[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/302 (2006.01)

[12] 发明专利说明书

专利号 ZL 200380110346.7

[45] 授权公告日 2008 年 4 月 2 日

[11] 授权公告号 CN 100378651C

[22] 申请日 2003.12.18
[21] 申请号 200380110346.7
[30] 优先权
[32] 2003.6.16 [33] US [31] 10/461,880
[86] 国际申请 PCT/GB2003/005548 2003.12.18
[87] 国际公布 WO2004/114127 英 2004.12.29
[85] 进入国家阶段日期 2005.12.13
[73] 专利权人 ARM 有限公司
地址 英国剑桥郡
[72] 发明人 D·克尔肖
[56] 参考文献
EP1302848A 2003.4.16
US5636351A 1997.6.3
审查员 孟宪超

[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 杨 凯 王 勇

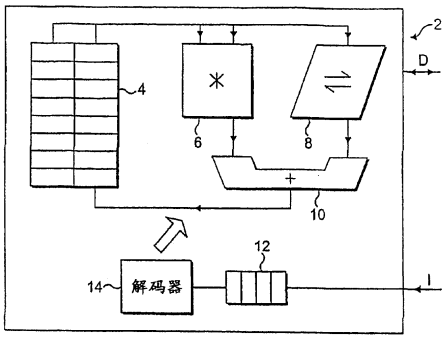
权利要求书 5 页 说明书 8 页 附图 6 页

[54] 发明名称

单指令多数据数据处理系统内的结果分割

[57] 摘要

在提供单指令多数据(SIMD)类型运算的处理器(2)中，单数据处理指令可用于控制处理逻辑(4、6、8、10)对多个独立输入值执行 SIMD 类型处理运算以产生具有比相应输入值更大的数据宽度的多个独立结果值。 适当控制的复用器形式的再分割器用于将这些结果数据值分割为存储在分开的寄存器(38、40)中的高阶位部分和低阶位部分。 保留要求的 SIMD 宽度的结果值可以从所需高阶结果寄存器(38)或低阶结果寄存器(40)中读取而无需进一步处理。 此外，保留完整结果有助于提高例如在扩展累加运算等等上的精确性。



1. 用于执行数据处理操作以响应数据处理指令的装置，所述装置包括:

处理设备，响应所述数据处理指令，从存储在一个或多个输入存储器中的多个独立数据值产生相应的多个结果数据值；以及

结果分割器，响应所述数据处理指令，将每个结果数据值的高阶位部分存储到高阶结果存储器中，并将每个结果数据值的低阶位部分存储到低阶结果存储器中。

2. 如权利要求 1 所述的装置，其特征在于，所述处理设备可用于将各对独立数据值相乘，一对的第一独立数据值取自第一输入存储器，而所述对的第二独立数据值取自第二输入存储器。

3. 如权利要求 2 所述的装置，其特征在于，处理设备可用于将已存储在所述高阶结果存储器和所述低阶结果存储器中的值与从所述各对独立数据值产生的值累加以产生所述多个结果数据值。

4. 如权利要求 1 至 3 中的任一项所述的装置，其特征在于，每个结果数据值的所述高阶位部分和所述低阶位部分是所述结果数据值的非交迭邻接部分。

5. 如权利要求 2 所述的装置，其特征在于，所述装置还包括多个复用器，当以带符号小数模式运算时，提供给所述复用器的小数模式指示信号用于将进位保留输出移一个位的位置，这相当于以补偿在最高有效位置的额外符号位的方式将值翻倍。

6. 如权利要求 1 所述的装置，其特征在于，每个输入存储器存储 M 个独立的 N 位数据值，其中 M 和 N 为大于或等于 2 的整数。

7. 如权利要求 6 所述的装置，其特征在于，所述数据处理指令指定所述独立数据值的数据宽度。

8. 如权利要求 2 所述的装置，其特征在于，所述处理设备包括整数乘法器，可用于将所述各对独立数据值相乘。

9. 如权利要求 1 所述的装置，其特征在于，所述处理设备可用于对所述独立数据值执行饱和数据处理运算。

10. 如权利要求 1 所述的装置，其特征在于，所述结果分割器包括依靠所述数据处理指令控制的多个复用器。

11. 如权利要求 1 所述的装置，其特征在于，所述装置是处理器核心。

12. 如权利要求 1 所述的装置，其特征在于，所述一个或多个输入存储器是以下各项中的一个或多个:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器；以及

内存。

13. 如权利要求 1 所述的装置，其特征在于，所述高阶结果存储器是以下各项其中之一:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器；以及

内存。

14. 如权利要求 1 所述的装置，其特征在于，所述低阶结果存储器是以下各项其中之一:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器；以及

内存。

15. 如权利要求 1 所述的装置，其特征在于，所述处理设备可用

于为每个结果数据值产生一个或多个高阶保护位，且所述结果分割器可用于将所述保护位存储在保护位存储器中。

16. 如权利要求 15 所述的装置，其特征在于，所述保护位存储器是以下各项其中之一:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器；以及

内存。

17. 一种执行数据处理操作以响应数据处理指令的方法，所述方法包括以下步骤:

响应所述数据处理指令，从存储在一个或多个输入存储器中的多个独立数据值产生相应的多个结果数据值；以及

响应所述数据处理指令，通过将每个结果数据值的高阶位部分存储到高阶结果存储器中，并将每个结果数据值的低阶位部分存储到低阶结果存储器中，分割所述结果数据值。

18. 如权利要求 17 所述的方法，其特征在于，将各对独立数据值相乘，一对的第一独立数据值取自第一输入存储器，而所述对的第二独立数据值取自第二输入存储器。

19. 如权利要求 18 所述的方法，其特征在于，将已存储在所述高阶结果存储器和所述低阶结果存储器中的值与从所述各对独立数据值产生的值累加以产生所述多个结果数据值。

20. 如权利要求 17 至 19 中的任一项所述的方法，其特征在于，每个结果数据值的所述高阶位部分和所述低阶位部分是所述结果数据值的非交迭邻接部分。

21. 如权利要求 18 所述的方法，其特征在于，所述方法依靠多个复用器，当以带符号小数模式运算时，提供给所述复用器的小数模式指示信号用于将进位保留输出移一个位的位置，这相当于以补

偿在最高有效位置的额外符号位的方式将值翻倍。

22. 如权利要求 17 所述的方法，其特征在于，每个输入存储器存储 M 个独立的 N 位数据值，其中 M 和 N 为大于或等于 2 的整数。

23. 如权利要求 22 所述的方法，其特征在于，所述数据处理指令指定所述独立数据值的数据宽度。

24. 如权利要求 18 所述的方法，其特征在于，整数乘法器可用于将所述各对独立数据值相乘。

25. 如权利要求 17 所述的方法，其特征在于，对所述独立数据值执行饱和数据处理运算。

26. 如权利要求 17 所述的方法，其特征在于，分割至少部分由依靠所述数据处理指令控制的多个复用器执行。

27. 如权利要求 17 所述的方法，其特征在于，所述方法在处理器核心内执行。

28. 如权利要求 17 所述的方法，其特征在于，所述一个或多个输入存储器是以下各项中的一个或多个:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器; 以及

内存。

29. 如权利要求 17 所述的方法，其特征在于，所述高阶结果存储器是以下各项其中之一:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器; 以及

内存。

30. 如权利要求 17 所述的方法，其特征在于，所述低阶结果存

储器是以下各项其中之一:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器;以及

内存。

31. 如权利要求 17 所述的方法,其特征在于,所述处理设备可用于为每个结果数据值产生一个或多个高阶保护位,并且所述结果分割器可用于将所述保护位存储在保护位存储器中。

32. 如权利要求 31 所述的方法,其特征在于,所述保护位存储器是以下各项其中之一:

寄存器组寄存器;

专用寄存器;

缓冲存储器;

先入先出缓冲器;以及

内存。

单指令多数据数据处理系统内的结果分割

技术领域

本发明涉及数据处理系统领域。更具体地说，本发明涉及在单指令多数据(SIMD)数据处理系统内包含多个结果数据值的结果的分割。

背景技术

为数据处理系统提供 SIMD 功能是已知的。在此类系统中，寄存器通常包含要处理的多个独立数据值。例如，32 位寄存器可以包含两个独立的 16 位数据值，要分别加到、乘以或组合到例如存储在另一个 32 位寄存器中的两个其他 16 位数据值。此类 SIMD 运算在数字信号处理领域是常见的，并且具有包括增大的处理速度和减小的代码密度在内的优势。

已知 SIMD 技术的示例是 Intel Corporation 生产的 Intel 处理器的 MMX 指令。MMX 指令包含将两个寄存器相乘的指令，每个寄存器包含四个 16 位数据值。在一个 16 位数据值乘以另一个 16 位数据值时，结果则为 32 位数据值。因此，MMX SIMD 指令中指定的四对 16 位数据值相乘时，结果为四个 32 位结果数据值。在许多情形下，在执行此类运算时希望保持 SIMD 格式和数据大小。为此，MMX 指令包括一种指令，其中在上述情形下产生的结果为四个 16 位结果数据值的形式，即各个 32 位结果的 16 个最高有效位，这些 16 位值在单个 64 位寄存器中组合，即产生 SIMD 类型的结果。或者，也可以将分开的指令在 64 位寄存器中组合，这些指令产生乘法结果的四个最低有效 16 位作为其输出。

发明内容

从一方面来看，本发明提供执行数据处理操作以响应数据处理指令的装置，所述装置包括:

处理设备，响应所述数据处理指令，从存储在一个或多个输入存储器中的多个独立数据值产生相应的多个结果数据值；以及

结果分割器，响应所述数据处理指令，将每个结果数据值的高阶位部分存储到高阶结果存储器中，并将每个结果数据值的低阶位部分存储到低阶结果存储器中。

本发明认识到，虽然在许多情形下可能希望产生 SIMD 类型的结果，但在某些情形下，保持结果完全精度以便避免不当舍入误差之类的不利后果是重要的。因此，本技术提供一个系统，其中响应单个数据处理指令，从而产生高代码密度，对多个独立数据值执行 SIMD 类型的运算，将多个结果数据值以 SIMD 形式存储，高阶部分在一个存储器中，而低阶部分在另一个存储器中。因此，如有要求，SIMD 类型的结果无需进一步处理就立即可用，且完全精度得以保持并可继续发扬，因为结果的所有精度在两个存储器的组合中保持并可以从中进行处理。

应该理解，由处理设备执行以从多个独立输入数据值产生结果数据值的数据处理操作类型可以采取多种多样的不同形式。处理设备的输入可能是单个存储器的内容，结果是存储在其中的独立数据值的平方，或那些值的平方根，以根据某些计算技术达到特定精度，等等。然而，在本发明的优选实施例中，处理设备可用于将各对独立数据值相乘，一对的第一独立数据值取自第一输入存储器，而一对的第二独立数据值取自第二输入存储器。

此类 SIMD 类型的乘法运算是常见的且增大结果的数据宽度，如要保持完全精度并直接产生 SIMD 类型的结果，则要求使用本技术。

本技术尤其适合于累加运算与乘法相关联的情形，因为本技术

保持的额外精度有助于避免在累加类运算中可能出现的多个舍入误差的累积影响。

虽然可以理解高阶位部分和低阶位部分可以具有各种不同的关系，但是当它们是有关结果数据值的非交迭邻接部分时是最有效和优选的。

数据处理指令可以指定各种不同形式的乘法运算，如整数乘法或带符号小数值乘法。但是，本发明尤其适合于如下情形，其中指定的乘法是带符号小数乘法且处理设备可用于将每个结果数据值翻倍以便考虑每个输入数据值上存在的带符号位。翻倍可有效地包含在其他运算中，具有极小额外开销。

独立 SIMD 数据值的数据宽度可以变化，并且在优选实施例中，数据处理指令指定相关的数据宽度。

取决于要求的具体情况，乘法器可以采取若干形式，但特别优选的形式是整数乘法器，因为这相对简单和快速，且能够以适当的配置产生多种多样不同类型的运算。

作为可由数据处理指令指定的处理运算类型的实例，处理可以可选地导致执行饱和运算。

结果分割器用于在不同存储器之间分割结果数据值，在优选实施例中，多个复用器用于执行此功能。本技术可应用于许多不同类型的数据处理系统，如 DSP，但特别适合用于处理器核心中。

应该理解，输入存储器、高阶结果存储器、低阶结果存储器以及系统内的存储器可以具有多种多样的不同形式，但优选实施例是一个或多个寄存器组寄存器、专用寄存器、缓冲存储器、先入先出缓冲器或存储器的一部分(例如高速缓存、主存储器、大容量存储器等)。这些不同类型的存储器可用于不同存储器具有不同形式的混合情况。在使用内存或缓冲器而不是寄存器作为存储器的情况下，要处理的数据值序列流可方便地提供。

作为以与本技术轻松兼容的方式增加要计算的结果范围的一种

方式，优选实施例还产生一个或多个高阶保护位，如可用于饱和运算的上下文中。可以为这些保护位提供其自己的存储器，结果分割把保护位存储到这些存储器。

从另一方面来看，本发明提供执行数据处理操作以响应数据处理指令的方法，所述方法包括以下步骤：

响应所述数据处理指令，从存储在一个或多个输入存储器中的多个独立数据值产生相应的多个结果数据值；以及

响应所述数据处理指令，通过将每个结果数据值的高阶位部分存储到高阶结果存储器中，并将每个结果数据值的低阶位部分存储到低阶结果存储器中，分割所述结果数据值。

附图说明

现在将通过实例，参照附图来描述本发明的实施例，其中：

图 1 示意说明可以利用本技术的类型的处理器核心；

图 2 示意说明各种 SIMD 数据格式；

图 3 示意说明根据用于各种数据宽度的本技术的输入数据值和输出数据值之间的关系；

图 4 示意说明图 1 的处理器核心内的数据处理路径的一部分；

图 5 和图 5 续说明用于根据本技术分割结果数据值的复用配置；以及

图 6 示意说明根据本技术的乘法累加运算的另一种形式。

具体实施方式

图 1 说明处理器核心 2，例如由 ARM Limited of Cambridge, England 制造。处理器核心 2 包括寄存器组 4、乘法器 6、移位器 8 和加法器 10，形成数据处理数据路径的一部分。数据处理指令从它们被指令解码器 14 解码的位置接收到指令流水线 12 中，以产生控制信号来控制处理器 2 中其他电路元件的操作。应该理解，处理器 2

通常包括更多电路元件，但为简洁起见，这些元件未示出。在图 1 的实例中，从寄存器组 4 中的寄存器读取输入数据值，且带有数据值的结果写回到寄存器组 4 的寄存器中。在其他实施例中，输入值和结果值可从不同类型的存储器读取并写入不同类型的存储器，如专用寄存器、缓冲存储器、先入先出缓冲器和通用存储器。这些可用作备选物并在各种混合组合中使用。这些不同的备选者在图 1 中未示出。

图 2 说明各种不同的 SIMD 数据格式。图 1 所示数据路径的数据宽度在经修改以支持此数据宽度的 ARM 处理器版本中可以是 64 位。此数据路径可以在非 SIMD 模式中处理完整长度的 64 位字 16。在此实例中，各种 SIMD 模式处理两个 32 位数据值、四个 16 位数据值或八个 8 位数据值。在 SIMD 模式中，数据值是相互独立的，并且图 1 的处理器 2 中的数据路径根据 SIMD 数据值的大小配置，以便独立地处理这些数据值，例如通过在适当的点断开进位链。执行 SIMD 类型运算的数据路径的适配本身已知，本文不作进一步说明。

图 3 说明根据本技术的不同 SIMD 数据宽度模式中的输入数据值和结果数据值之间的关系。在实例(i)中，输入数据值包含存储在第一 64 位寄存器中的两个 32 位输入值 A0 和 A1，以及存储在第二寄存器中的两个 32 位输入值 B0 和 B1。在此实例中，由处理指令指定的数据处理运算为 SIMD 乘法，且相应地 32 位值 A0 与 32 位值 B0 相乘，32 位值 A1 与 32 位值 B1 相乘。这些乘法都会产生 64 位结果，分别为 A0 B0 和 A1 B1。这两个结果的最高有效 32 位写入高阶结果寄存器 17。这两个结果的最低有效 32 位写入低阶结果寄存器 18。写入不同的寄存器 17 和 18 的这两个部分是非交迭且邻接的。

实例(ii)和(iii)是类似的，分别涉及 16 位输入值和 8 位输入值作为 SIMD 乘法指令的相乘对象，在不同的寄存器中产生相应结果数据值是总结果的高阶半部分或总结果的低阶半部分。

如果希望在相同数据宽度的进一步 SIMD 类型运算中使用由乘法产生的结果继续进一步处理，则高阶结果寄存器 17 可以被直接读取并用作此类进一步运算的输入。无需移位或重新配置以改进代码密度、速度，功耗等等。特别优选的环境是高阶结果寄存器 17 和低阶结果寄存器 18 被用作累加运算的目的地，在其中连续乘法的结果可以累加到这些寄存器中，且保留在低阶结果寄存器 18 中的低阶结果值被连续更新以便产生更精确的结果并避免舍入误差。这样，本技术允许使用单个指令直接访问正确的数据宽度值，并且由于保持结果的完整数据宽度，还保持了精确性。

图 4 示意说明图 1 的一部分数据路径的更多细节。取自寄存器组 4 的相应寄存器的两个 64 位输入值被提供给 SIMD 整数乘法器 20。这些输入值可以代表单个 64 位乘以 64 位非 SIMD 运算，或上述三种 SIMD 类型的运算之一。SIMD 乘法器 20 在进位链中包括适当的中断等，以正确分割独立输入值和结果输出值。SIMD 乘法器 20 的输出为进位保留格式。系统以带符号小数模式运算时，提供给复用器 22 和 24 的小数模式指示信号用于将进位保留输出移一个位的位置，这相当于以补偿在最高有效位置的额外符号位的方式将值翻倍。加法器 26 用于将来自 SIMD 乘法器 20 的进位保留输出与来自保存和进位寄存器 28、30 的再循环部分累加值相加，或与来自由复用器 32、34 所选的寄存器组 4 的寄存器 D、C 的 128 位值相加。复用器 32、34 由累加控制信号控制，其各种值如图 4 底部的表格所示。系统可以配置为从源寄存器文件累加、相乘而不累加、或累加到先前部分计算的结果，例如在矢量类型运算期间，绕过作为累加值的源的寄存器组。

在给定处理运算的乘法和加法运算已完成时，来自寄存器 28、30 的最终 128 位保存和进位值将传递到加法器 36，它们在此相加以形成常规的 128 位结果表示。乘法和加法可以是流水线操作。应该理解，加法器 36 的输出与来自寄存器 A、B 的 64 位输入值相比具有双

倍位宽。这样，SIMD 结果值具有独立 SIMD 输入值两倍的宽度。加法器 36 的输出提供给结果分割器，在此示范实施例中，结果分割器具有图 5 和图 5 续中所示的各种复用器的形式。

在图 5 续中，高阶结果寄存器 38 接收每个结果值的所选部分，即其高阶部分。在图 5 中，低阶结果寄存器 40 接收结果值的相应低阶部分。控制信号 B、H、W 和 L 代表正在使用的 SIMD 数据宽度(字节、半字、字或长字)。在任何一个时刻，这些值之一被断言为“1”，其他为“0”。这些宽度指定信号根据图 5 和图 5 续所示的各复用器旁边提供的逻辑表达式控制每个复用器，以便在相关复用器的各个输入之间选择。由其控制信号控制的图 5 和图 5 续的复用器的全部动作是从加法器 36 输出的 128 位中选择/重新分割以形成高阶结果寄存器 38 和低阶结果寄存器 40 的内容，如图 3 的不同实例中所示。

提供给图 1 的解码器 14 以通过所示方式控制图 4 和图 5 以及图 5 续的电路的程序指令具有语法，包括指定所使用的数据宽度是非 SIMD 完整数据宽度还是多种 SIMD 数据宽度之一的参数。程序指令还指定是否进行累加以及这是使用外部寄存器值还是“内部”部分结果来完成。

除图 5 和图 5 续中的两个结果寄存器 38 和 40 之外，还可以提供保护寄存器。在这个保护寄存器中提供从累加结果的扩展版本计算的保护位。举例来说，如果 16 位 SIMD 数据值用于乘法累加运算，取决于提供两个还是四个保护位，累加器将大于 32 位，例如 34 或 36 位，使得累加值的溢出被容纳于保护位中。在此类实施例中，保护位可以被分割到分开的保护位寄存器中，且在这种形式中，保护位寄存器可以视为在结果的最高有效端提供保护位，低阶结果寄存器在结果值的低端提供保护位，而高阶结果寄存器提供通常要求的 SIMD 宽度保留数据值。

图 6 示意说明多数据格式的多累加运算，给出堆叠的寄存器结果。

寄存器 A 和 B 是 64 位 SIMD 寄存器，在此例中保存 4 个 16 位量(A0 - A3 和 B0 – B3)。这些寄存器相乘的结果是 4 个结果的矢量，其中每个可以最多 32 位宽。

可以将 4 个 32 位乘法结果与保存在另两个各保存 2 个 32 位量的寄存器(C 和 D)中的四个 32 位值累加。

然后，加法结果可以以堆叠格式存储在寄存器 RL 和 RH 中。

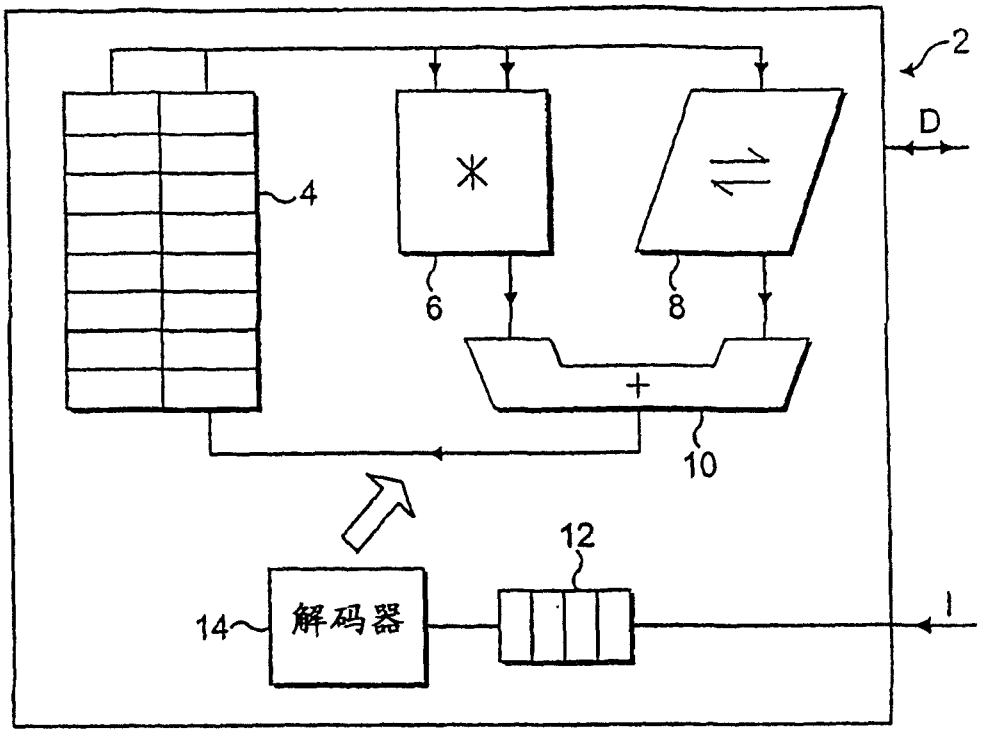


图 1

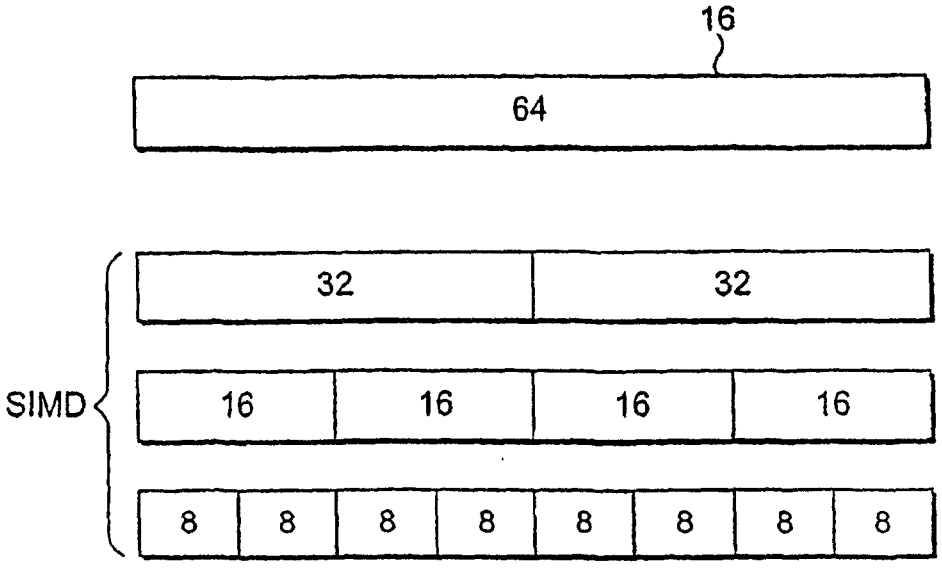
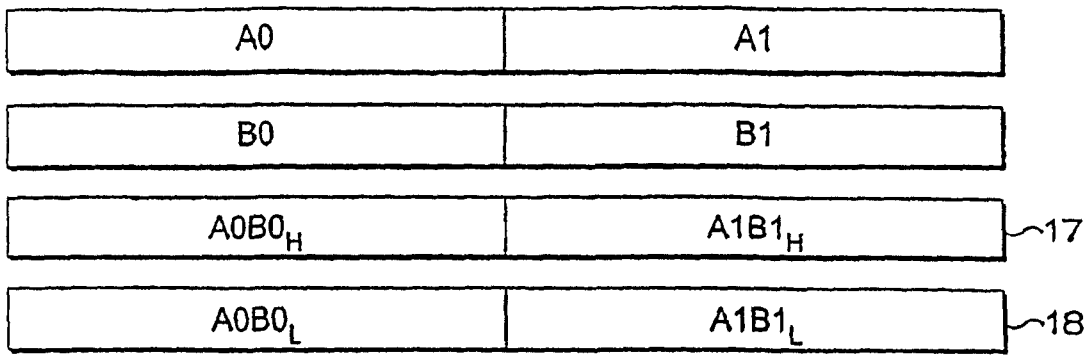
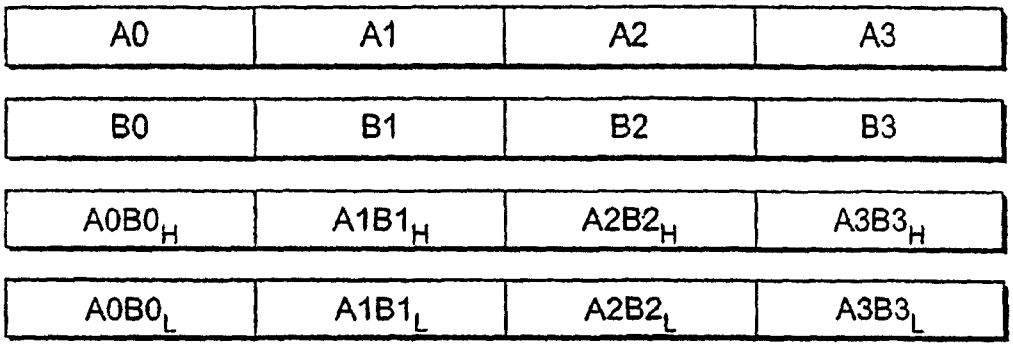


图 2

(i)


A0
A1
B0
B1
A0B0H
A1B1H
17
A0B0L
A1B1L
18

(ii)


A0
A1
A2
A3
B0
B1
B2
B3
A0B0H
A1B1H
A2B2H
A3B3H
A0B0L
A1B1L
A2B2L
A3B3L

(iii)

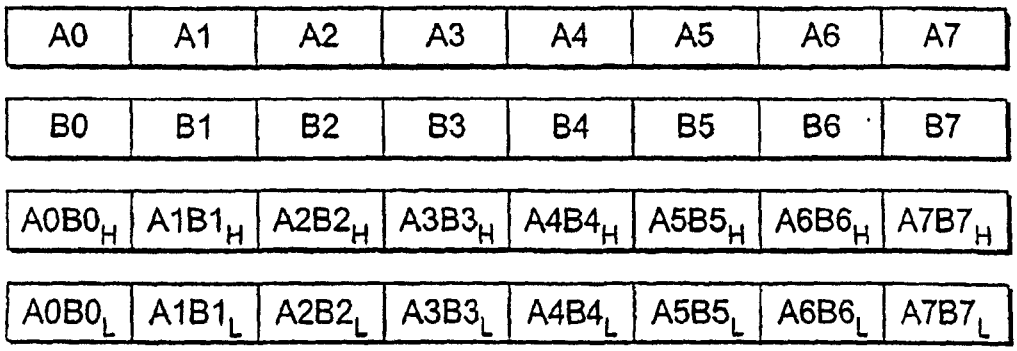
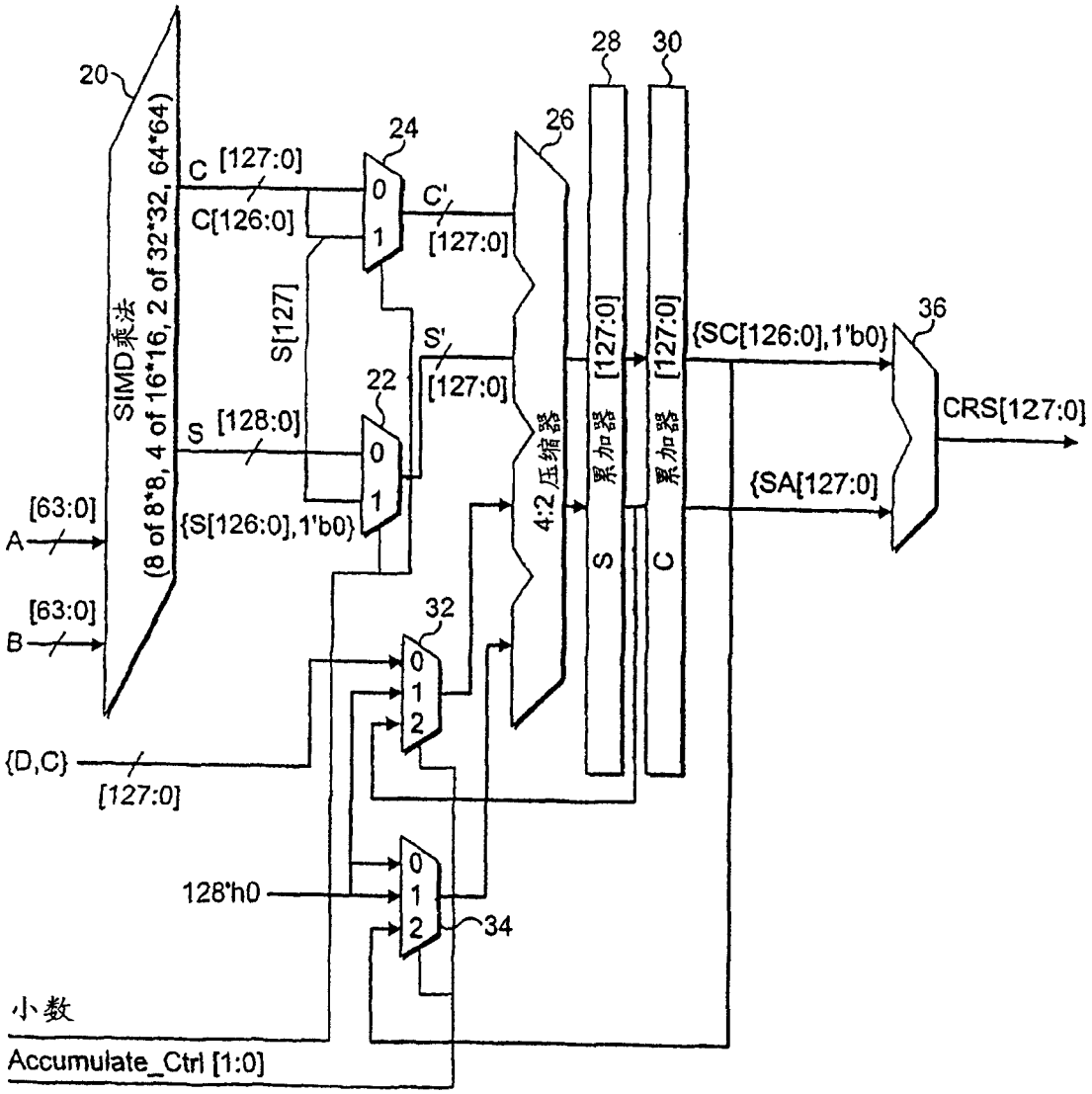

A0
A1
A2
A3
A4
A5
A6
A7
B0
B1
B2
B3
B4
B5
B6
B7
A0B0H
A1B1H
A2B2H
A3B3H
A4B4H
A5B5H
A6B6H
A7B7H
A0B0L
A1B1L
A2B2L
A3B3L
A4B4L
A5B5L
A6B6L
A7B7L

图 3

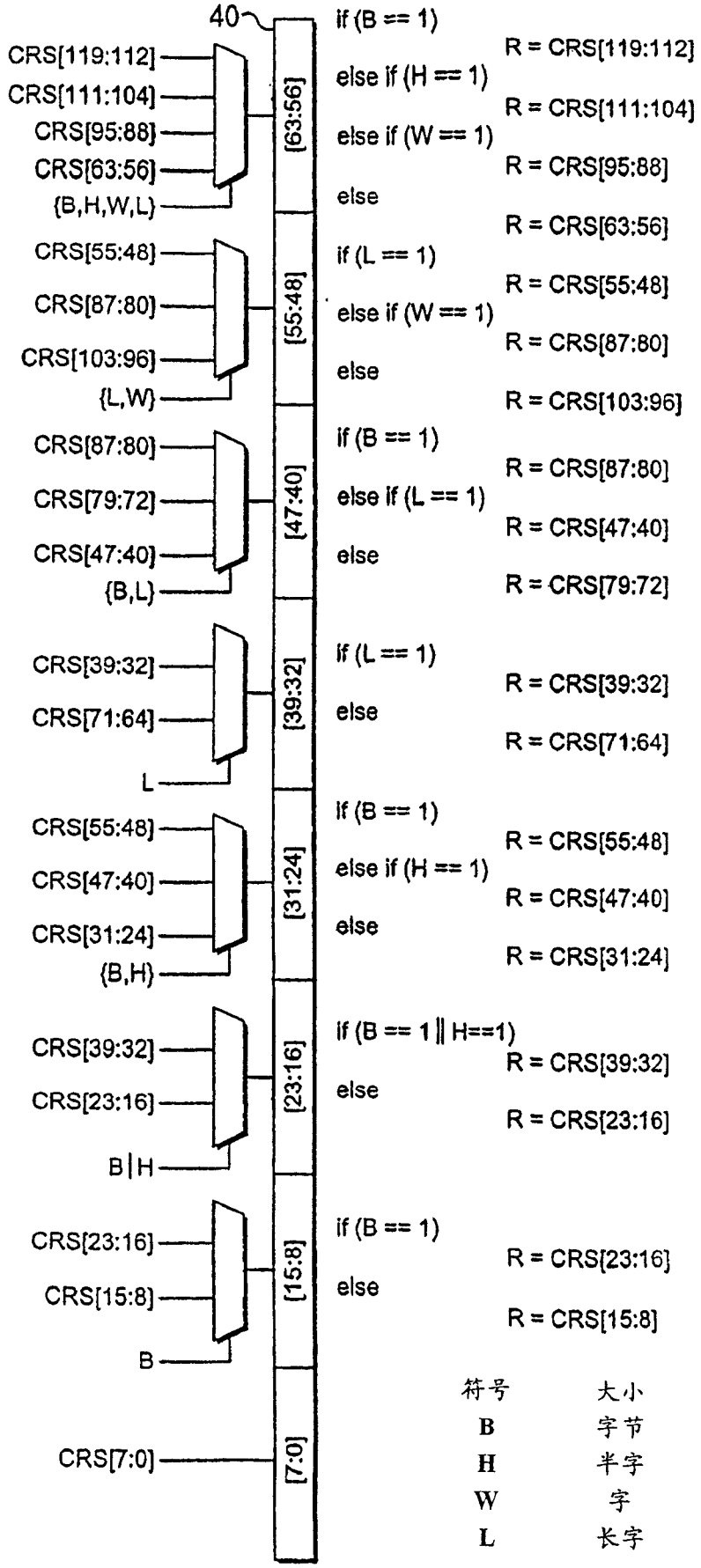


Accumulate_Ctrl

[0]	[1]	动作
0	0	累加来自寄存器文件的源
1	0	乘法(无累加)
0	1	累加至先前结果-绕过寄存器组源
1	1	非法

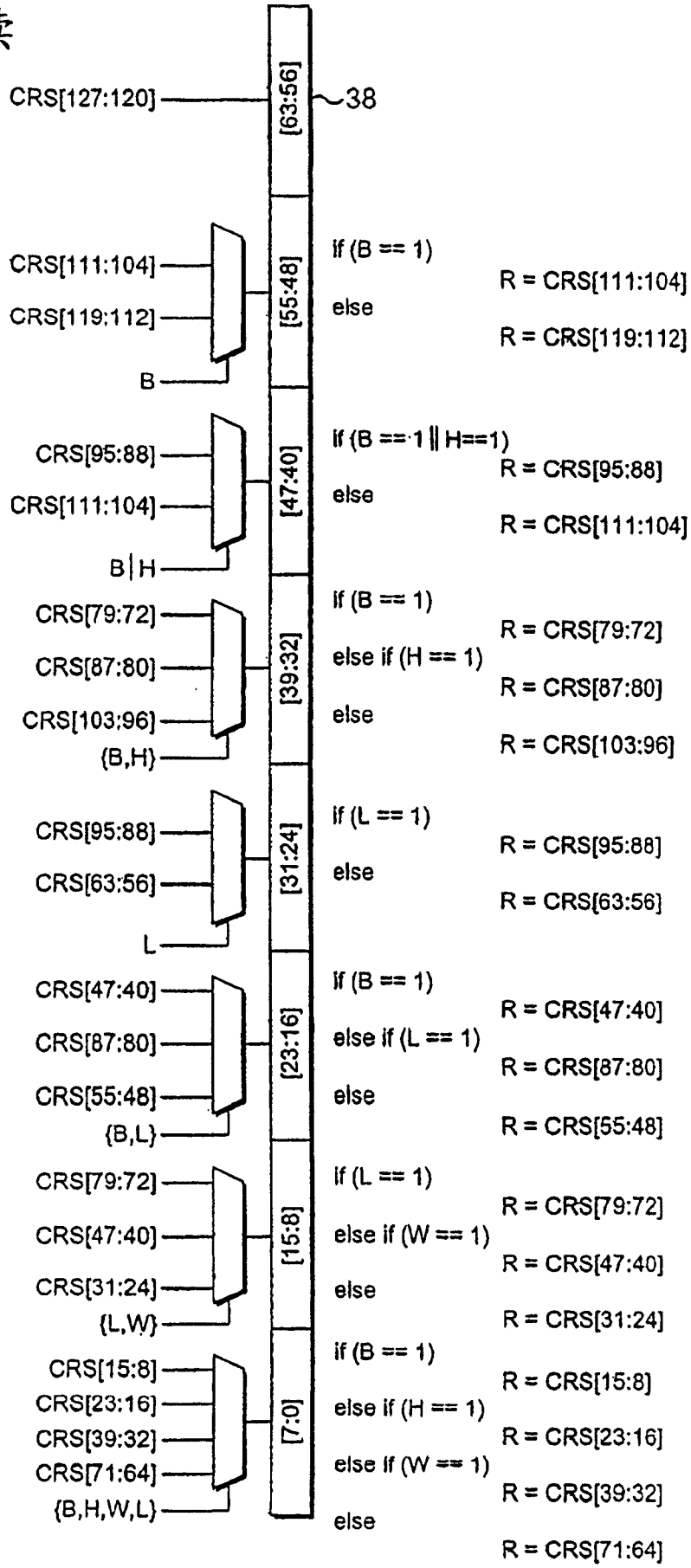
图 4

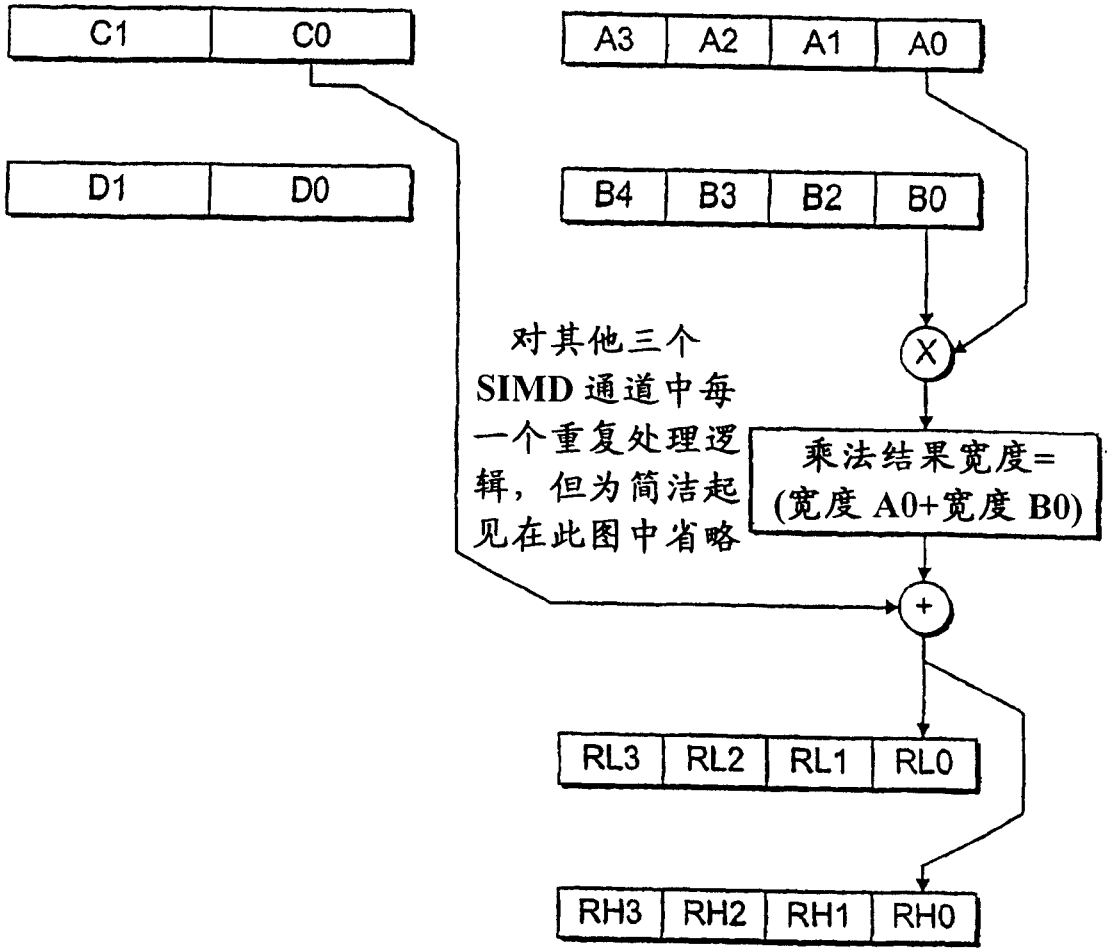
图 5



符号	大小	
B	字节	(8 位)
H	半字	(16 位)
W	字	(32 位)
L	长字	(64 位)

图 5 续


CRS[127:120]
[63:56]
38
CRS[111:104]
CRS[119:112]
B
[55:48]
if (B == 1)
R = CRS[111:104]
else
R = CRS[119:112]
CRS[95:88]
CRS[111:104]
B|H
[47:40]
if (B == 1 || H==1)
R = CRS[95:88]
else
R = CRS[111:104]
CRS[79:72]
CRS[87:80]
CRS[103:96]
{B,H}
[39:32]
if (B == 1)
R = CRS[79:72]
else if (H == 1)
R = CRS[87:80]
else
R = CRS[103:96]
CRS[95:88]
CRS[63:56]
L
[31:24]
if (L == 1)
R = CRS[95:88]
else
R = CRS[63:56]
CRS[47:40]
CRS[87:80]
CRS[55:48]
{B,L}
[23:16]
if (B == 1)
R = CRS[47:40]
else if (L == 1)
R = CRS[87:80]
else
R = CRS[55:48]
CRS[79:72]
CRS[47:40]
CRS[31:24]
{L,W}
[15:8]
if (L == 1)
R = CRS[79:72]
else if (W == 1)
R = CRS[47:40]
else
R = CRS[31:24]
CRS[15:8]
CRS[23:16]
CRS[39:32]
CRS[71:64]
{B,H,W,L}
[7:0]
if (B == 1)
R = CRS[15:8]
else if (H == 1)
R = CRS[23:16]
else if (W == 1)
R = CRS[39:32]
else
R = CRS[71:64]



RL0 = 下部分 (C0 + A0*B0)
RH0 = 上部分 (C0 + A0*B0)

RL1 = 下部分 (C1 + A1*B1)
RH1 = 上部分 (C1 + A1*B1)

RL2 = 下部分 (D0 + A2*B2)
RH2 = 上部分 (D0 + A2*B2)

RL3 = 下部分 (D1 + A3*B3)
RH3 = 上部分 (D1 + A3*B3)

图 6