=0 TO K1-1
DEST[I] = SRC[0];
ENDFOR
DEST[MAX_K1-1:K1] <= 0

图3A

```
KBROADCAST[B/W/D/Q] K1, K2, K3

352 操作

KL <= 64, 32, 16, 8                ; 要置位的掩码比特的数量（基于MNEMONIC）
FOR I = <=0 TO KL-1
        DEST[I] = SRC2[0] AND SRC1[I];
ENDFOR
DEST[MAX_KL-1:KL] <= 0
```

图3B

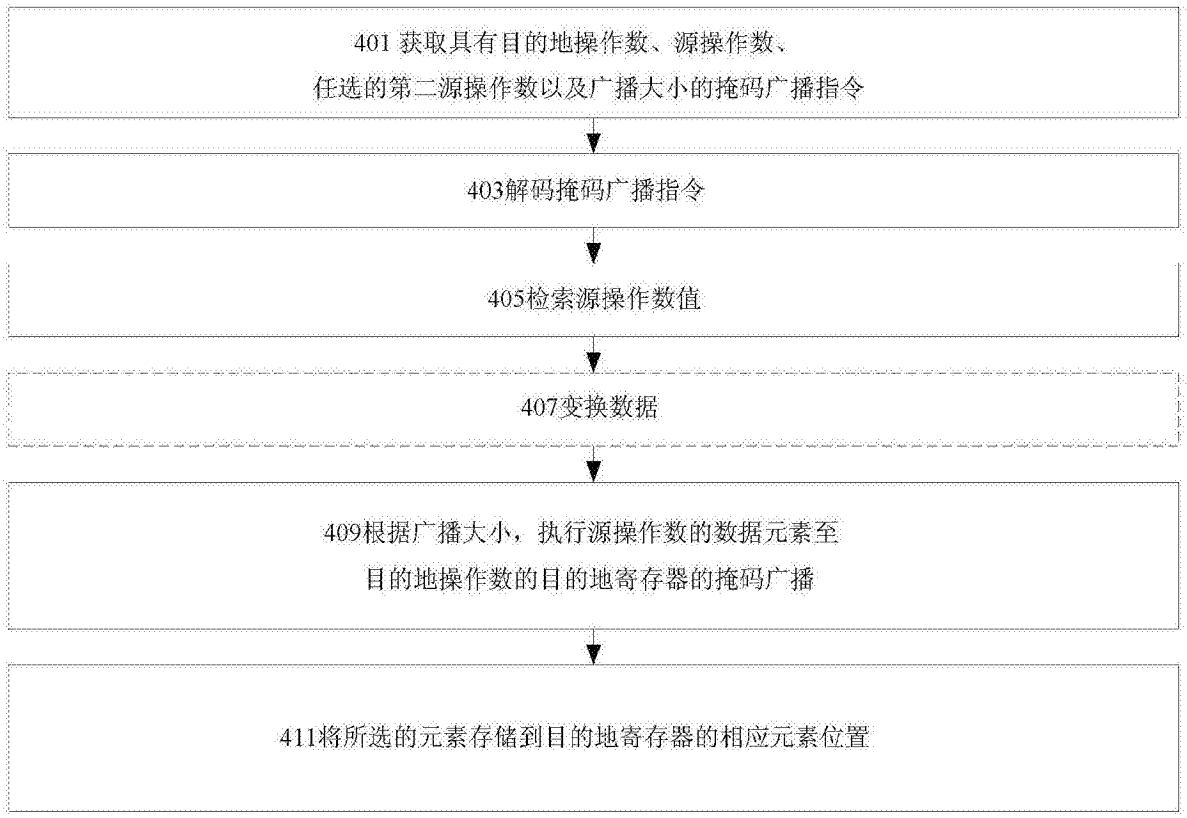

401 获取具有目的地操作数、源操作数、
任选的第二源操作数以及广播大小的掩码广播指令
403解码掩码广播指令
405检索源操作数值
407变换数据
409根据广播大小，执行源操作数的数据元素至
目的地操作数的目的地寄存器的掩码广播
411将所选的元素存储到目的地寄存器的相应元素位置

图4

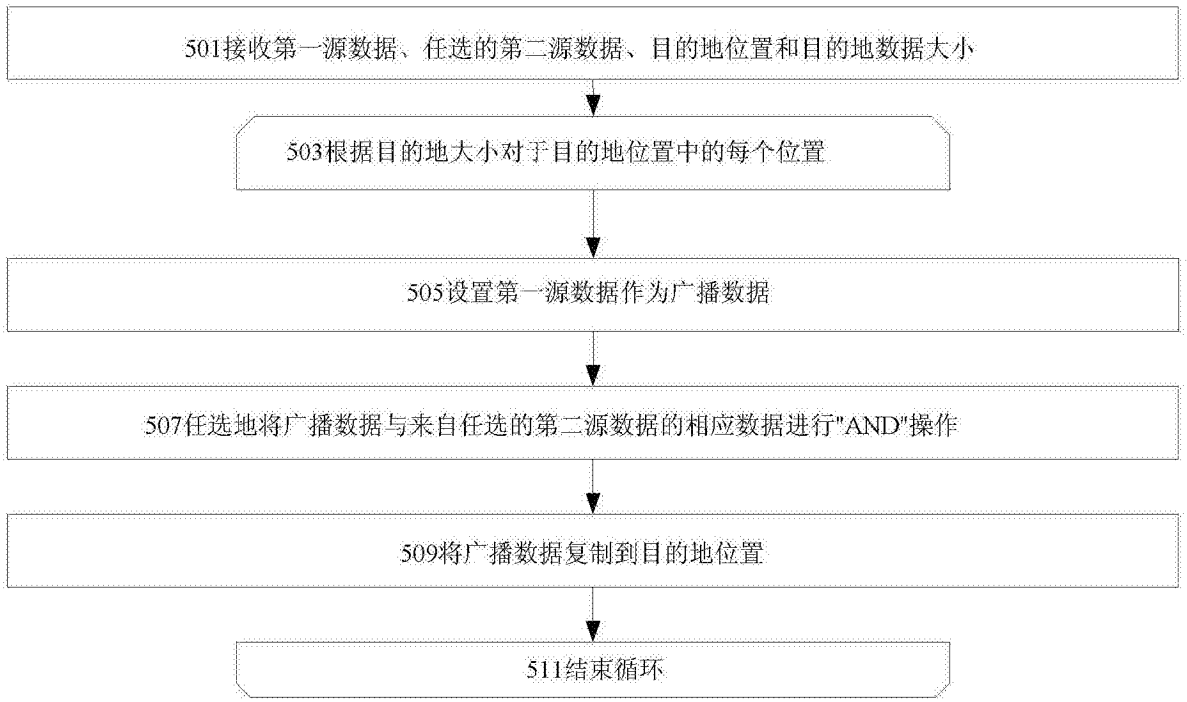

501接收第一源数据、任选的第二源数据、目的地位置和目的地数据大小
503根据目的地大小对于目的地位置中的每个位置
505设置第一源数据作为广播数据
507任选地将广播数据与来自任选的第二源数据的相应数据进行"AND"操作
509将广播数据复制到目的地位置
511结束循环

图5


601对于写掩码的每个目的地比特位置，
该比特位置的值需要两个源的组合吗？
否
是
603对于每个
目的地比特位置，
并行地存储来自
第 源的相应元素
605对于每个目的地比特位置，
并行地对来自源的相应值
进行"AND"操作并将结果
存储在目的地比特位置

图6

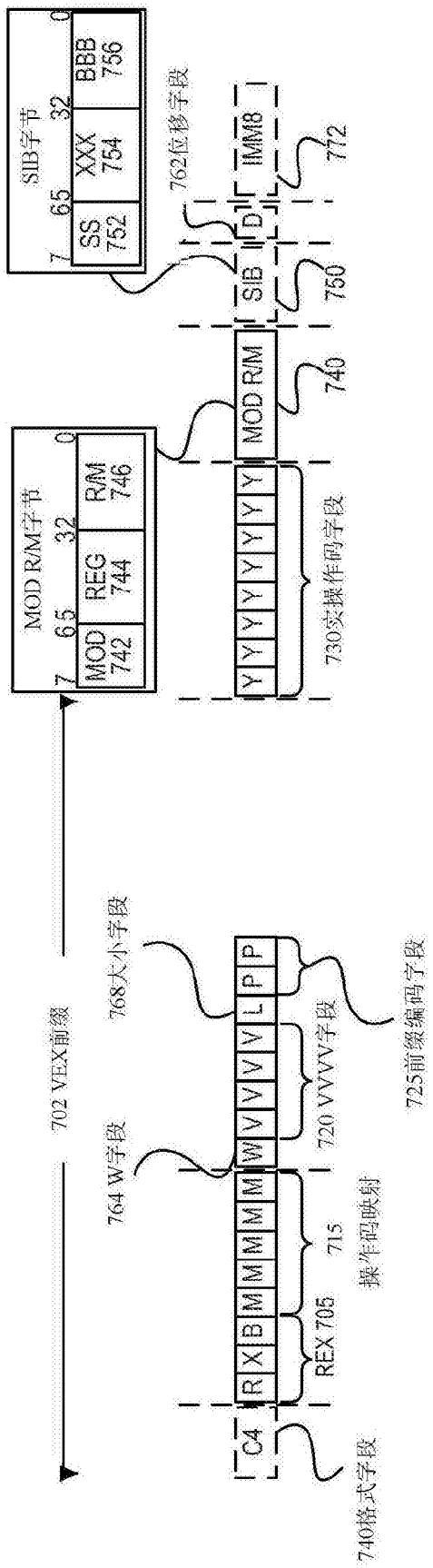

SIB字节
SS 752
XXX 754
BBB 756
762位移字段
MOD R/M字节
MOD 742
REG 744
R/M 746
SIB
D
IMM8
772
750
MOD R/M
740
730实操作码字段
702 VEX前缀
768大小字段
764 W字段
720 VVVV字段
725前缀编码字段
C4
R X B M M M M M W V V V V L P P
REX 705
715
操作码映射
740格式字段

图7A

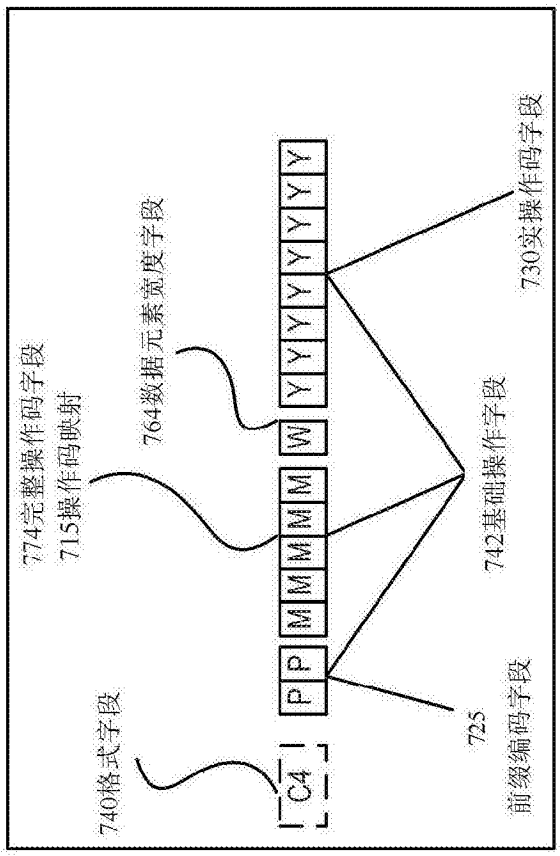

774完整操作码字段
715操作码映射
764数据元素宽度字段
740格式字段
C4
P
P
M
M
M
M
M
W
Y
Y
Y
Y
Y
Y
Y
Y
725
前缀编码字段
742基础操作字段
730实操作码字段

图7B

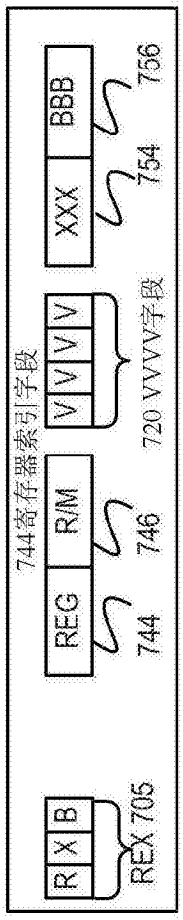

744寄存器索引字段
R
X
B
REX 705
REG
R/M
744
746
V
V
V
V
720 VVVV字段
XXX
BBB
754
756

图7C

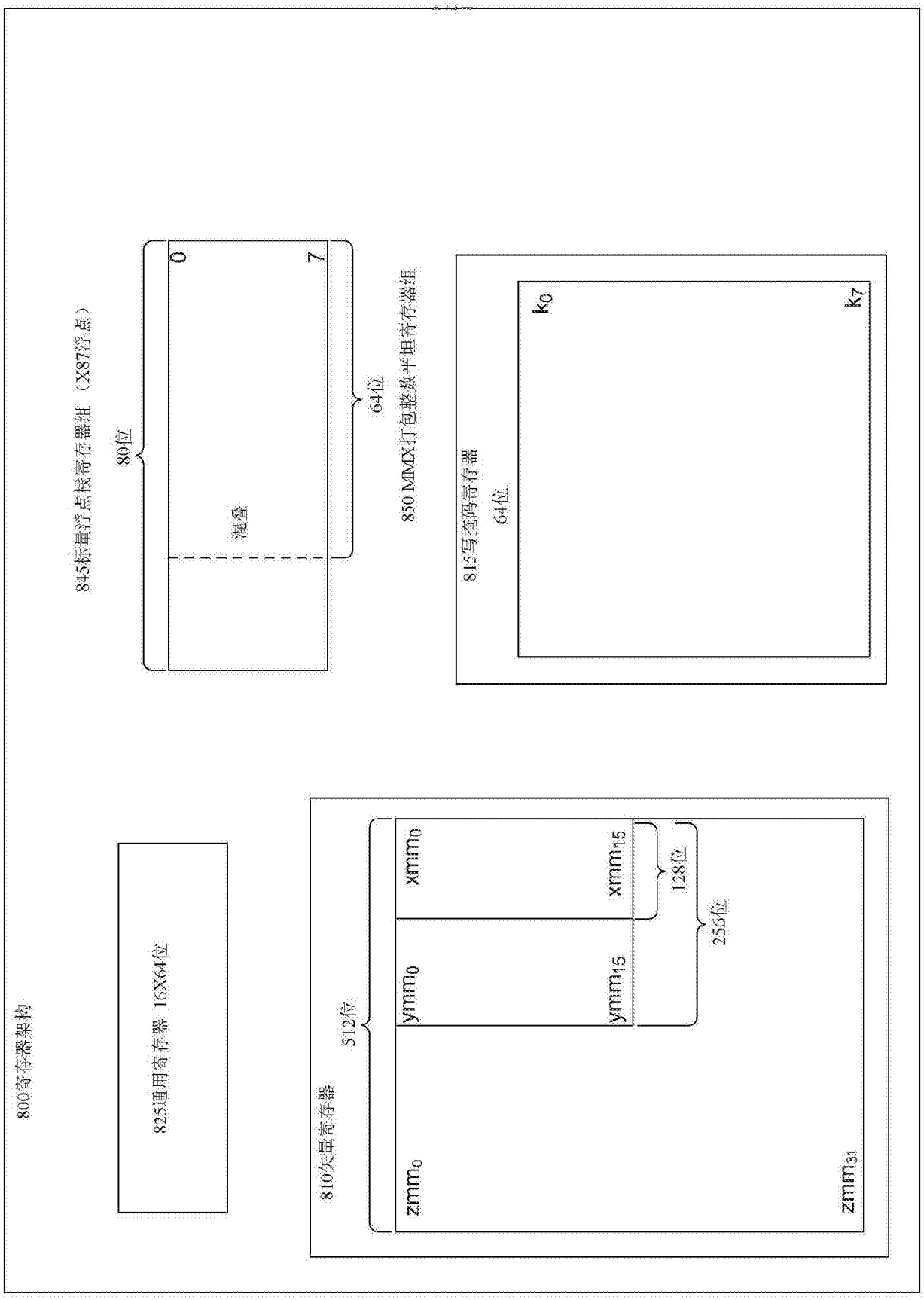

800寄存器架构
825通用寄存器 16X64位
845标量浮点栈寄存器组（X87浮点）
80位
0
7
混叠
64位
850 MMX打包整数平坦寄存器组
815写掩码寄存器
64位
k0
k7
810矢量寄存器
512位
zmm0
ymm0
xmm0
xmm15
ymm15
128位
256位
zmm31

图8

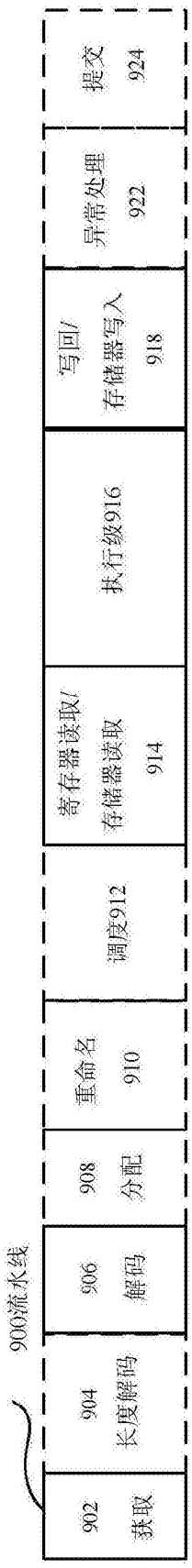

900流水线
902 获取
904 长度解码
906 解码
908 分配
重命名 910
调度912
寄存器读取/ 存储器读取 914
执行级916
写回/ 存储器写入 918
异常处理 922
提交 924

图9A

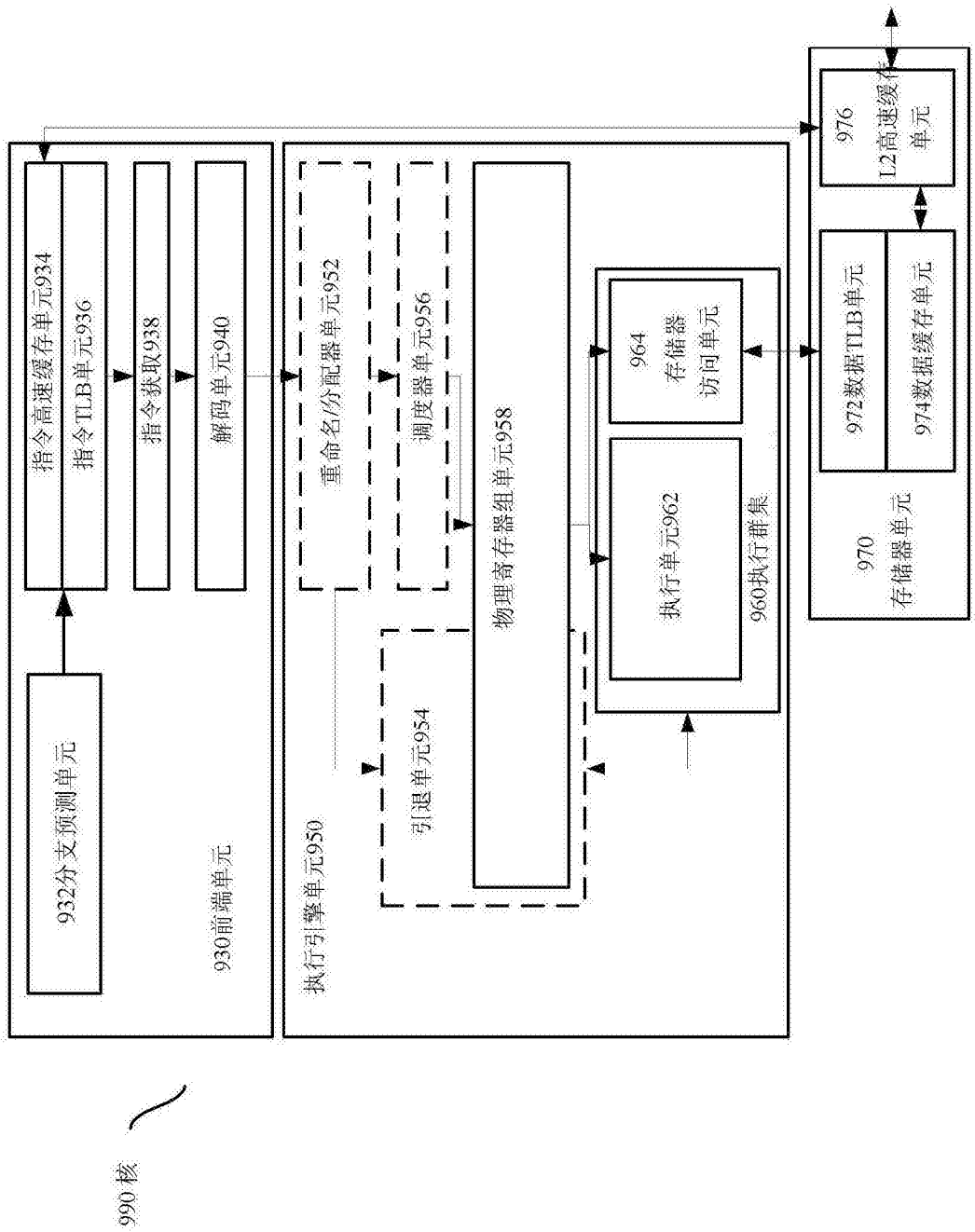

990 核
930前端单元
932分支预测单元
指令高速缓存单元934
指令TLB单元936
指令获取938
解码单元940
执行引擎单元950
重命名/分配器单元952
调度器单元956
物理寄存器组单元958
引退单元954
960执行群集
执行单元962
964
存储器
访问单元
970
存储器单元
972数据TLB单元
974数据缓存单元
976
L2高速缓存
单元

图9B

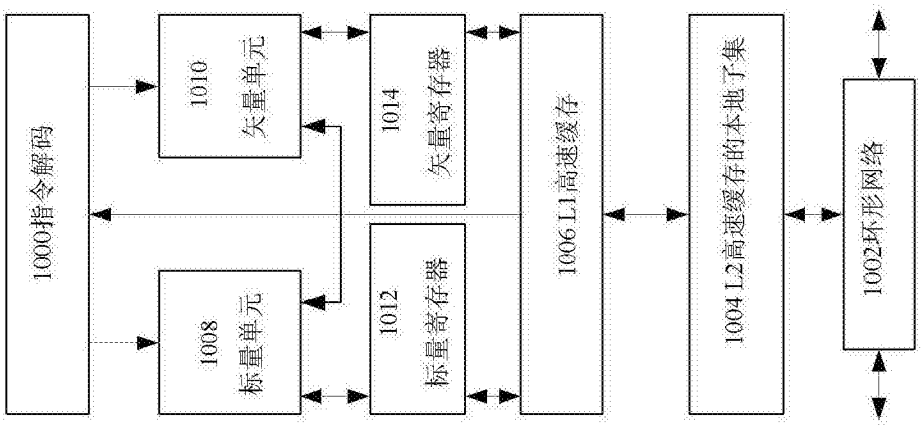

1000指令解码
1010
矢量单元
1008
标量单元
1014
矢量寄存器
1012
标量寄存器
1006 L1高速缓存
1004 L2高速缓存的本地子集
1002环形网络

图10A

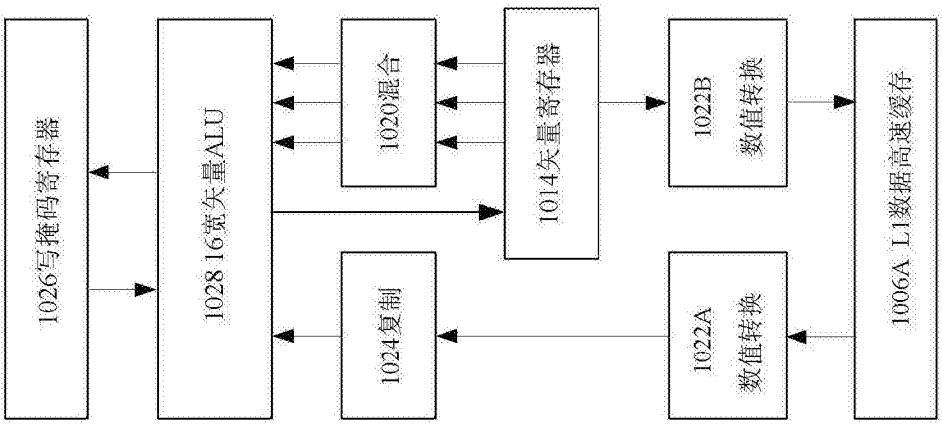

1026写掩码寄存器
1028 16宽矢量ALU
1020混合
1014矢量寄存器
1022B
数值转换
1006A L1数据高速缓存
1024复制
1022A
数值转换

图10B

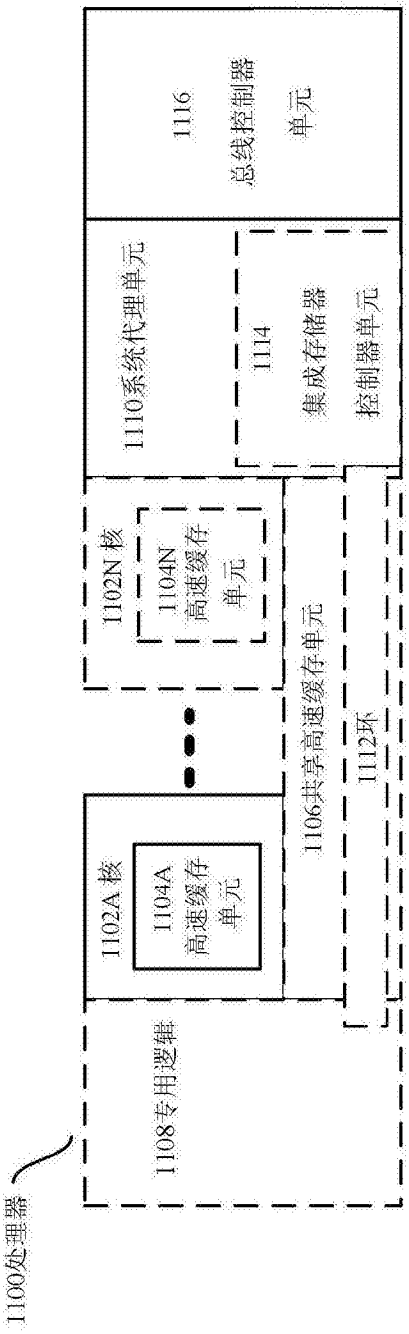

1100处理器
1108专用逻辑
1102A 核
1104A 高速缓存单元
1102N 核
1104N 高速缓存单元
1106共享高速缓存单元
1112环
1110系统代理单元
1114 集成存储器控制器单元
1116 总线控制器单元

图11

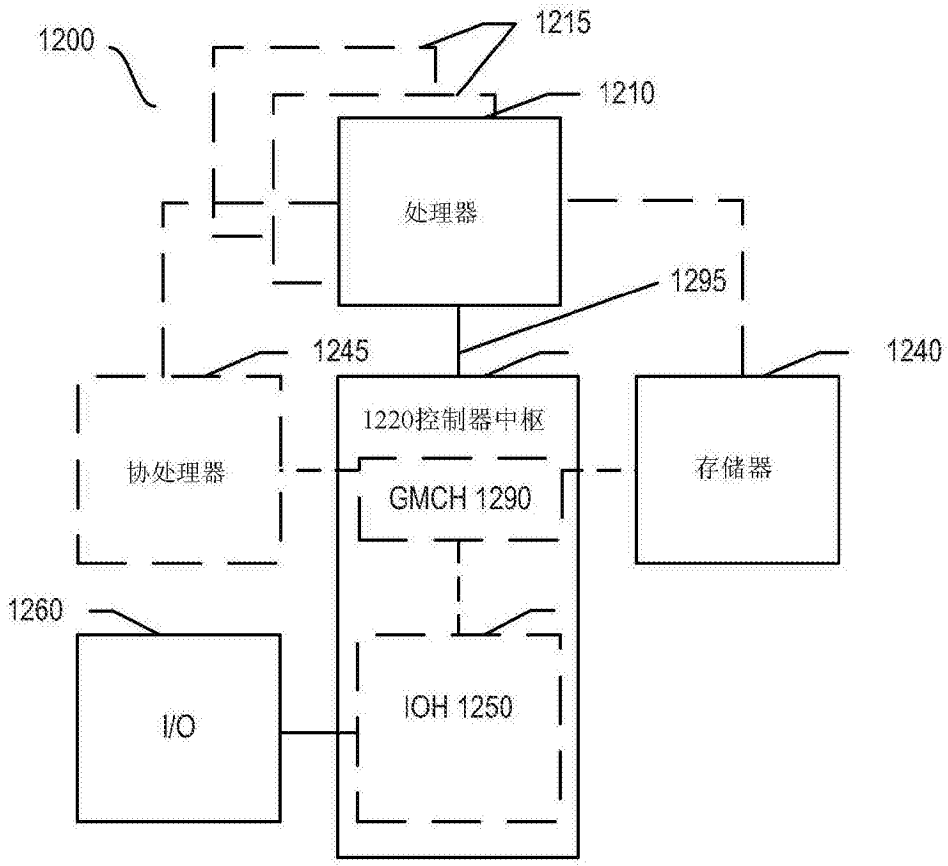

1200
1215
1210
处理器
1295
1245
1240
1220控制器中枢
协处理器
GMCH 1290
存储器
1260
I/O
IOH 1250

图12

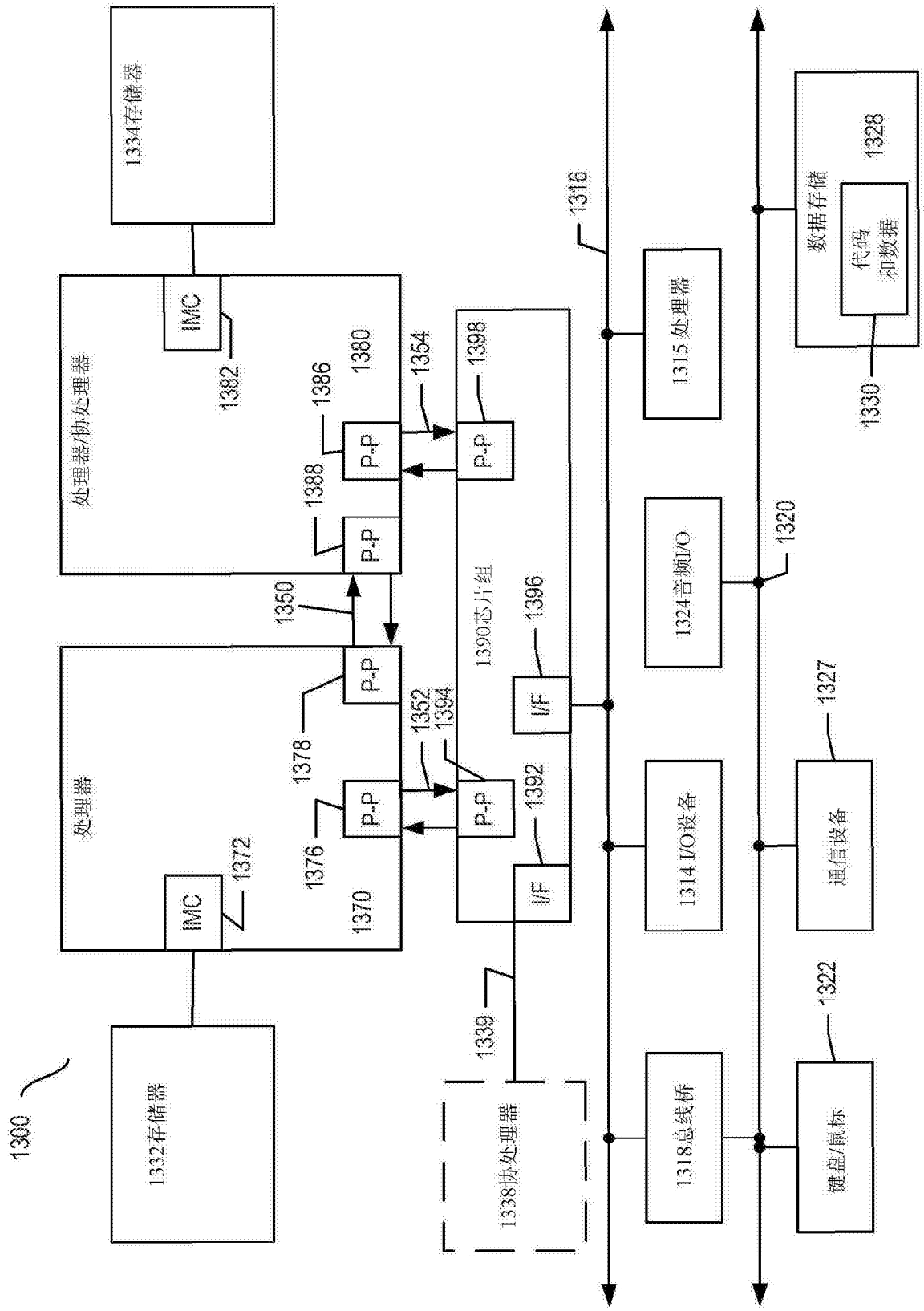

1300
1332存储器
1334存储器
处理器
处理器/协处理器
IMC
IMC
1372
1382
1370
1380
1376
1378
1386
1388
1350
1352
1354
1394
1398
1396
1392
P-P
P-P
P-P
P-P
P-P
P-P
1390芯片组
I/F
I/F
1339
1338协处理器
1316
1315处理器
1324音频I/O
1314 I/O设备
1318总线桥
1320
1327
通信设备
1322
键盘/鼠标
数据存储
1328
代码
和数据
1330

图13

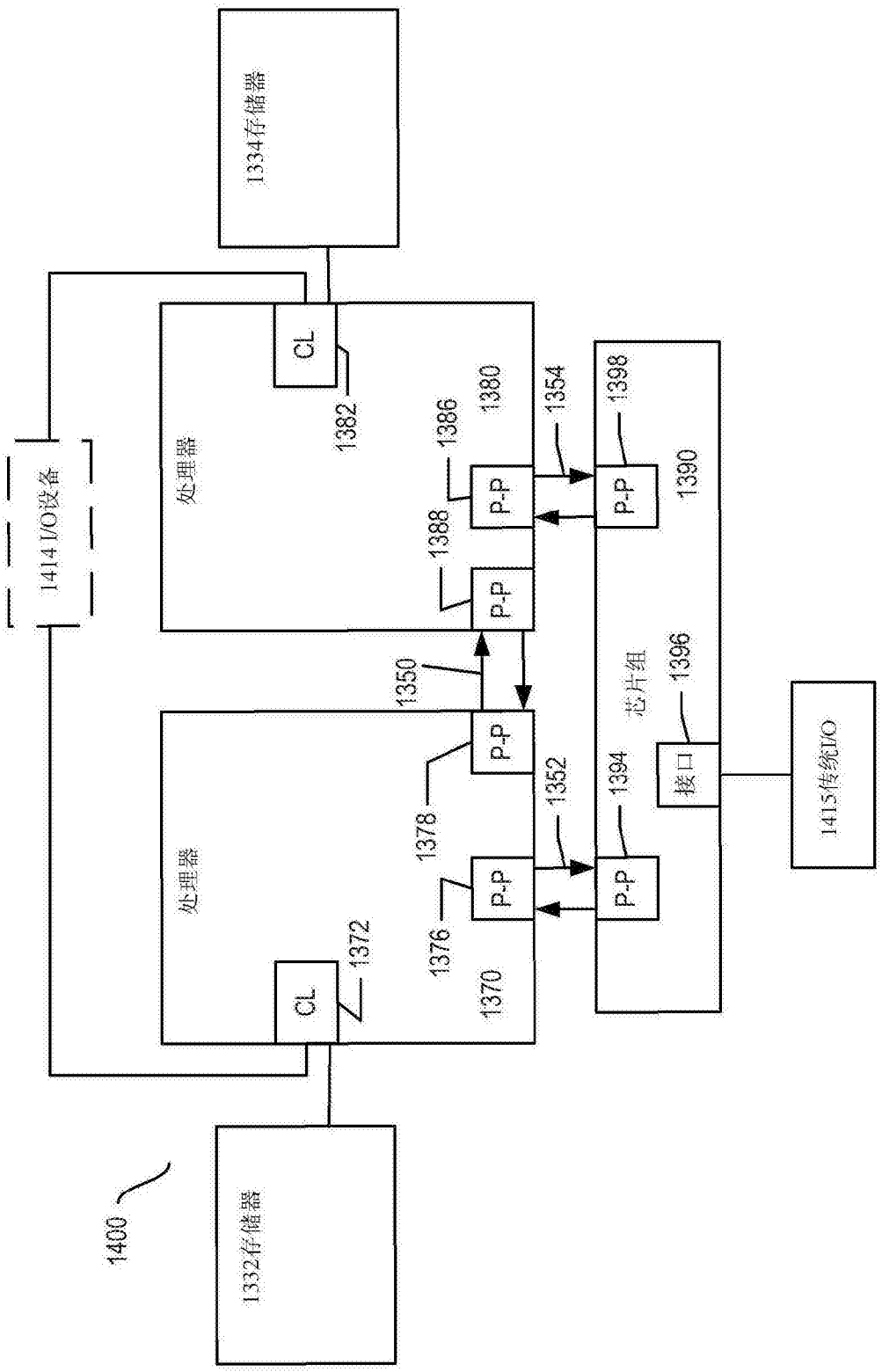

1334存储器
CL
1382
处理器
1386
1380
1354
1398
P-P
P-P
1390
1388
P-P
1414 I/O设备
1350
P-P
芯片组
1396
接口
1415传统I/O
1378
1352
1394
处理器
P-P
P-P
1376
1372
CL
1370
1400
1332存储器

图14

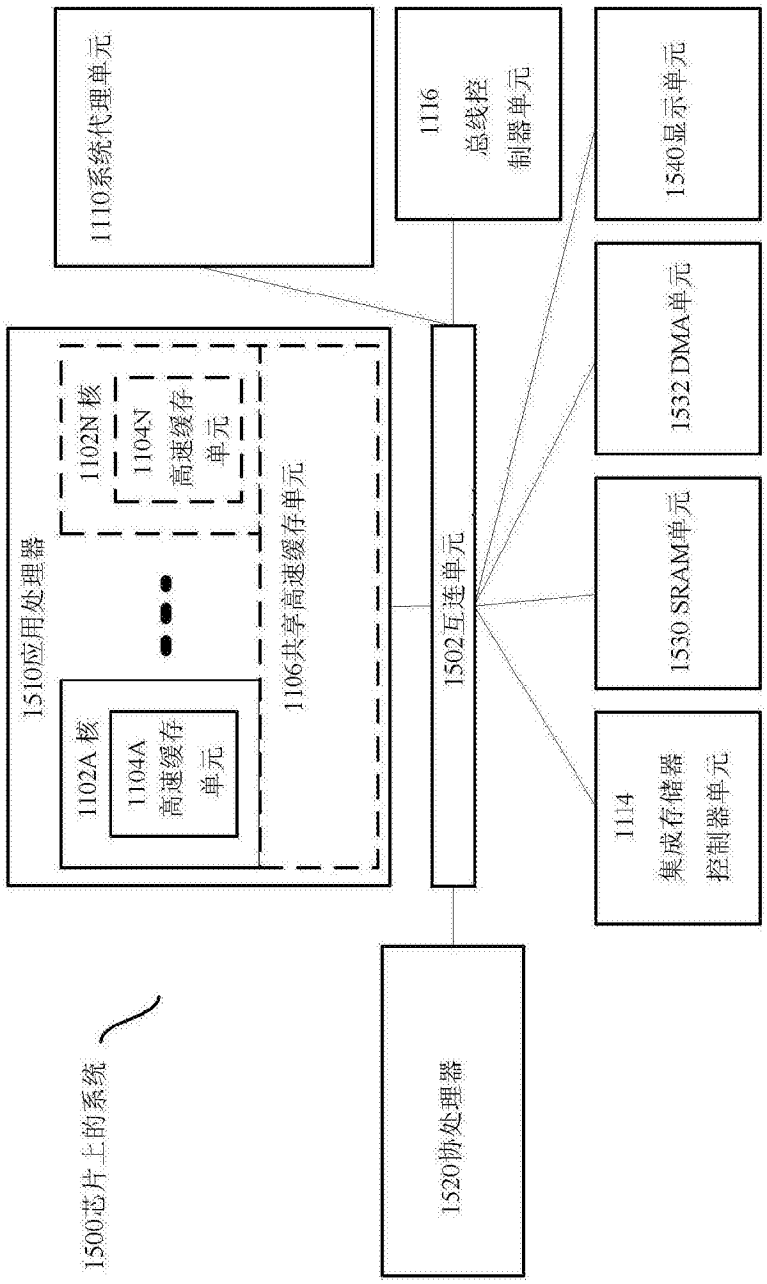

1110系统代理单元
1116
总线控
制器单元
1540显示单元
1532 DMA单元
1530 SRAM单元
1114
集成存储器
控制器单元
1502互连单元
1510应用处理器
1102N 核
1104N
高速缓存
单元
1102A 核
1104A
高速缓存
单元
1106共享高速缓存单元
1500芯片上的系统
1520协处理器

图15

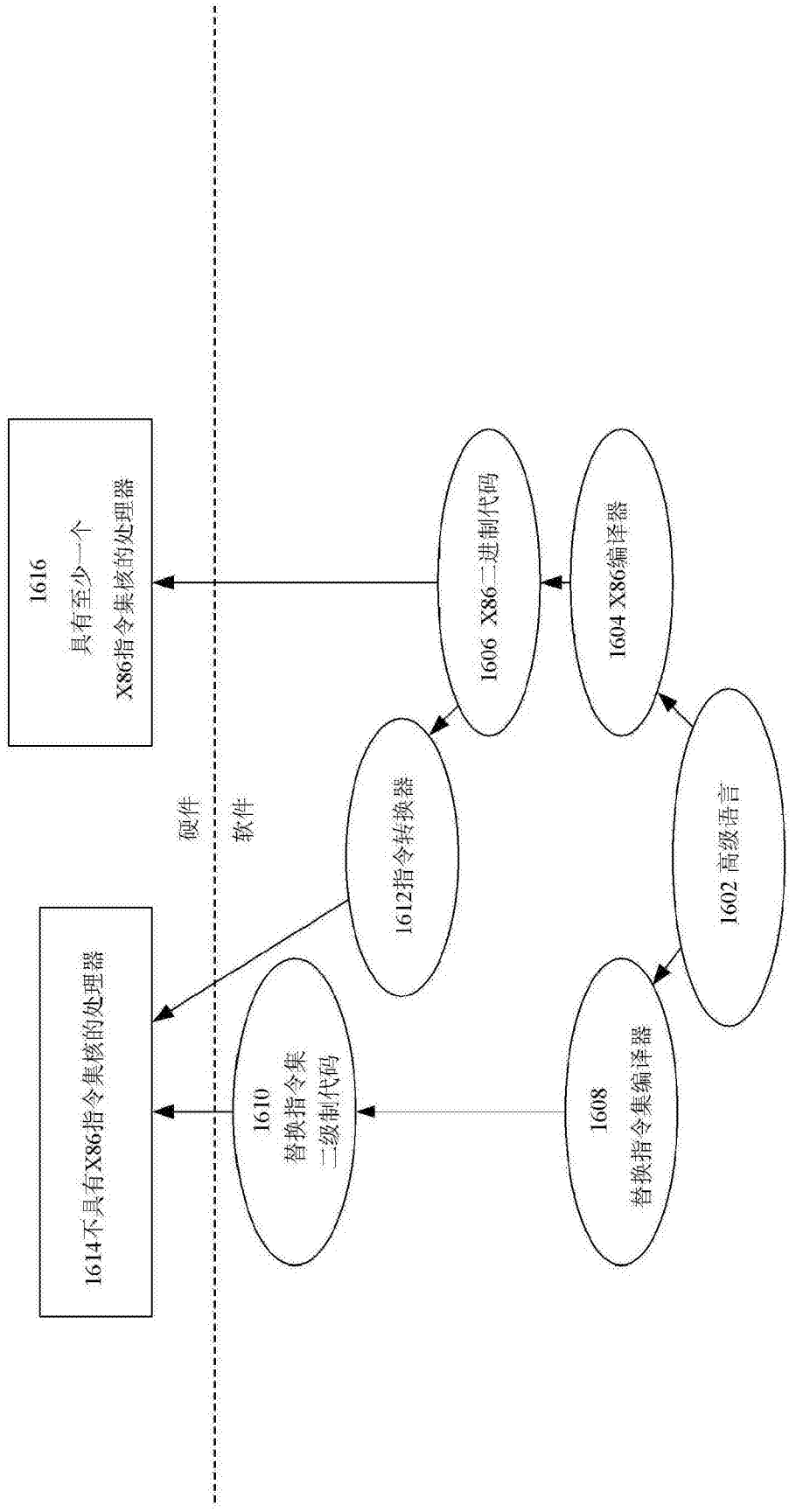

1616
具有至少一个
X86指令集核的处理器
1606 X86二进制代码
1604 X86编译器
硬件
软件
1612指令转换器
1602 高级语言
1614不具有X86指令集核的处理器
1610
替换指令集
二级制代码
1608
替换指令集编译器

图16