[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
G06F 9/30
G06F 9/302 G06F 9/318

[12] 发明专利说明书

[21] ZL 专利号 00810883.8

[45] 授权公告日 2004 年 8 月 4 日

[11] 授权公告号 CN 1160621C

[22] 申请日 2000.6.26 [21] 申请号 00810883.8
[30] 优先权
[32] 1999. 7.26 [33] US [31] 09/360,612
[86] 国际申请 PCT/US2000/017630 2000.6.26
[87] 国际公布 WO2001/008005 英 2001.2.1
[85] 进入国家阶段日期 2002.1.25
[71] 专利权人 英特尔公司
地址 美国加利福尼亚州
[72] 发明人 G·K·陈
审查员 袁文婷

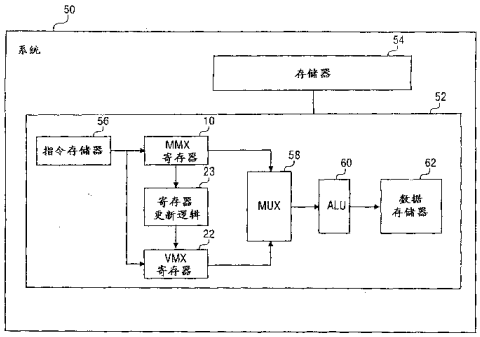
[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 吴立明 王忠忠

权利要求书 3 页 说明书 6 页 附图 6 页

[54] 发明名称 处理器以及由处理器为矩阵处理使用两组寄存器的方法

[57] 摘要

一个处理器具有至少两组寄存器。 第一组存储数据矩阵，第二组存储该数据矩阵的转置拷贝。当修改第一组的任何行的任何部分时，也自动修改在第二组中的转置拷贝的列的相应部分。 由处理器使用为矩阵处理的两组寄存器的方法包括存储数据矩阵到第一组寄存器，第一组寄存器具有第一数目的寄存器，每一寄存器包括第一数目的存储单元，每一存储单元存储矩阵的一个元素，和转置该数据矩阵到第二组寄存器，第二组寄存器具有第二数目的寄存器，每一寄存器包括第二数目的存储单元。 该方法还包括引用第一组寄存器中的一个寄存器来操作数据矩阵的一行和引用第二组寄存器中的一个寄存器来操作数据矩阵的一列。



ISSN1008-4274

1. 一种处理器，包括:

第一组存储数据矩阵的寄存器;

第二组存储数据矩阵的转置的寄存器; 和

修改第二组寄存器的寄存器更新逻辑，以便维护在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改; 以及在两组寄存器之间选择的多路复用逻辑。

2. 权利要求 1 所述处理器，其中，在第一组寄存器中的数据矩阵的一行的修改导致在第二组寄存器中的数据矩阵的转置的一列的相应修改。

3. 权利要求 1 所述处理器，其中，第一组寄存器包括第一数目的寄存器，每一寄存器包括第一数目的存储单元，第二组寄存器包括第二数目的寄存器，每一寄存器包括第二数目的存储单元，并且第二数目的存储单元大于或等于第一数目的寄存器。

4. 权利要求 3 所述处理器，其中，第一数目的寄存器等于第二数目的寄存器。

5. 权利要求 4 所述处理器，其中，第一组寄存器包括 MMX™ 寄存器，第一数目的寄存器是 8，数据矩阵包含图像数据。

6. 权利要求 1 所述处理器，其中，处理器执行引用第一组寄存器中的一个寄存器的指令来操作数据矩阵的一行和执行引用第二组寄存器中的一个寄存器的指令来操作数据矩阵的一列。

7. 一种由处理器为矩阵处理使用两组寄存器的方法，包括:

存储数据矩阵到第一组寄存器，该第一组寄存器包括第一数目的寄存器，每一寄存器包括第一数目的存储单元，每一存储单元存储矩阵的一个元素;

数据矩阵转置到第二组寄存器，该第二组寄存器包括第二数目的寄存器，每一寄存器包括第二数目的存储单元;

使用寄存器更新逻辑修改第二组寄存器，以便维护在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改;

引用第一组寄存器中的一个寄存器来操作数据矩阵的一行;

引用第二组寄存器中的一个寄存器来操作数据矩阵的一列;以及

使用多路复用逻辑在两组寄存器之间选择。

8. 权利要求 7 所述方法，另外包括修改在第一组寄存器中的数据矩阵的一行和修改在第二组寄存器中的数据矩阵的转置的相应列。

9. 权利要求 7 所述方法，另外包括对存储在第二组寄存器中的一个寄存器中的列数据执行变换操作。

10. 权利要求 9 所述方法，其中，执行变换操作包括对存储在第二组寄存器中的一个寄存器中的列数据执行离散余弦变换操作。

11. 权利要求 7 所述方法，其中，第二数目的存储单元大于或等于第一数目的寄存器。

12. 权利要求 7 所述方法，其中，第一数目的寄存器等于第二数目的寄存器。

13. 权利要求 7 所述方法，其中，第一组寄存器包括 MMX™ 寄存器，第一数目的寄存器是 8，数据矩阵包含图像数据。

14. 一种系统，包括:

存储器;

连接到存储器的处理器，该处理器包括:

第一组存储数据矩阵的寄存器;

第二组存储数据矩阵的转置的寄存器;和

修改第二组寄存器的寄存器更新逻辑，以便维持在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改;以及在两组寄存器之间选择的多路复用逻辑。

15. 权利要求 14 所述系统，其中，在第一组寄存器中的数据矩阵的一行的修改导致在第二组寄存器中的数据矩阵的转置的一列的相应修改。

16. 权利要求 14 所述系统，其中，第一组寄存器包括第一数目的寄存器，每一寄存器包括第一数目的存储单元，第二组寄存器包括第二数目的寄存器，每一寄存器包括第二数目的存储单元，并且第二数目的存储单元大于或等于第一数目的寄存器。

17. 权利要求 16 所述系统，其中，第一数目的寄存器等于第二

数目的寄存器。

18. 权利要求 17 所述系统，其中，第一组寄存器包括 MMX™ 寄存器，第一数目的寄存器是 8，数据矩阵包含图像数据。

19. 权利要求 14 所述系统，其中，处理器执行引用第一组寄存器中的一个寄存器的指令来操作数据矩阵的一行和执行引用第二组寄存器中的一个寄存器的指令来操作数据矩阵的一列。

20. 一种由处理器为图像数据矩阵的离散余弦变换（DCT）处理使用两组寄存器的方法，包括:

存储矩阵到第一组寄存器，该第一组寄存器包括第一数目的寄存器，每一寄存器包括第一数目的存储单元，每一存储单元存储矩阵的一个元素;

矩阵转置到第二组寄存器，该第二组寄存器包括第二数目的寄存器，每一寄存器包括第二数目的存储单元;

使用寄存器更新逻辑修改第二组寄存器，以便维持在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改;

使用多路复用逻辑在两组寄存器之间选择; 和

至少部分引用第二组寄存器中的一个寄存器执行离散余弦变换(DCT)处理来操作数据矩阵的一列。

21. 权利要求 20 所述方法，另外包括修改第一组寄存器中的数据矩阵的一行和修改第二组寄存器中的数据矩阵的转置的相应一列。

处理器以及由处理器为矩阵处理使用两组寄存器的方法

技术领域

本发明一般涉及计算机系统，具体说，涉及处理器结构。

背景技术

一些处理器设计为为多媒体操作提供它们的指令集结构（ISA）的扩展。例如，由 Pentium® II，Pentium® III，和 Celeron™ 处理器支持的 MMX™ 指令可从加里福尼亚州的 Santa Clara 市的因特尔公司购买，它可以实现对多媒体应用有用的各种功能，诸如数字信号处理，和音频视频处理。这些指令支持对多媒体和通信数据类型的“单指令多数据”（SIMD）操作。虽然这些指令的使用提供对预存在指令的组合的改进来执行一个给定的功能，以及个别 MMX™ 指令对某些类型的处理是有效的，但是对更快的多媒体处理仍然存在各种障碍。例如，基于块的图像的许多实现和视频处理算法（诸如联合图像专家组（JPEG）和运动图像专家组（MPEG）模式）导致存储在一组可作为为 MMX™ 指令的操作数访问的寄存器中存储的数据在矩阵数学运算期间被转置。

发明概述

寄存器中间的数据的转置担负相当大的开销，从而减少了对多媒体处理的总的处理器的通过量。因此，为避免或减小这些延迟的任何技术都是处理器技术中的有价值的进步。

根据本发明的一个方面，提供一种处理器，包括:

第一组存储数据矩阵的寄存器;

第二组存储数据矩阵的转置的寄存器；和

修改第二组寄存器的寄存器更新逻辑，以便维护在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改；以及在两组寄存器之间选择的多路复用逻辑。根据本发明的一个方面，提供一种由处理器为矩阵处理使用两组寄存器的方法，包括:

存储数据矩阵到第一组寄存器，该第一组寄存器包括第一数目的寄存器，每一寄存器包括第一数目的存储单元，每一存储单元存储矩

阵的一个元素;

数据矩阵转置到第二组寄存器，该第二组寄存器包括第二数目的寄存器，每一寄存器包括第二数目的存储单元;

使用寄存器更新逻辑修改第二组寄存器，以便维护在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改;

引用第一组寄存器中的一个寄存器来操作数据矩阵的一行;

引用第二组寄存器中的一个寄存器来操作数据矩阵的一列;以及

使用多路复用逻辑在两组寄存器之间选择。

根据本发明的一个方面，提供一种系统，包括:

存储器;

连接到存储器的处理器，该处理器包括:

第一组存储数据矩阵的寄存器;

第二组存储数据矩阵的转置的寄存器；和

修改第二组寄存器的寄存器更新逻辑，以便维持在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改；以及在两组寄存器之间选择的多路复用逻辑。

根据本发明的一个方面，提供一种由处理器为图像数据矩阵的离散余弦变换（DCT）处理使用两组寄存器的方法，包括:

存储矩阵到第一组寄存器，该第一组寄存器包括第一数目的寄存器，每一寄存器包括第一数目的存储单元，每一存储单元存储矩阵的一个元素;

矩阵转置到第二组寄存器，该第二组寄存器包括第二数目的寄存器，每一寄存器包括第二数目的存储单元;

使用寄存器更新修改第二组寄存器，以便维持在第一组寄存器中存储的数据矩阵以及在第二组寄存器中存储的数据矩阵的转置之间的转置关系，来响应第一组寄存器的修改;

使用多路复用逻辑在两组寄存器之间选择；和

至少部分引用第二组寄存器中的一个寄存器执行离散余弦变换(DCT)处理来操作数据矩阵的一列。

本发明的一个实施例是一个处理器，它具有存储数据矩阵的第一组寄存器，和连接到第一组的第二组寄存器，第二组寄存器存储数据矩阵的转置拷贝。

本发明的另一个实施例是一种由处理器为矩阵处理使用两组寄存器的方法。该方法包括在第一组寄存器中存储数据矩阵，第一组寄存器具有第一数目的寄存器，每一寄存器包括第一数目的存储单元，每一存储单元存储矩阵的一个元素，和数据矩阵转置到第二组寄存器，该第二组寄存器具有第二数目的寄存器，每一寄存器包括第二数目的存储单元。该方法还包括引用第一组寄存器中的一个来操作数据矩阵的一行，和引用第二组寄存器中的一个来操作数据矩阵的一列。

附图说明

从下面对本发明的详细说明可以显见本发明的特征和优点，其中，

图 1 是一组根据现有技术的 MMX™ 寄存器的示意图；

图 2 是存储图像数据的 8 象素乘 8 象素块的该组 MMX™ 寄存器的示意图；

图 3 是存储图像数据转置的 8 象素乘 8 象素块的该组 MMX™ 寄存器的示意图；

图 4 是根据本发明的一个实施例连接到 MMX™ 寄存器组的一个虚拟 MMX™ 寄存器组的示意图；

图 5 是存储图像数据转置的 8 象素乘 8 象素块的虚拟 MMX™ 寄存器组的示意图；

图 6 是一个系统的示意图，该系统具有一个处理器，它具有按照本发明的一个 MMX™ 寄存器组和一个虚拟 MMX™ 寄存器组的示意图。

具体实施方式

本发明的一个实施例包括一种为扩展 MMX™ 寄存器为用于 2 维（2-D）矩阵运算更加有效的方法和装置。

说明书中对本发明的“一个实施例”的参考意味着结合该实施例说明的一种特定的特征、结构或特性至少被包含在本发明的一个实施例中。这样，在本说明书中不同地方出现的短语“在一个实施例中”不一定所有都指同一个实施例。

当执行一个指令时，一个处理器通常引用一个或多个寄存器操作数。对于 MMX™ 指令，寄存器操作数可以是称为 MMX™ 寄存器的一组或

多组特殊的寄存器。图 1 是一组根据现有技术的 MMX™ 寄存器的示意图。在图 1 所示的寄存器组 10 中，有 8 个 MMX™ 寄存器，标以 mm0 12 到 mm7 14。在其它实施例中，寄存器的数目可以多于或少于 8 个。每一寄存器包括多个数据单元，从所示低单元 16 到高单元 18 排序。在一个实施例中，一个单元包括一个字节。在其它实施例中，一个单元可以包括一个字、一个双字、或其它存储单元。在至少一个已知系统中，每个 MMX™ 寄存器的单元数目（例如字节）是 8，虽然在其它系统中可以使用其它单元数目。为使用 MMX™ 寄存器有效实现 SIMD 多媒体处理，要被处理的数据应该这样排列，使得多个相关的数据项排列在单一 MMX™ 寄存器中。例如，假定一个 8 象素乘 8 象素的图像数据块在 MMX™ 寄存器中如图 2 所示排列，每一象素值 P（i, j）表示一个单元，总寄存器组表示一个矩阵。8 象素乘 8 象素块可以是一个较大图像的一部分。在该例中，该图像数据块的第一行存储在第一 MMX™ 寄存器 mm0 12 中，第一行的第一列存储在 mm0 的低单元中，第一行的最后一列存储在 mm0 的高单元中，图像数据的第二行存储在第二 MMX™ 寄存器 mm1 20 中，该第二行的第一列存储在 mm1 的低单元中，第二行的最后一列存储在 mm1 的高单元中，等等。

一旦数据如所示存储在 MMX™ 寄存器中，则处理器可以执行指令，每次一行有效操作该 8 X 8 矩阵。这一类型处理通常例如用于基于块的图像应用和其它应用中。例如，行 0 的所有数据可以使用单一 MMX™ 指令加到行 3 的数据上，如下所示。

PADDB MM0, MM3; 把行 0 加到行 3 上并把结果存储到行 0 中。

然而，为一次一列操作 8 X 8 矩阵会出现问题，因为每一列的数据分布在 8 个 MMX™ 寄存器中间。例如，第一列的数据分别分布在 mm0 12 到 mm7 14 的低单元中间，最后一列数据分布在 mm0 到 mm7 的高单元中间。为继续增加使用 MMX™ SIMD 处理的好处，必须如图 3 所示转置 8 X 8 矩阵。矩阵转置在数学中公知。在转置后，第一 MMX™ 寄存器 mm0 12 存储数据原来的第一列，原来第一列的第一行存储在低单元 16 中，原来第一列的最后一列存储在高单元 18 中，如图所示。相似地，其它 MMX™ 寄存器如所示存储 8 X 8 矩阵的列。

虽然 8 X 8 矩阵的转置可以使用公知的打包和解包指令执行，但是这一处理效率很低并且引起相当高的处理开销。一般，对于一个 8 X

8 矩阵的转置处理通过执行 4 X 4 矩阵的转置实现，每一转置对 Pentium® III 处理器需要至少 12 个处理周期。这样，仅为转置该数据至少使用 64 个周期，以便可以使用一个 MMX™ 指令来操作一个给定列的 8 个元素。

本发明的实施例通过在处理器中提供另一组寄存器帮助多媒体处理通过一系列打包和解包指令克服了转置 MMX™ 寄存器中数据的需要。在本发明的实施例中，在处理器结构中设计一个等价的“镜像” MMX™ 寄存器组。该寄存器组可以叫作虚拟 MMX™ 寄存器组，或 VMX 组。图 4 是根据本发明的一个实施例连接到 MMX™ 寄存器的虚拟 MMX™ 寄存器组的示意图。在一个实施例中，在 MMX™ 寄存器组 10 中的 MMX™ 寄存器的数目和在 VMX 寄存器组 22 中提供的 VMX 寄存器的数目相同。另外，在一个实施例中，VMX 寄存器的数目大于或等于在每一寄存器中的单元数目（取决于实现，单元是字节，字，双字，或其它存储单元）。VMX 寄存器组 22 存储来自 MMX™ 寄存器组中的转置的矩阵数据并当 MMX™ 寄存器组中的任何寄存器的任何单元被更新时通过寄存器更新逻辑 23 可以自动地更新。因此，MMX™ 寄存器组 10 的一行的加载自动导致 VMX 寄存器组 22 的一列加载。例如，来自第一 MMX™ 寄存器 mm0 12 的数据可以自动地存储在 VMX 寄存器 VM0 到 VM7 的低单元中，来自第二 MMX™ 寄存器 mm1 20 的数据可以自动地存储在 VMX 寄存器的次最低单元中，等等。

回过来看图 2，如果 MMX™ 寄存器用一个用图示 P0，0 到 P7，7 指示的 8 X 8 矩阵加载的话，则 VMX 寄存器可以由寄存器更新逻辑用转置的矩阵自动加载，如图 5 所示。为操作该矩阵的行元素，一个程序可以简单地引用一个或多个 MMX™ 寄存器 mm0 12 到 nm7 14。然而，为操作该矩阵的列元素，一个程序可以代之以引用 VMX 寄存器 vm0 24 到 vm7 26。因为 MMX™ 寄存器在处理器硬件中用 VMX 寄存器镜像，因此没有一致性问题。所有 MMX™ 指令都可以用任何一个寄存器组通过使用对所需要的适合的寄存器的引用来操作。不需对处理器的指令集进行改变，只有操作数引用可能在程序中改变。

本发明的实施例对现有处理器结构的一个优点是本发明为矩阵操作提供更大的平行性。这通过执行多个打包和解包指令避免了为列操作进行的费用大的转置而实现。

怎样使用本发明的一个例子是在许多视频处理方案中为处理 8 象素乘 8 象素块使用的离散余弦变换（DCT）。当前，为执行 8 X 8 DCT 时，处理包括首先执行一个 1 X 8 的列变换，转置 8 X 8 矩阵，执行另一个 1 X 8 列变换，然后再次转置该结果以得到 DCT 系数。当前在 Pentium®类处理器上运行的最优化的 DCT 需要大约 300 个周期的处理。在这一数量中，大约 100 个周期用于执行为 8 X 8 矩阵的转置操作。这样，本发明的实现为 DCT 处理产生大约 30%的改善。对逆 DCT 可以实现相似的性能增益。虽然这里讨论的是 DCT 的例子，但是本发明的实施例可以对任何矩阵操作都有用，包括在各种图像和视频压缩算法中使用的那些。

虽然上面在 2 维（2-D）矩阵的意义上讨论本发明，但是该概念可以适用于 3 维或更多维。例如，可以在处理器的设计中包括一个第三寄存器组来存储矩阵数据的其它转置。

图 6 是包括具有按照本发明的实施例的 MMX™ 寄存器组和虚拟 MMX™ 寄存器组的处理器的系统的示意图。系统 50 可以包括连接到存储器 54 的处理器 52。处理器 52 包括在该技术中公知的各种部件，为清楚起见其中许多在图 6 中省略。指令存储器 56 存储可以引用一个或者多个 MMX™ 寄存器 10 或一个或多个 VMX 寄存器 22 的指令。当 MMX™ 寄存器改变时寄存器更新逻辑 23 协调自动更新 VMX 寄存器 22。多路转接器（MUX）58 从 MMX™ 寄存器或 VMX 寄存器中的一个为给算术逻辑单元（ALU）60 的输入选择数据。ALU 产生为数据存储器 62 的数据。

本发明允许以更直觉的方式操作 MMX™ 寄存器。通过增加镜像寄存器组，可以简化任何基于块的算法的实现，并且它们的性能增加。可以从本发明得到好处的一些应用例子包括用于视频压缩算法的离散余弦变换（DCT），3 维（3-D）图形算法中的矩阵变换，及其它。

虽然本发明参考图示实施例说明，但是该说明并不打算被解释为限制的意义。对熟悉本发明所属技术领域的人明显的对图示实施例以及本发明的其它实施例的各种修改都被视为在本发明的精神和范围之内。

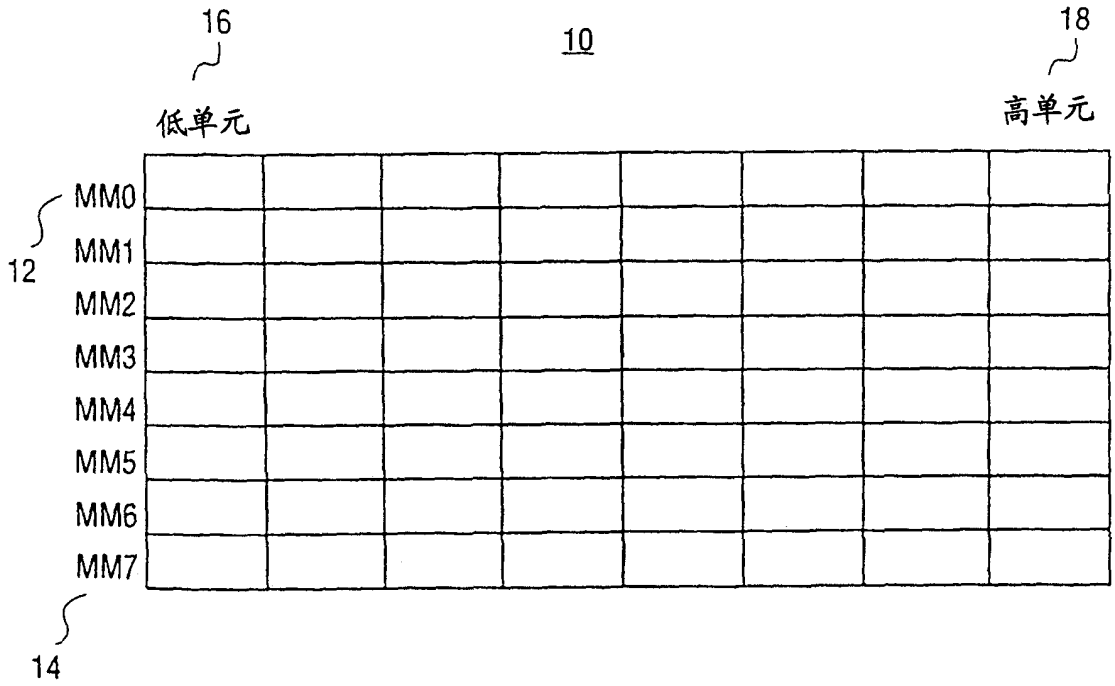


图 1(现有技术)

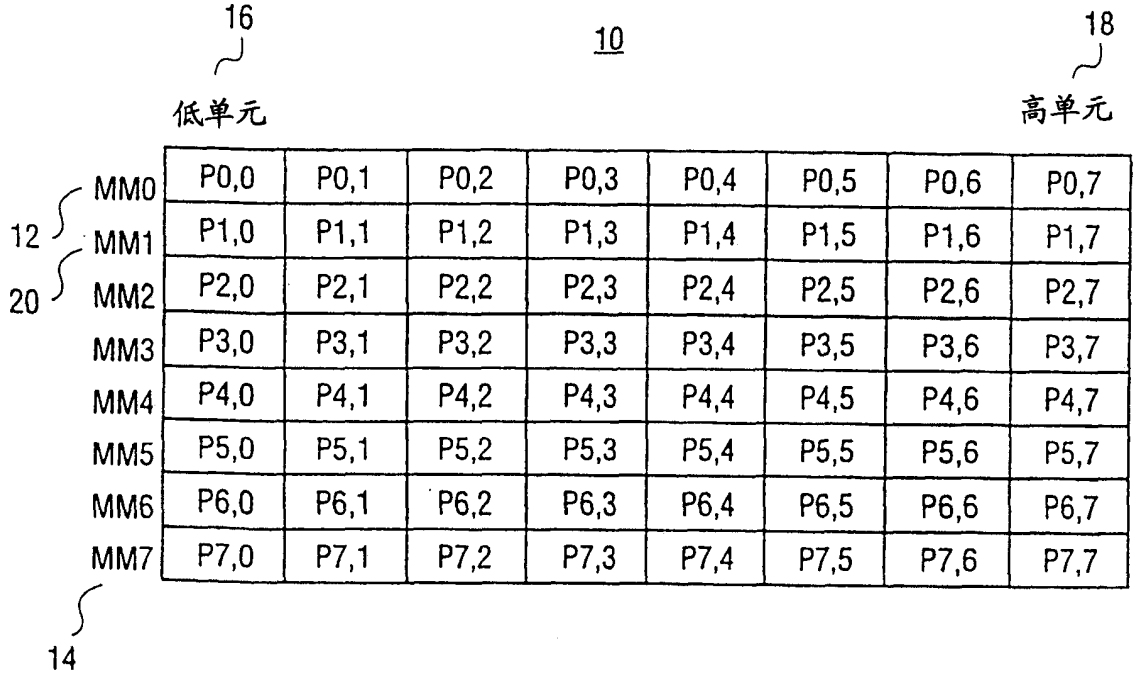

16
10
18
低单元
高单元
12
20
14
MM0
MM1
MM2
MM3
MM4
MM5
MM6
MM7
P0,0 P0,1 P0,2 P0,3 P0,4 P0,5 P0,6 P0,7
P1,0 P1,1 P1,2 P1,3 P1,4 P1,5 P1,6 P1,7
P2,0 P2,1 P2,2 P2,3 P2,4 P2,5 P2,6 P2,7
P3,0 P3,1 P3,2 P3,3 P3,4 P3,5 P3,6 P3,7
P4,0 P4,1 P4,2 P4,3 P4,4 P4,5 P4,6 P4,7
P5,0 P5,1 P5,2 P5,3 P5,4 P5,5 P5,6 P5,7
P6,0 P6,1 P6,2 P6,3 P6,4 P6,5 P6,6 P6,7
P7,0 P7,1 P7,2 P7,3 P7,4 P7,5 P7,6 P7,7

图 2

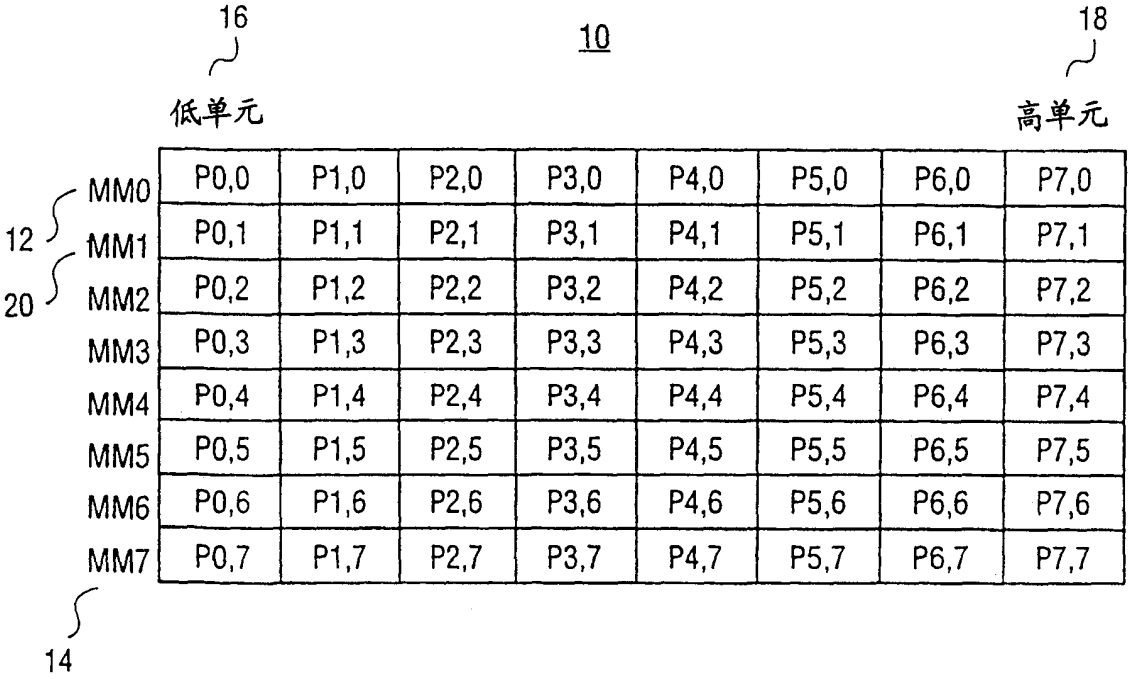

16
10
18
低单元
高单元
12
20
14
MM0
MM1
MM2
MM3
MM4
MM5
MM6
MM7
P0,0 P1,0 P2,0 P3,0 P4,0 P5,0 P6,0 P7,0
P0,1 P1,1 P2,1 P3,1 P4,1 P5,1 P6,1 P7,1
P0,2 P1,2 P2,2 P3,2 P4,2 P5,2 P6,2 P7,2
P0,3 P1,3 P2,3 P3,3 P4,3 P5,3 P6,3 P7,3
P0,4 P1,4 P2,4 P3,4 P4,4 P5,4 P6,4 P7,4
P0,5 P1,5 P2,5 P3,5 P4,5 P5,5 P6,5 P7,5
P0,6 P1,6 P2,6 P3,6 P4,6 P5,6 P6,6 P7,6
P0,7 P1,7 P2,7 P3,7 P4,7 P5,7 P6,7 P7,7

图 3

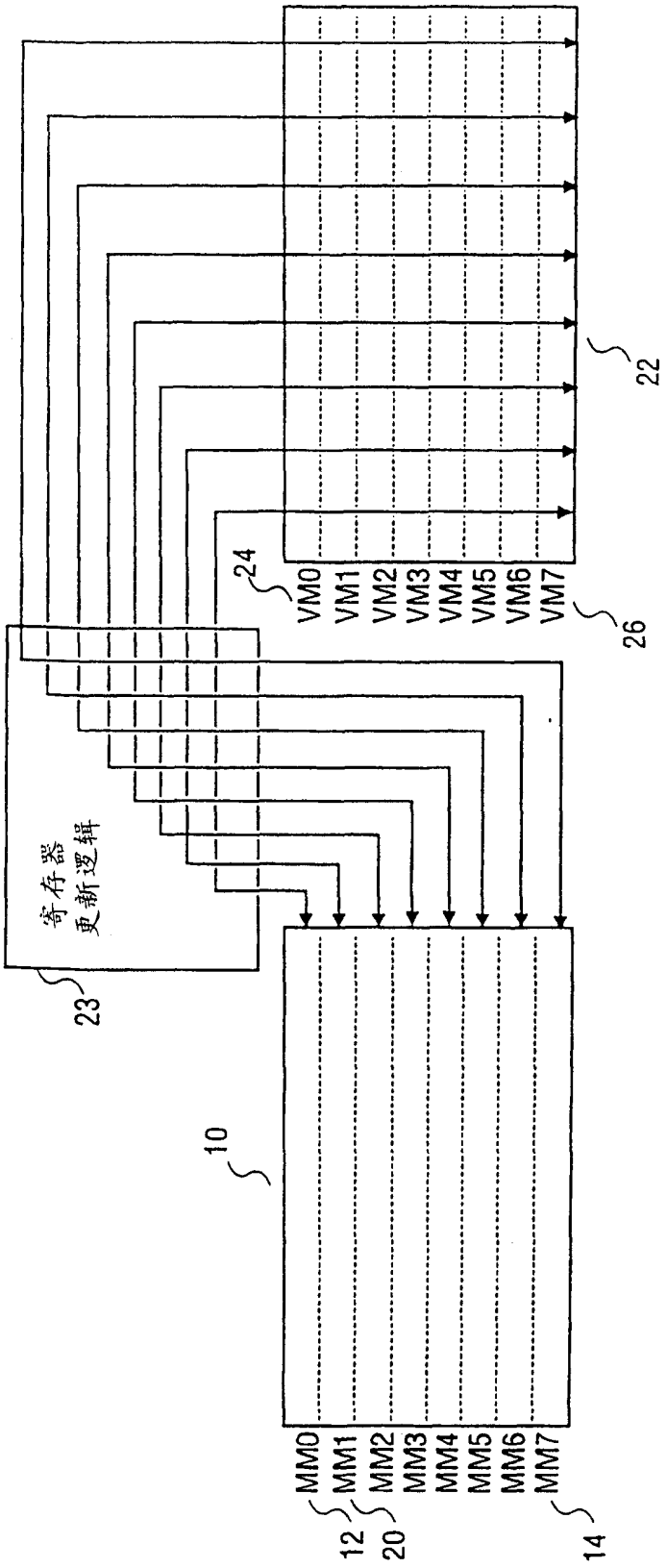

寄存器
更新逻辑
23
10
12
20
14
22
24
26
VM0
VM1
VM2
VM3
VM4
VM5
VM6
VM7
MM0
MM1
MM2
MM3
MM4
MM5
MM6
MM7

图 4

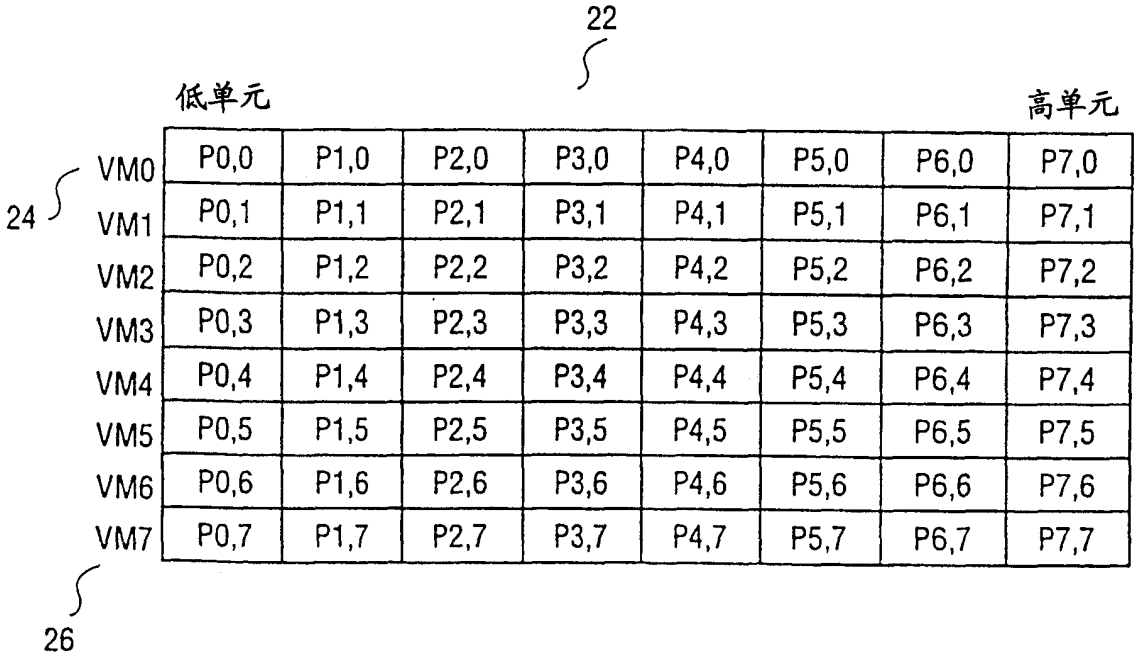

22
低单元
高单元
24
26
VM0
VM1
VM2
VM3
VM4
VM5
VM6
VM7
P0,0 P1,0 P2,0 P3,0 P4,0 P5,0 P6,0 P7,0
P0,1 P1,1 P2,1 P3,1 P4,1 P5,1 P6,1 P7,1
P0,2 P1,2 P2,2 P3,2 P4,2 P5,2 P6,2 P7,2
P0,3 P1,3 P2,3 P3,3 P4,3 P5,3 P6,3 P7,3
P0,4 P1,4 P2,4 P3,4 P4,4 P5,4 P6,4 P7,4
P0,5 P1,5 P2,5 P3,5 P4,5 P5,5 P6,5 P7,5
P0,6 P1,6 P2,6 P3,6 P4,6 P5,6 P6,6 P7,6
P0,7 P1,7 P2,7 P3,7 P4,7 P5,7 P6,7 P7,7

图 5

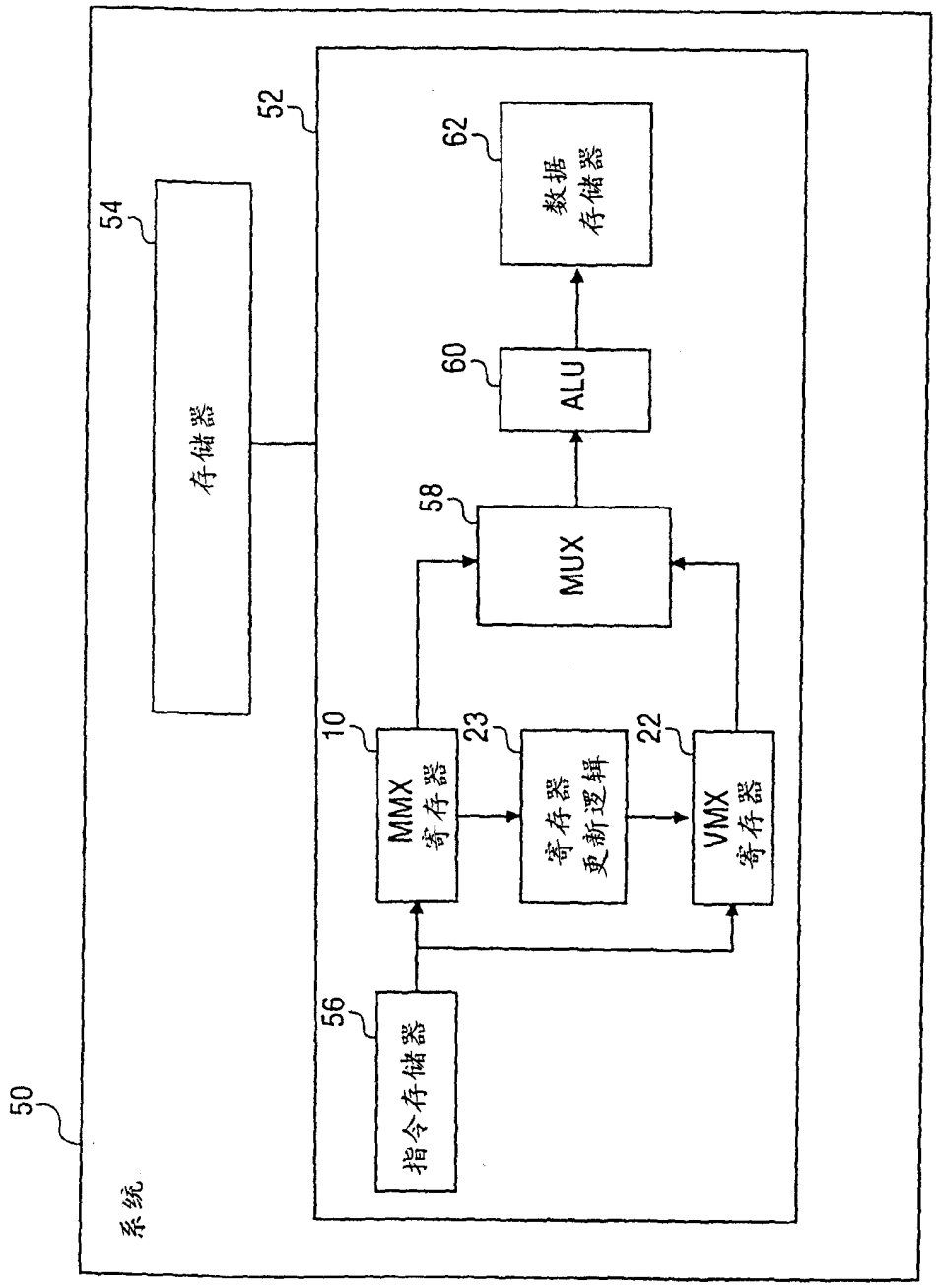

50
系统
52
54
存储器
62
数据存储器
60
ALU
58
MUX
10
MMX寄存器
23
寄存器更新逻辑
22
VMX寄存器
56
指令存储器

图 6