(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 103678190 B
(45)授权公告日 2016.10.26

(21)申请号 201310384449.8

(22)申请日 2013.08.29

(65)同一申请的已公布的文献号
申请公布号 CN 103678190 A

(43)申请公布日 2014.03.26

(30)优先权数据
1215425.8 2012.08.30 GB

(73)专利权人 想象力科技有限公司
地址 英国赫特福德郡

(72)发明人 P·默林 A·J·安德森
M·厄尔-哈加

(74)专利代理机构 永新专利商标代理有限公司
72002
代理人 刘瑜 王英

(51)Int.Cl.
G06F 13/16(2006.01)
G06F 13/20(2006.01)
G06F 13/28(2006.01)

(56)对比文件
CN 1713678 A,2005.12.28,
CN 101237240 A,2008.08.06,
US 2009/0313399 A1,2009.12.17,

审查员 徐春

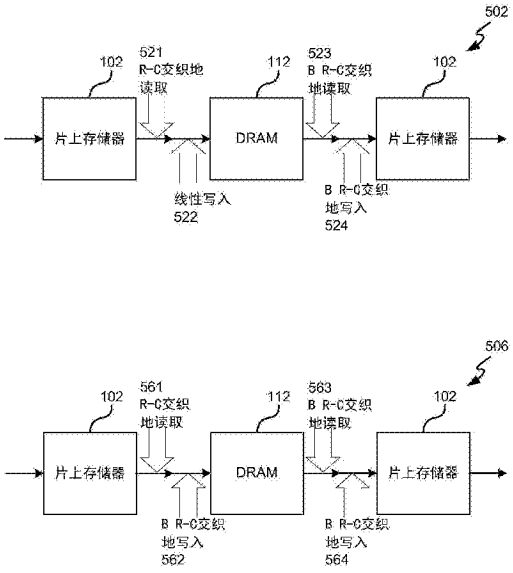
权利要求书3页 说明书17页 附图10页

(54)发明名称

用于数字信号处理的基于片区的交织和解交织

(57)摘要

描述了行列交织的数据的基于片区(tile)的交织和解交织。在一个示例中,将解交织划分成两个存储器传输阶段,第一阶段从片上存储器到DRAM,第二阶段从DRAM到片上存储器。每一阶段对行列交织的数据块的一部分进行操作,并对这些数据项进行重新排序,使得第二阶段的输出包括解交织的数据。在第一阶段中,根据非线性存储器读地址序列从片上存储器读取数据项,并将这些数据项写入DRAM。在第二阶段中,根据线性地址序列的突发,从DRAM读取数据项,这高效地利用了DRAM接口,并根据非线性存储器写地址序列,将这些数据项写回到片上存储器。



CN 103678190 B

1.一种片上数字信号处理系统,包括:

第一存储器(102),其存储以第一序列排列的多个数据项,每一个数据项具有所述第一存储器上的相关联的存储器地址,并且所述多个数据项包括数据项块的子集;

第二存储器;以及

传输引擎,其耦接到所述第一存储器和所述第二存储器,并且包括通往动态随机存取存储器DRAM的端口,其中,所述传输引擎被配置为:在第一存储器传输阶段中,将所述多个数据项从所述第一存储器直接传输到所述DRAM,并且在第二存储器传输阶段中,将所述多个数据项从所述DRAM直接传输到所述第二存储器,并且

其中,在所述第一存储器传输阶段中,所述传输引擎用于:根据预定的非线性存储器读地址序列,从所述第一存储器读取所述多个数据项,并将所述多个数据项写入所述DRAM,并且

其中,在所述第二存储器传输阶段中,所述传输引擎用于:根据线性地址序列的突发,从所述DRAM读取所述多个数据项,线性地址序列的每一个突发具有基于DRAM接口突发大小所选定的长度,以及根据预定的非线性存储器写地址序列,将所述多个数据项写入所述第二存储器,使得所述多个数据项以与所述第一序列不相同的第二序列排列在所述第二存储器上,并且其中,所述第一序列和所述第二序列中的一个包括行列交织的数据,所述第一序列和所述第二序列中的另一个包括行列解交织的数据。

2.根据权利要求1所述的片上数字信号处理系统,其中,所述第一存储器和所述第二存储器都是静态随机存取存储器。

3.根据权利要求1所述的片上数字信号处理系统,其中,所述第一存储器和所述第二存储器是同一片上存储器。

4.根据权利要求1所述的片上数字信号处理系统,还包括所述DRAM。

5.根据权利要求1所述的片上数字信号处理系统,其中,所述传输引擎还用于:重复所述第一存储器传输阶段和所述第二存储器传输阶段,直到所述数据项块的全部都已经被写入所述第二存储器为止。

6.根据权利要求1所述的片上数字信号处理系统,还包括:

至少一个地址生成元件(210),其用于生成所述预定的非线性存储器读地址序列和所述预定的非线性存储器写地址序列。

7.根据权利要求1所述的片上数字信号处理系统,其中,所述数据项块被定义为排列成网格,所述网格包括多个数据项行和多个数据项列。

8.根据权利要求7所述的片上数字信号处理系统,其中,所述网格还包括多个片区,每一个片区包括所述网格的一个矩形部分,并且还包括R个数据项行和C个数据项列,并且其中,所述多个数据项包括一个或多个片区。

9.根据权利要求8所述的片上数字信号处理系统,其中,对于第一多个数据项中的每一个片区,所述预定的非线性存储器读地址序列包括:间隔开固定数量的存储器地址并开始于初始起始地址直到到达该片区的边界为止的非连续存储器地址序列,之后是一个或多个额外的非连续存储器地址序列,所述固定数量与所述网格中的行的数量减去一相对应,每一个额外的序列开始于偏移的初始起始地址。

10.根据权利要求8所述的片上数字信号处理系统,其中,所述预定的非线性存储器写

地址序列包括:间隔开所述第二存储器中的固定数量的存储器地址并开始于所述第二存储器中的初始起始地址的具有C个连续存储器地址的组的序列,所述固定数量与所述网格中的列的数量减去C相对应。

11.根据权利要求8所述的片上数字信号处理系统,其中,所述多个数据项包括所述网格的片区。

12.根据权利要求8所述的片上数字信号处理系统,其中,在所述第二存储器传输阶段中,所述线性地址序列的突发包括:间隔开所述第二存储器中的固定数量的存储器地址并开始于所述第二存储器中的初始起始地址的具有X个连续存储器地址的突发的序列,其中,X等于所述网格的片区中的数据项的数量。

13.根据权利要求8所述的片上数字信号处理系统,其中,在所述第一存储器传输阶段中,所述传输引擎用于:根据线性地址序列的突发,将所述多个数据项写入所述DRAM,线性地址序列的每一个突发具有基于DRAM接口突发大小所选定的长度。

14.根据权利要求13所述的片上数字信号处理系统,其中,在所述第一存储器传输阶段中,所述线性地址序列的突发包括:间隔开所述第二存储器中的固定数量的存储器地址并开始于所述第二存储器中的初始起始地址的具有X个连续存储器地址的突发的序列,其中,X等于所述网格的片区中的数据项的数量。

15.根据权利要求8所述的片上数字信号处理系统,其中,片区的大小是基于所述DRAM接口突发的大小进行调整的。

16.一种在数字信号处理系统中对数据项块执行交织或解交织操作的方法,所述方法包括:

根据预定的非线性存储器读地址序列,从第一片上存储器读取(521、561、804)以第一序列存储的第一多个数据项,其中,所述第一多个数据项包括所述数据项块的子集;

将所述第一多个数据项写入(522、562、806)动态随机存取存储器DRAM;

根据线性地址序列的突发,从所述DRAM读取(523、563、814)所述第一多个数据项,线性地址序列的每一个突发具有基于DRAM接口突发大小所选定的长度;以及

根据预定的非线性存储器写地址序列,将所述第一多个数据项写入(524、564、816)第二片上存储器,使得所述数据项以与所述第一序列不相同的第二序列排列在所述第二片上存储器上,并且其中,所述第一序列和所述第二序列中的一个包括行列交织的数据,所述第一序列和所述第二序列中的另一个包括行列解交织的数据。

17.根据权利要求16所述的方法,其中,所述数据项块被定义为排列成网格,所述网格包括多个数据项行和多个数据项列。

18.根据权利要求17所述的方法,其中,所述网格还包括多个片区,每一个片区包括所述网格的一个矩形部分,并且还包括R个数据项行和C个数据项列,并且其中,所述第一多个数据项包括一个或多个片区。

19.根据权利要求18所述的方法,其中,根据预定的非线性存储器读地址序列从第一片上存储器读取以第一序列存储的第一多个数据项包括:对于所述第一多个数据项中的每一个片区,

(i)读取位于所述第一片上存储器中的初始起始地址的数据项;

(ii)跳过固定数量的数据项,所述固定数量与所述网格中的行的数量减去一相对应;

(iii)读取数据项;

(iv)重复步骤(ii)和(iii),直到到达该片区的边界为止;

(v)向所述初始起始地址增加一个偏移;以及

(vi)重复步骤(i)-(v),直到已经读取了该片区中的每一个数据项为止。

20.根据权利要求18所述的方法,其中,根据预定的非线性存储器写地址序列将所述第一多个数据项写入第二片上存储器包括:

(i)开始于所述第二片上存储器中针对该片区的初始起始地址,将来自所述第一多个数据项的C个数据项写入所述第二片上存储器中的多个连续地址;

(ii)跳过所述第二片上存储器中的固定数量的地址,所述固定数量与所述网格中的列的数量减去C相对应;

(iii)将来自所述第一多个数据项的C个数据项写入所述第二片上存储器中的多个连续地址;以及

(iv)重复步骤(ii)和(iii)。

21.根据权利要求18所述的方法,其中,将所述第一多个数据项写入(562)所述DRAM包括:

(i)开始于所述DRAM中针对该片区的初始起始地址,将来自所述第一多个数据项的X个数据项写入所述DRAM中的多个连续地址;

(ii)跳过所述DRAM中的固定数量的地址;

(iii)将来自所述第一多个数据项的X个数据项写入所述DRAM中的多个连续地址;以及

(iv)重复步骤(ii)和(iii),其中,X等于所述网格的片区中的数据项的数量。

22.根据权利要求18所述的方法,其中,根据线性地址序列的突发从所述DRAM读取(563)所述第一多个数据项包括:

(i)开始于所述DRAM中的初始起始地址,从所述DRAM中的多个连续地址,读取来自所述第一多个数据项的X个数据项;

(ii)跳过所述DRAM中的固定数量的地址;

(iii)从所述DRAM中的多个连续地址,读取来自所述第一多个数据项的X个数据项;以及

(iv)重复步骤(ii)和(iii),其中,X等于所述网格的片区中的数据项的数量。

用于数字信号处理的基于片区的交织和解交织

背景技术

[0001] 数字信号处理在多种多样的应用程序中获得了使用。这些应用中的很多都是实时的，在该意义上，对于数据的处理存在时间约束，以便其对于终端用户来说是有意义的或者有用的。这方面的一个例子是数字广播流，例如数字电视和数字无线电。数据信号处理系统需要能够足够快速地对实时流进行处理和解码，以便使数据能够像其被接收那样快速地被输出(除非缓冲)。

[0002] 数字信号处理系统除了使用更加通用的数字信号处理器之外，通常还使用一个或多个专用硬件外围设备。这些硬件外围设备是被设计为以快速和高效方式来执行特定的信号处理任务的处理模块。例如，交织和解交织是通常使用硬件外围设备来针对实时数据执行的操作。交织和解交织是存储器密集型的操作，执行该操作的硬件外围设备使用相关联的专用存储器设备对数据进行重新排序。

[0003] 但是，不同类型的实时数据的要求变化非常大。例如，世界范围内使用的各种不同的数字电视和无线标准通常具有不同结构的实时数据，例如，使用不同的类型或者参数进行编码、交织、均衡等等。如果数字信号处理系统能足够灵活地应用于不同的标准，那么用于交织/解交织的专用存储器设备必须足够地大，以便处理具有最大存储器要求的标准。结果，与交织/解交织硬件外围设备一起使用的存储器经常是未充分利用的。

[0004] 存储器设备的一个示例是DRAM(动态随机存取存储器)设备。DRAM设备以页来组织它们所存储的内容，每一页的大小通常为几千字节。每一个DRAM一次只打开有限数量的页(通常为四页)，并且打开一页来存取数据需要许多开销周期。

[0005] 下面所描述的实施例并不限于解决已知数字信号处理系统的任何或者所有缺点的实现。

发明内容

[0006] 提供该概括以便以简化形式来介绍在下面的具体实施方式中进一步描述的构思的精华。该概括并不是旨在标识要求保护的主题的关键特征或者必要特征，也不是旨在用作帮助确定所要求保护的主题的范围。

[0007] 描述了行列交织的数据的基于片区(Tile)的交织和解交织。在一个示例中，将解交织划分成两个存储器传输阶段，第一阶段从片上存储器到DRAM，第二阶段从DRAM到片上存储器。每一阶段都对行列交织的数据块的一部分进行操作，并对这些数据项进行重新排序，使得第二阶段的输出包括解交织的数据。在第一阶段中，根据非线性的存储器读地址序列从片上存储器读取数据项，并将这些数据项写入DRAM。在第二阶段中，根据高效地利用了DRAM接口的线性地址序列的突发，从DRAM读取数据项，并根据非线性的存储器写地址序列，将这些数据项写回到片上存储器。

[0008] 第一方面提供了一种片上数字信号处理系统，其包括：第一存储器，用于存储以第一序列排列的多个数据项，每一个数据项在所述第一存储器上具有相关联的存储器地址，所述多个数据项包括一块数据项的一个子集；第二存储器；耦接到所述第一存储器和所述

第二存储器的传输引擎，其包括去往动态随机存取存储器DRAM的端口，其中所述传输引擎配置为：在第一存储器传输阶段中，将所述多个数据项从所述第一存储器直接传输到所述DRAM，在第二存储器传输阶段中，将所述多个数据项从所述DRAM直接传输到所述第二存储器，其中，在所述第一存储器传输阶段中，所述传输引擎用于：根据预定的非线性的存储器读地址序列，从所述第一存储器读取所述多个数据项，并将所述多个数据项写入到所述DRAM，并且其中，在所述第二存储器传输阶段中，所述传输引擎用于：根据线性地址序列的突发，从所述DRAM读取所述多个数据项，线性地址序列的每一个突发都具有基于DRAM接口突发大小所选定的长度，以及根据预定的非线性的存储器写地址序列，将所述多个数据项写入到所述第二存储器，使得所述多个数据项按照与所述第一序列不相同的第二序列排列在所述第二存储器上，其中所述第一序列和所述第二序列中的一个包括行列交织的数据。

[0009] 第二方面提供了一种在数字信号处理系统中对一块数据项执行交织或解交织操作的方法，该方法包括：根据预定的非线性的存储器读地址序列，从第一片上存储器读取以第一序列存储的第一多个数据项，其中所述第一多个数据项包括该块数据项的一个子集；将所述第一多个数据项写入动态随机存取存储器DRAM；根据线性地址序列的突发，从所述DRAM读取所述第一多个数据项，线性地址序列的每一个突发均具有基于DRAM接口突发大小所选定的长度；以及根据预定的非线性的存储器写地址序列，将所述第一多个数据项写入第二片上存储器，使得所述数据项以与所述第一序列不相同的第二序列排列在所述第二片上存储器上，并且其中，所述第一序列和所述第二序列中的一个包括行列交织的数据。

[0010] 第三方面提供了包括计算机程序代码模块的计算机程序，其中这些计算机程序代码模块用于在所述程序运行在计算机上时，执行上面所描述的方法中的任何一个方法的所有步骤。所述计算机程序可以包含在计算机可读介质上。

[0011] 第四方面提供了执行交织或解交织操作的方法，基本如参照附图中的图5-10中的任何一个所描述的。

[0012] 本申请所描述的方法可以由配置有机器可读形式的软件的计算机执行，其中该软件以例如计算机程序的形式存储在有形存储介质上，计算机程序包括用于配置计算机执行所描述方法的构成部分的计算机程序代码。有形(或非临时性)存储介质的例子包括磁盘、指状驱动器、存储卡等等，其不包括传播的信号。软件可适合于在并行处理器或者串行处理器上执行，使得这些方法步骤可以按照任何适当的顺序执行，或可以同时地执行。

[0013] 本申请承认固件和软件可以是有价值的可单独交易的商品。旨在涵盖在“哑”或标准硬件上运行或对“哑”或标准硬件进行控制以执行期望功能的软件。还旨在涵盖“描述”或定义硬件的配置的软件，例如HDL(硬件描述语言)软件，如用于设计硅芯片，或用于配置通用编程芯片，以执行期望功能。

[0014] 上面的特征可以适当地组合，这对于技术人员将是显而易见的，并且可以与例子的任意方面进行组合。

附图说明

[0015] 通过示例的方式，参照下面的附图来描述实施例，其中：

[0016] 图1描绘了一种数字信号处理系统；

[0017] 图2描绘了传输引擎的示意图；

[0018] 图3示出了用于描绘解交织的各种示例性方法的示意图；

[0019] 图4描绘了使用传输引擎在两块数据上执行的行列操作的示例；

[0020] 图5示出了用于描绘解交织的两个另外示例性方法的示意图；

[0021] 图6描绘了图4的行列操作的示例，其具有抵消DRAM设备的限制的增强措施；

[0022] 图7示出了一种示例性的时间交织的数据块；

[0023] 图8是一种解交织的示例性方法的流程图；

[0024] 图9示出了针对图7中所示的输入交织块而在图8的方法的第一阶段的结束处，在DRAM中存储的数据项的网格表示；以及

[0025] 图10示出了针对图7中所示的输入交织块而在图8的方法的第二阶段的结束处，在片上存储器中存储的数据项的网格表示。

[0026] 贯穿附图使用共同的参考数字来指示类似的特征。

具体实施方式

[0027] 下面仅仅是通过例子的方式来描述实施例。这些例子代表了申请人当前已知的将这些实施例付诸于实践的最佳方式，虽然这些最佳方式并不是能够实现本发明的仅有的方式。描述给出了例子的功能以及用于构造和操作例子的步骤的序列。然而，可以通过不同的例子来实现相同的或等同的功能和序列。

[0028] 下文描述了利用通用数字信号处理器(DSP)以及专用硬件外围设备的数字信号处理系统。为了能高效地使用存储器，该系统的不同元件可以访问共享的片上存储器。诸如直接存储器存取(DMA)控制器之类的传输引擎可以将数据项写入到片上存储器，也可以从片上存储器读取数据项。片上存储器包括静态随机存取存储器(SRAM)，传输引擎还具有去往动态RAM(DRAM)的端口，其中该DRAM可以是在外部的，也可以是片上的。传输引擎具有地址生成元件，地址生成元件使得能够从存储器读取不同的数据项序列，和/或能够将不同的数据项序列写入到存储器，并且这些序列可以包括线性的和非线性的数据项序列。

[0029] 本申请使用的术语“线性”与读取/写入数据项序列有关，其指代读取/写入连续的(或者毗邻的)数据项。相比而言，本申请使用的术语“非线性”与读取/写入数据项序列有关，其指代读取/写入非连续的(或者非毗邻的)数据项，下面将描述非线性序列的示例。

[0030] 下面的描述之中的DRAM的任何使用，旨在覆盖任何形式的DRAM，包括同步DRAM、双倍数据速率(DDR)DRAM(其可以称为DDR RAM)和突发存取DRAM。如上所述，DRAM设备用页来组织它们的存储内容，DRAM设备一次只打开有限数量的页。当存取任何类型的DRAM时，频繁地存取不同的页的数据存取模式是不高效的，这是由于其花费许多开销周期来打开一个页。在突发存取DRAM中，DRAM接口读取/写入4、8、16、32或者64(或更多)个连续字节的突发。使用不完整DRAM接口突发的存取模式也是不高效的。

[0031] 读取/写入不同数据项序列的能力使得能够对数据项即时地执行重新排序操作(例如，交织或者解交织)，同时在存储器单元之间传送这些数据项，或者将这些数据项从一个存储器传送到另一个存储器(例如，在SRAM和DRAM之间传送)。这避免为了结合交织或解交织使用而在数字信号处理系统上包括专用(非共享)存储器的需要，从而减少了芯片面积和费用。可以排列所使用的不同序列，以抵消某些类型的存储器设备的性能限制，例如DRAM(就面积和因此产生的费用而言，与SRAM相比，使用DRAM更便宜，从而可以使用更大的

DRAM)，如下面所更详细描述的。

[0032] 在下文的描述中，只是通过示例的方式使用了时间交织/解交织；但是，应当理解的是，这些方法还可适用于其它形式的交织/解交织，例如比特交织/解交织。

[0033] 首先参见图1，图1示出了一种示例性片上数字信号处理系统100的结构。系统100包括连接到传输引擎106的片上存储器102和DRAM112。存储器设备102、112都用于存储数据项，它们均可以提供共享的存储空间(例如，存储与该数字信号处理系统有关的数据，以及MPEG或者其它与视频流有关的数据)。片上存储器102可以是任何适当形式的RAM，例如(但不限于)SRAM，但不是DRAM。DRAM112可以是片上的，也可以在芯片之外(即，其不能由DSP104进行直接存取)，在下面的描述中，术语‘片上’存储器用于指代片上存储器102(其是非DRAM存储器元件)，尽管事实是DRAM112也可以是片上存储器(即，当其形成在相同的硅片上时是片上系统100的集成部分)。

[0034] 一个或多个DSP104连接到片上存储器102。DSP104是可编程以便于对数据执行信号处理计算(例如，快速傅里叶变换和均衡)的处理器。虽然没有视作为通用处理器，但与下面所描述的硬件外围设备相比，DSP104是更加可配置的。DSP104执行程序代码/指令，以便从片上存储器102读取数据，对该数据执行信号处理操作，并将数据写回到片上存储器102。

[0035] 此外，传输引擎106也连接到片上存储器102，其中传输引擎106为多个硬件(HW)外围设备108提供对于片上存储器102的存取。在一些示例中，传输引擎106可以具有直接存储器存取(DMA)控制器的形式。传输引擎106提供可以由硬件外围设备108使用的多个存储器存取通道(例如，DMA通道)，以实现从片上存储器102读取数据或者向片上存储器102写入数据。

[0036] 如上所述，硬件外围设备108是特殊的、专用固定功能硬件模块，其配置为执行特定的信号处理任务。例如，一个硬件外围设备可以是专用的Viterbi解码模块，另一个可以是专用的Reed-Solomon解码模块。此外，这些硬件外围设备还可以称为加速器。这些硬件外围设备中的每一个彼此之间独立地操作。可以对硬件外围设备进行充分地配置，以便提供特定于它们的任务的操作参数，但不能对它们进行充分地配置来改变它们的任务(例如，不能将Viterbi模块重新配置成Reed-Solomon模块)。因此，与DSP104相比，硬件外围设备更加专用于特定的任务。但是，硬件外围设备被布置为以非常快速和高效的方式来执行它们的特定任务。此外，通用控制处理器110也连接到片上存储器102，其中通用控制处理器110可以用于初始化、配置和控制该数字信号处理系统的操作。

[0037] 上面所描述的数字信号处理系统提供信号处理操作方面的灵活性。例如，该系统可以配置为进行操作，使得不同的DSP104和硬件外围设备108以任何期望的配置或者序列对数据进行处理。每一个硬件外围设备或DSP可以对一块或多块数据(本申请还将其称为数据缓冲区)进行操作，其中所述一块或多块数据由系统的其它部件提供并存储在片上存储器102中，每一个硬件外围设备或DSP生成和存储由该系统的其它元件使用的一个或多个数据缓冲区。这使该数字信号处理系统能用于多种不同类型的信号，例如用于不同的广播/电信标准。

[0038] 使用片上存储器102所提供的公共存储空间，能减少在该片上系统100中提供的存储器存贮的总量。在不使用公共存储空间的情况下，向每一个处理元件提供其自己的专用存储器。例如，每一个DSP104都可以具有它们自己的工作空间存储器，通用控制处理器110

具有用于存储执行代码和数据的另一个单独的存储器，硬件外围设备108具有单独的输入和输出缓冲区，并且可以使用一个或多个另外的存储器来在这些处理元件之间交换数据。

[0039] 由于可对该数字信号处理系统进行配置以便允许实现不同的通信标准，因此针对具体的标准（其中该标准对于任何给定的存储器都具有最大要求），需要对这些单独的存储器中的每一个单独地确定大小。换言之，DSP存储器需要足够的大，以便适应对于DSP存储器具有最大要求的标准。类似地，硬件外围设备缓冲器也需要足够的大，以便适应对于硬件外围设备缓冲器具有最高要求的标准（其与具有较高DSP存储器要求的标准不相同）。结果，非常大量的存储器通常并不被处理元件中的一些使用。

[0040] 但是，如果由片上存储器102提供公共存储空间的话，那么可以将不同标准的存储器需求作为一个整体进行考虑（而不是它们对于系统的各个元件的需求）。换言之，片上存储器102需要足够的大，以便适应这些标准的最大整体、总存储器需求。这具有对标准之间的不同存储器需求进行平均的效果（例如，一种标准可能需要更多的DSP存储器、但更小的缓冲器，而另一种标准可能则是相反的）。这具有只需要显著的更低的总存储器量的效果，并因此节省了硅片面积。

[0041] 因此，片上存储器102所提供的公共存储空间可以保存该系统所使用的所有不同类型的数据，例如，数字信号处理器工作空间、用于通用控制处理器的执行代码和数据、用于硬件外围设备中的一个或多个的输入和输出缓冲区、用于在处理器之间交换数据的一个或多个缓冲区、以及用于该数字信号处理系统的其它配置数据。

[0042] 现参见图2，图2描绘了传输引擎106的示意图。传输引擎106包括第一存储器端口202（其配置为连接到片上存储器102）和第二存储器端口204（其配置为连接到DRAM112）。此外，传输引擎106还包括多个外围设备端口206，每一个外围设备端口配置为连接到相关联的硬件外围设备108。存储器端口202、204和外围设备端口206都连接到交叉线（crossbar）208，其使得这些端口中的任何一个端口能够连接到这些端口中的任何其它端口。

[0043] 此外，传输引擎106还包括地址生成元件210，其耦接到存储器端口202、204，并配置为生成针对连接到存储器端口202、204的存储器中的任意一个或者二者的读和/或写地址序列。在一些示例中，地址生成元件210可以包括可配置的地址生成器，其可以编程为以多种不同的模式（例如，线性模式和非线性模式）进行操作，并且其配置为从一组可能的模式之中选择一个或多个操作模式。在其它示例中，地址生成元件210可以包括：用于生成特定地址序列（例如，将行列模式用于特定的数据项排列的序列，和将突发行列模式用于特定的数据项排列的序列）的一个或多个专用硬件模块。在一些示例中，地址生成元件210可以生成线性和非线性序列，在其它示例中，针对线性序列可以使用直接连接，地址生成元件可以用于只生成非线性序列。

[0044] 通过生成非线性的读和/或写地址序列，地址生成元件210可以对连接到传输引擎106的一个端口的存储器（例如，片上存储器102或者DRAM112）上所存储的数据项执行非线性重新排序。例如，图2描绘了如何在传输到DRAM112期间，对片上存储器102上所存储的第一数据项序列212进行重新排序。在图2的示例中，在片上存储器102上有八个数据项，它们存储在被表示为0到7的存储器地址。在其它示例中，存储器地址可以从不同于零的基地址开始，和/或每一个单独数据项可以比存储器设备上的单个存储单元更大。在该示例中，将这些数据项传输到DRAM112，但按照与第一序列212不相同的第二序列214进行排序。为了清

楚说明起见，在DRAM112上，将第二序列214中的数据项存储在被表示为0’到7’的存储器地址，但在其它示例中，这些地址也可以从不同于零的基地址开始。

[0045] 在第一示例中，地址生成元件210可以生成非线性读序列[3,6,4,1,2,7,0,5]，并将该读序列提供给第一存储器端口202。地址生成元件210还可以生成线性写序列[0’,1’,2’,3’,4’,5’,6’,7’]，并将其提供给第二存储器端口204(其中，只是为了便于说明起见，将DRAM112上的地址表示为0’,1’等等以便与片上存储器102上的地址进行区分)。这使得第一存储器端口202首先从读序列中的第一地址(地址3)读取数据项，在该示例中其是数据项“A”。该数据项穿过交叉线208到达第二存储器204，第二存储器204将该数据项写入到写序列中的第一存储器地址(地址0’)。这导致将数据项“A”从第一序列212中的第四个数据项重新排序成第二序列214中的第一数据项。通过读取读序列中的下一个地址(地址6、地址4等等)，并将相应的数据项(B、C、...)写入到写序列中的下一个地址(地址1’、地址2’、...)，来重复该操作。作为该操作的结果，来自第一序列的数据项(其被表示为G、D、E、A、C、H、B、F)现在按照第二序列(A、B、C、D、E、F、G、H)存储在DRAM上。

[0046] 在第二示例中，通过地址生成元件210生成线性读序列[0,1,2,3,4,5,6,7]和非线性写序列[6’,3’,4’,0’,2’,7’,1’,5’]，也可以实现对数据项的相同的重新排序。在该示例中，首先从片上存储器上的地址0读取数据项“G”，并将其写入到DRAM上的地址6’，其后从片上存储器上的地址1读取数据项“D”，并将其写入到DRAM上的地址3’，等等。类似地，在第三示例中，通过地址生成元件210生成非线性读序列以及也是非线性的写序列，也可以实现对数据项的相同的重新排序。这种方式的一个示例可以是读序列[0,2,4,6,1,3,5,7]和写序列[6’,4’,2’,1’,3’,0’,7’,5’]。

[0047] 在上面的每一个示例中，在传输引擎106从片上存储器102向DRAM112直接传输数据项期间，即时地执行从第一序列到第二序列的重新排序。对于从DRAM112到片上存储器102的传输，或者从片上存储器到该片上存储器中的另一个位置的传输，以及类似地对于从DRAM到DRAM中的另一个地址的传输，也可以执行相似的操作。

[0048] 此外，上面的示例还示出了在执行传输之前，完全地生成读地址序列和写地址序列。但是，也可以与传输一起并发地执行地址序列的这种生成，例如，通过在对一个或多个之前数据项进行读/写时生成一个或多个读地址和写地址。

[0049] 上面所描述的过程，作为向DRAM112进行的存储器传输操作的组成部分，使得片上存储器102上的数据项能够被重新排序成不同的序列，并且类似地，作为向片上存储器102进行的存储器传输操作的一部分，DRAM112上的数据项可以被重新排序成不同的序列。这可以用于例如通过使用地址生成元件210来实现交织或解交织，其中地址生成元件210配置为根据某种交织方案来生成读/写地址序列。

[0050] 图3示出了描绘解交织的各种示例性方法的示意图。在第一示意图300中，在从片上存储器102到片上存储器102的单个传输中，执行解交织。在后续的两个示意图302、304中，存在两个传输：一个是从片上存储器102到DRAM112的传输，第二个是从DRAM返回到片上存储器102的传输。在第二示意图302中，可以通过根据线性写序列将存储在片上存储器102上的数据项写入到DRAM112，随后使用特定的非线性序列从DRAM112读回这些数据项(其可以称为‘行列模式’或‘行列交织’)，来执行这些数据项的解交织。下面参照图4来详细地描述这种非线性序列。替代地，可以通过使用行列模式将数据项写入DRAM112，随后线性地读

回这些数据项，来执行数据项的解交织，如图3中的第三示意图304所示。

[0051] 在图3中所示的所有实现中，在所有交织的数据都位于输入存储器（即，图3中的每一个图的左手方所示出的片上存储器102）之前，这种解交织过程不能开始。

[0052] 行列模式考虑将数据项排列在具有多个行和列的一个或多个网格或表中。在图4中对其进行了描绘，其中图4示出了具有从0到23的连续存储器地址（只是为了说明起见）的第一块的输入数据项402，以及具有从24到47的连续存储器地址（同样只是为了说明起见）的第二块的输入数据项404。如果参照图3中的第二示例302来描述行列模式，则这些存储器地址位于DRAM112中。在图4所示的示例中，考虑这些数据项每隔六个数据项具有列中断，如图4中的虚线所指示的。这意味着将这些连续存储器地址被考虑为沿着具有六个行的网格的列进行排列（这可以描述成沿着这些列来写入/读取数据）。

[0053] 在图4中示出了以网格形式呈现的数据项，图4示出了第一块的输入数据项402的第一网格406和第二块的输入数据项404的第二网格408。第一网格和第二网格均具有六个行和四个列。应当注意的是，连续寻址的数据项是沿着列进行排列的。但是，在其它示例中，也可以将数据项呈现为：相反地，连续的项是沿着行进行排列的，在该情况下，下面的描述仍然适用，但参照倒置的行和列。

[0054] 行列模式的目的是对每一个网格进行转置，使得当以穿过网格的列的序列来排列（例如，来自于DRAM112的）输入数据项时，以穿过网格的行的序列来排列输出数据项（例如，作为到片上存储器102的输出）。例如，参见网格406，如果输入数据序列的前四个数据项是A、B、C、D（沿着第一列读取四个项），那么输出数据序列的前四个数据项是A、G、M、S（沿着第一行读取四个项）。因此，诸如该方式的行列操作改变了数据项的顺序，这其取决于定义了多少个行存在于网格中。

[0055] 为了实现该行列模式，地址生成元件210生成用于导致行列转置的读和写序列。这可以通过生成非线性读序列（例如，从DRAM112读）和线性写序列（如图4中所示，下面将更详细地进行描述），或者通过生成线性读序列（例如，从片上存储器102读）和非线性写序列（例如，如图3中的第三示例304所示）来实现。在另外的示例中，为了实现高效的存储器存取，还可以使用非线性读序列和非线性写序列，如下面参照图6所描述的。

[0056] 图4示出了非线性读序列410的示例，可以观察到其包括非连续的存储器地址。在一个示例中，可以使用下面的伪代码所描绘的算法，来生成该地址序列：

```
N0 = rows * columns;

N1 = rows;

N2 = numBlocks * rows * columns;
```

[0057]

```
For ind = 1 to numItems

   nextItemAddr = a + o;

   a = a + N1;
   if a >= N0

      a = a - N0 + 1;

     b = b + 1;

      if b >= N1

         a = 0;
```

[0058]

```
         b = 0;

         o = rem(o + N0, N2);

      end

  end

end
```

[0059] 其中“rows”(N1)是网格中的行的数量(在图4的示例中,其是6),“columns”是网格中的列的数量(在图4的示例中,其是4),“numBlocks”是数据项的块的数量(在图4的示例中,其是2),“numItems”是所有块上的数据项的全部数量(在图4的示例中,其是48)。变量“a”、“b”和“o”是在该算法中使用的内部变量,它们可以都被初始化为零,或者一个或多个可以被初始化为非零值以便应用偏移量。

[0060] 在计算了N0(网格中的行的数量)、N1(行的数量与列的数量相乘)和N2(行的数量、列的数量和数据项的块的数量的乘积)的初始值之后,该算法迭代的次数为所给出的数据项的数量,在每一次迭代计算该序列中的下一个地址(“nextItemAddr”)。实际上,该算法跳过了输入序列中的固定数量的数据项(例如,在图4中是6个),直到到达一行的结束位置为止(由第一个“if”语句来确定),随后将该行的起始点递增1,并进行重复。第二个“if”语句

检测一个块的结束，该“if”语句对计算进行重置，但增加通过求余运算(rem(.))所计算的偏移量(在图4中，其是24)。随后，重复该过程，直到到达“numItems”为止。应当注意，可以将“numItems”设置为比所给出的数据项的总数更大的值，并且如果是这样的话，则一旦访问了所有的块，该算法就卷绕回到第一块。

[0061] 在图4中示出了通过上面的算法所生成的读序列410，其中最上面一行示出了用于第一块(网格406)的序列，最下面一行示出了用于第二块(网格408)的序列。将读序列410的前四个项作为例子，它们从地址0、6、12、18进行读取，地址0、6、12、18对应于来自输入数据项402的数据项A、G、M、S。可以观察到，其与网格406的第一行相对应。

[0062] 地址生成元件210生成具有连续的存储器地址的线性写序列412，使得当传输引擎106使用读序列410和写序列412时，以非线性序列读取数据项，并以线性序列写入这些数据项。应当注意，为了简单起见，图4中的写序列具有从0到47的地址，但在其它示例中，这些地址可以从任何基地址开始。在第一输出数据项块414和第二输出数据项块416中，可以观察到读序列410和写序列412的组合的结果。通过将这些输出数据项与网格406和408进行比较，可以观察到，已经成功地执行了行列操作。

[0063] 通过生成线性读序列和非线性写序列(例如，如图3中的第二示例304所示)，也可以获得相同的结果，如下所示(为了简洁起见，只示出了第一块)：

[0064] 读序列：

[0065]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23

[0066] 写序列：

[0067]

0	4	8	12	16	20	1	5	9	13	17	21	2	6	10	14	18	22	3	7	11	15	19	23

[0068] 可以使用与上面详细描述的非线性读序列相类似的技术，来生成非线性写序列。上面的示例描绘了可以如何使用地址生成元件210来实现对一组数据项的交织/解交织操作，如行列交换。

[0069] 图5示出了描绘解交织的两个另外示例性方法的示意图502、506，其中这两种方法在与DRAM112进行交互方面更加高效。这两种方法都将时间解交织过程划分成两个存储器到存储器的传输操作(片上存储器到DRAM，随后DRAM到片上存储器)，每一个操作都使用至少一个非线性的地址序列。此外，这两种方法还都使用行列(R-C)模式(如上面参照图4所描述的，箭头521和561)和突发行列(B R-C)模式(箭头523、524和562—564)的组合。

[0070] 虽然图5示出了将数据项从片上存储器102传输到DRAM112，随后再传输回片上存储器102，但应当理解的是，可以替代地将这些数据项写回与初始从其读取这些数据项的片上存储器不相同的片上存储器，或者将这些数据项写回到该片上存储器102的不同部分。举例而言，(在箭头521和561所指示的操作中)可以从片上存储器102的一部分(其可以称为片区缓冲区)读取数据项，随后(在箭头524和564所指示的操作中)将这些数据项写回(以解交织形式)到片上存储器102的不同部分(其可以称为解交织器输出缓冲区)。这两个缓冲区可以具有不同的大小。在下面的描述中，任何提及从相同片上存储器102传输数据项并随后将数据项传输回相同片上存储器102，只是示例的方式，并且所描述的方法还可以用于将数据项从一个片上存储器元件(通过DRAM)传输到另一个片上存储器元件，或者从片上存储器102的一个部分(通过DRAM)传输到该片上存储器102的另一个部分。

[0071] 突发行列模式可以视作为上面所描述的行列模式的一个变型，或者替代地，行列模式可以视作为突发行列模式的、突发长度为一的特定实例。突发行列模式考虑数据排列在具有行和列的网格中(如上所述)；但是，当沿着行穿过时，不是只从每一个列读取一个数据项(如在行列情况中)，突发行列模式在沿着行跳到下一个列之前(即，通过跳过r-L个数据项，其中r=该网格中的行的数量)，读取预定数量的连续地址(其中该预定数量称为‘突发长度’L)。例如，参见图4的网格406，如果突发长度为3，则突发行列模式首先在一个突发中读取三个连续项(项A、B、C)，随后移动到下一列，读取接着的三个连续项(G、H、I)，接着是M、N、O并且随后是S、T、u。随后，卷绕回第一列，并读取D、E、F，接着读取J、K、L等等。因此，可以将突发行列模式视作为与行列模式相同，除了读取一组连续的数据项而不是仅仅一个数据项之外，或者替代地，可以将行列模式视作为突发长度等于1的突发行列模式。

[0072] 在一个示例中，可以使用通过下面的伪代码所描绘的算法，来生成用于突发行列模式的读序列：

```
N1 = rows;

N3 = burstLength;

N4 = rows * columns - burstLength;


For ind = 1 to numItems

   nextItemAddr = a + o;

   a = a + 1;

   if a >= N3

[0073]        a = 0;

              o = o + N1;

              if o >= N4 + N1

                 o = 0;

              elseif o >= N4 + N3

                 o = o - N4;

              end

     end

end
```

[0074] 如上面在行列模式的描述中所阐述的，对该伪代码中的变量进行定义。此外，“突发长度”(N3)是要在每一个突发中读取的连续项或毗邻项的数量，N4是行的数量(N1)和列的数量的乘积减去N3。应当注意，还可以用类似的方式来生成用于突发行列操作的写序列。

[0075] 突发行列模式可以用于使得能够用某些类型的存储器设备(例如，DRAM112)来高效地执行解交织操作，特别是在B R-C模式中的突发长度(L)与DRAM接口突发大小相同或者接近的情况下。通过用此方式基于DRAM接口突发大小来选择突发长度(或突发大小)(或者根据下面所描述的其它示例)，这高效地利用了DRAM接口。相比而言，很多传统的解交织器存取模式尝试连续地读取/写入间隔很大的数据项，在使用DRAM设备的情况下这导致了不高效存储器存取，这是由于不完整(DRAM接口)突发和很多DRAM页的交叉引起的。

[0076] 例如，图4的行列操作读取连续的数据项，其中这些数据项被间隔开网格中的行的

数量。在存在很大数量的行的示例中，这会导致在该存储器设备中进行间隔很大的存取，从而造成从不同的DRAM页进行不高效的存取。返回到图3中所示的示例，可以观察到，在第二示例302中，以行列模式从DRAM读取是不高效的，在第三示例304中，以行列模式向DRAM写入也是不高效的。

[0077] 图6描绘了一种解交织操作的例子，其中该操作不会引发与频繁地存取不同的页或者部分地填充突发相关联的DRAM存取的低效性。在图5中的第一示意图502里，也示出了该示例。图6的示例生成与图4相同的行列结果(即，具有6行、4列和2个块的交换)，但是使用线性顺序存储器存取的多次运行来生成来获得该结果，其导致类似DRAM的分页设备的高效操作。在图6的示例中，传输引擎从片上存储器102读取输入数据项序列，在DRAM112上存储这些数据项，随后从DRAM112中读回这些数据项，并将它们写入到片上存储器102(潜在地覆盖它们的原始位置)，其中行和列进行了交换。

[0078] 为了解释的目的，输入数据项602与图4的示例中所使用的数据项相同。总共存在48个数据项，它们具有从零开始的连续的存储器地址序列。首先，以每一个块或片区具有六行和两列的行列模式，从片上存储器102读取这些数据项。如图6中所示，可以考虑将这些数据项排列在片区604中，每一个片区具有六行和两列。本申请使用的这种片区大小只是示例，在很多实现中，可以将片区大小设置为等于DRAM接口突发大小。用于依次地沿着这些片区中的每一个片区的行进行读取(即，使用行列模式)的非线性读序列606是由地址生成元件210生成的，如上所述的。此外，线性写序列608也是由地址生成元件210生成的。传输引擎106使用非线性读序列606从片上存储器102进行读取(图5中的箭头521)，使用线性写序列608向DRAM112进行写入(箭头522)。用此方式向DRAM进行写入是高效的，这是由于其线性地向连续的地址进行写入，因此如果数据项的数量充足，则只会偶尔地跨越DRAM页边界。

[0079] 作为该操作的结果，可以观察到DRAM112上的数据项610与来自片区604的行列交换相对应。随后，地址生成元件210生成用于从DRAM112读回这些数据项的非线性读序列612。使用突发行列模式来生成该读序列，将该读序列配置为避免低效的存取。在该示例中，突发行列模式使用每突发6个项、12行和2列。由于DRAM读序列612读取数据项的突发，所以这些数据项位于DRAM上的连续地址，并且因此不太可能跨越页边界，并且还能高效地利用DRAM接口上的可用的突发(特别是如果地址生成器突发长度L接近DRAM接口突发大小)。因此，相对于(非突发)行列模式，将跨越明显更少的页边界。

[0080] 此外，还生成非线性写序列614，以便将这些数据项写回到片上存储器102。该写序列614也是使用突发行列模式来生成的，并且在该示例中，使用每突发两个项、四个行和三个列。读序列612(图5中的箭头523)和写序列614(箭头524)的组合使得与执行具有六行、四列和两个块的基本行列操作相比，写回到片上存储器102的输出数据项616处于相同的序列(这可以与图4进行比较)，除了在DRAM112上存储该数据以外，而不会引起由于页边界和不完整的突发所造成的低效。此外，由于从片上存储器102进行的初始读取(图5中的箭头521)使用只具有两列的片区的行列操作，因此这使得一个整个片区一到达片上存储器102就能开始向DRAM传输数据，其与图4中使用四个列的块相比更加快速。这可以提高实时数据情形下的性能，其中在该情形下，数据随着时间以流的方式到达。

[0081] 图7-10描绘了另一种解交织操作的示例，其中该解交织操作不会引起与存取不同的页相关联的DRAM存取的低效性。在图5的第二示意图中，也示出了该方法。如通过图5所观

察到的，该方法只涉及在一个突发之中与DRAM112的线性交互，即使用突发行列模式来向DRAM进行写入（箭头562）和从DRAM进行读取（箭头563）。如上所述，这意味着将跨越很少的页边界，并且高效地使用DRAM接口突发，这提高了该方法的整体效率。

[0082] 仅仅为了说明的目的，该方法考虑将数据项排列在具有多个行和列的一个或多个网格或表中（如在先前的示例之中），并且该方法还使用片区的概念，其中片区是由行列结构的一组数据形成的。如下所述，可以根据DRAM接口突发或者页大小，来确定片区的大小。应当理解的是，存储器中的数据是存储在连续的存储单元中的。

[0083] 图7示出了包括200个数据项（被标记了地址0-199）的示例性时间交织的数据块700，其中这200个数据项排列在10个片区702（T_0-T_9）之中，每一个片区包括20个项。在DRAM112是突发模式存取的DRAM的情况下，可以对片区大小进行选择以便与DRAM接口突发大小相匹配，并且由于存储器传输（如下所述）高效地利用了DRAM接口，因此这进一步提高了该方法的效率。如果片区大小与DRAM接口突发大小不匹配，则片区大小可以替代地小于DRAM接口突发大小，或者每一片区可以有多个突发。在多个示例中，在片区大小不是精确地与DRAM接口突发大小匹配的情况下，将这些片区与DRAM中的页边界进行对齐，这可以在DRAM接口效率方面提供显著的改善。如下面所更详细描述的，由于在片区缓冲区中存储至少一个完整的片区之前，该方法不能开始，因此片区大小的选择对于片上RAM片区缓冲区（即，从其读取数据的片上存储器102）的大小设置了约束。

[0084] 应当理解的是，虽然图7中的示例性时间交织的块700包括200个数据项，但这些块可以明显地大于与该数值，并且可以包括数千个数据项。此外，时间交织的块之中的行和列的排列可以由使用该方法的系统来设置。

[0085] 将该示例中的解交织过程划分成几个阶段的存储器到存储器传输，其中每一次传输（或者‘片区工作（tiling job）’）传输多个片区，如参照图8中所示的流程图可以解释的。图8中所示的方法在每一次片区工作中传输N个片区，并且可以对N的值进行选择，以便等于片区的列（例如，在图7所示的示例中，N=2）。但是，在其它示例中，片区工作可以包括片区的多个列（例如，一个以上的列），以便减少所需要的片区工作的数量。仅仅是为了解释的目的，将参照图7中所示的时间交织的数据块700并且在N=2的情况下来描述图8中所示的方法。在片区工作包括片区的一个以上的列的示例中，该方法如下所述地进行操作，并且仅仅改变地址生成器的配置（即，告诉地址生成器处理更多的数据）。

[0086] 一旦在片上存储器102中存储了来自时间交织的块的最小值N个片区（即，至少N个片区），例如，一旦在片上存储器102中存储了片区T_0和T_1，该方法就可以开始（方框802）。如上所述，存储了这些交织的片区T_0和T_1的片上存储器102的一部分，可以称为片区缓冲区，并且在第一存储器到存储器传输阶段81对N个片区进行操作时，可以仅将该片区缓冲区的大小调整为能够存储数据的N个片区。在一个例子中，片区缓冲区可以是弹性缓冲区，其大小可以以某种方式根据系统吞吐量、可用的存储器带宽和DRAM接口进行调整以顾及一个或多个片区工作。

[0087] 使用行列模式从片上存储器102读取第一片区T_0（方框804和图5中的箭头561）。因此，所使用的用于该第一片区的非线性读序列为：

[0088]

0	10	20	30	1	11	21	31	2	12	22	32	3	13	23	33	4	14	24	34

[0089] 其中，上面的数字与片上存储器中的数据项的地址相对应，如图7中所示。返回参见行列模式的早期描述(具体而言，所提供的伪代码示例)，可以观察到，读取数据项(即，一个数据项)，并且随后在读取另一个数据项之前，跳过接着的9个数据项。重复该过程，直到读取了总共4个数据项为止(一个片区中的列的数量)，随后在偏移一个数据项的情况下，重复整个过程(即，读取地址1，之后读取地址11)，以此类推等等，直到读取了整个片区为止。

[0090] 随后，使用突发行列模式将该数据项序列写入到DRAM112(方框806和箭头562)，其中突发长度L等于一个片区中的数据元素的数量(例如，L=20)：

[0091]

0’	1’	2’	3’	4’	5’	6’	7’	8’	9’	10’	11’	12’	13’	14’	15’	16’	17’	18’	19’
0	10	20	30	1	11	21	31	2	12	22	32	3	13	23	33	4	14	24	34

[0092] 其中，第一行与DRAM中的数据项的地址相对应，被标记为0’-19’，以便将它们与从其读取这些数据项的片上存储器102中的原始地址进行区分，在第二行中示出了这些原始地址。

[0093] 随后，重复这两个操作(方框804中的读操作和方框806中的写操作)，直到将所有N个片区都写入到DRAM中为止(方框808中的‘是’)。在该阶段，在将N个片区写入到DRAM之后，可能已经从片上存储器102读取了所有存储的数据项，并且在该情况下，可以使用来自时间交织的块的其它N个片区的数据项来对该片上存储器进行重新填充(方框810)。或者，当在片上存储器中已经存储有另外的片区(例如，至少N个另外的片区)时，该方法可以继续读取另外的片区(在方框804中)，将它们写入到DRAM(在方框806中)，而无需对该片上存储器进行重新填充(即，省略方框810)。

[0094] 对第一阶段81进行重复，直到从片上存储器102读取了整个时间交织的块700，并将其写入到DRAM为止(方框812中的‘是’)，其中根据需要对片上存储器102进行重新填充(在方框810中)。在该示例中，将存在五次传输，每一次传输两个片区(这是由于N=2，块700包含10个片区)。

[0095] 图9示出了针对图7中所示的时间交织的块700在第一阶段81结束时在DRAM中存储的数据项的网格表示902(其由块700中的原始地址来指代)。在该网格902旁边的是第二网格904，第二网格904标识了DRAM122中的每一个数据项的地址(其被标记为0’-199’，以便将它们与片上存储器102中的原始地址0-199进行区分)。在该网格表示中，对原始的片区进行了重新排序(与块700相比而言)，但没有解交织，并且来自一个片区的重新排序的数据项占据了连续的存储器地址(例如，T_0存储在地址0’-19’中)。如通过图9所观察到的，该网格包括40行和5列，使得数据项的每一列(其中将连续的数据项排列在列中)包括两个片区。虚线906标出了位于一个列之中的片区之间的边界。

[0096] 在该方法的第二阶段82中，将数据项传输回片上存储器102(或者传输回另一个片上存储器元件，如上所述)，并且使用另外的重新排序操作来完成数据的解交织。使用突发行列模式从DRAM112读取第一片区T_0(方框814和图5中的箭头563)，其中突发长度L再次等于一个片区中的数据元素的数量(在该示例中，L=20)，即读序列是：

[0097]

0’	1’	2’	3’	4’	5’	6’	7’	8’	9’	10’	11’	12’	13’	14’	15’	16’	17’	18’	19’
0	10	20	30	1	11	21	31	2	12	22	32	3	13	23	33	4	14	24	34

[0098] 其中，第一行与DRAM112中的数据项的地址相对应，第二行示出了从其读取了这些

数据项的片上存储器102中的原始地址。

[0099] 随后，使用突发行列模式将该片区T_0写入到片上存储器102(方框816和箭头564)。该突发行列模式使用突发长度L，其等于原始时间交织的块700中的一个片区里的列的数量，例如，在图7所示的示例中，其是四。因此，将数据写入到片上存储器中的四个连续的地址，跳过接着的16个地址(原始时间交织的块中的列的数量=转置块中的行的数量=20，20-4=16)，随后将数据写入到接着的四个连续的地址，以此类推。因此，该非线性写序列是：

[0100]

0″	1″	2″	3″	20″	21″	22″	23″	40″	41″	42″	43″	60″	…	80″	81″	82″	83″
0	10	20	30	1	11	21	31	2	12	22	32	3	…	4	14	24	34

[0101] 其中，第一行与要对其进行写的片上存储器中的地址相对应，这些地址被标记为0″、1″等等，以便将它们与在第一阶段81中从其读取这些数据项的片上存储器102中的原始地址进行区分，在第二行中示出了这些原始的地址。

[0102] 应当注意的是，在向DRAM进行写和从DRAM进行读的前两次突发行列操作(箭头562和563)中使用的突发长度，使用相同的突发长度(例如，L=20)，并且向片上存储器进行写的这种第三次突发行列操作(箭头564)使用不同的突发长度(例如，L=4)。

[0103] 随后，逐片区地(并使用与第一阶段81相同的片区大小)对第二阶段82进行重复，直到将所有的片区都写入到片上存储器102为止(方框818中的‘是’)。

[0104] 图10示出了针对图7中所示的输入时间交织的块700在第二阶段82结束时在片上存储器中存储的数据项的网格表示1002(其通过块700中的原始地址来指代)。在该网格1002旁边的是第二网格1004，第二网格1004标识了片上存储器102中的每一个数据项的地址(其被标记为0″-199″，以便将它们与片上存储器102中的原始地址0-199和DRAM112中的地址0’-199’进行区分)。在该网格表示中，对原始的数据项进行了解交织，如通过图10所观察到的，使得第一片区T_0现在包括四行和五列(而不是如数据块700中的五行和四列)，如虚线轮廓所示出的。如通过图10所观察到的，用于一个解交织的块的网格包括20行和10列。

[0105] 应当理解的是，虽然图7、9和10示出了从基地址0开始的地址，但在其它示例中，地址也可以从任何基地址开始。

[0106] 通过上面的解释和图8可以观察到，该方法中的读/写工作对多个片区(例如，一个或多个片区)而不是整个时间交织的块进行操作。这使该方法能针对具体的DRAM接口突发大小进行优化，例如，可以将一个片区设置为与一个DRAM接口突发具有相同的大小，随后片区工作将是DRAM接口突发的整数倍(例如，在上面参照图7、9和10所描述的示例中，其是2)。由DRAM接口所规定的DRAM接口突发大小，可以按照该DRAM中的页或者子页水平进行设置，并且将取决于总线带宽，并且可以被设置使得一个突发的起始与一个页的起始对齐，并且可能时突发完全地在一个页中完成(以防止由于存储器分页而造成的低效)。如上所述，当片区大小没有与DRAM接口突发大小准确地匹配时，或者其是DRAM接口突发大小的倍数时，这些片区可以替代地与页边界对齐，以便提高DRAM的效率，代价是未使用的DRAM容量。

[0107] 虽然上面的描述和图8示出了串行地执行该方法(即，在完成第一阶段81之后，执行第二阶段82)，但该方法的方面可以并行地执行，使得可以从SRAM中读取来自一个时间交织的块的片区，并将其写入到DRAM中(在第一阶段81中)，同时可以从DRAM中读取来自另一个时间交织的块的片区，并将其写入到SRAM中(在第二阶段82中)。这允许进行存储器再利用，这是由于向DRAM进行写的操作(方框806)可以使用与在第二阶段82中进行读取时(方框

814)相同的地址集,只要时序使得在使用来自另一个时间交织的块的数据项来重写一个特定的地址(在方框806中)之前对该地址进行读取(在方框814中)即可。

[0108] 图8中所示出和上面所描述的方法将对于数据项的网格进行转置(以便执行解交织)的操作,划分成两个单独的部分。当从SRAM中进行读取(方框804和图5中的箭头561)并向DRAM进行写入(方框806和箭头562)时,执行该转置的第一部分,当从DRAM中进行读取(方框814和箭头563)并向SRAM写回(方框816和箭头564)时,执行该转置的第二部分。所有这些转置都使用非线性的地址序列;但是,使用了不同的非线性序列。在第一部分中,将行列模式用于从SRAM进行读取(突发长度=1),在第二部分中,将突发行列模式用于向SRAM进行写入(突发长度=一个片区中的列的数量)。与DRAM的交互(方框806中的写和方框814中的读)使用的突发行列模式所具有的突发长度等于一个片区中的数据元素的数量(例如,在图7-10所示的示例中,L=20)。

[0109] 上面参照图5(示例506)和图7-10所描述的方法高效地使用了可用的DRAM(具体而言,突发存取的DRAM)带宽,这是由于使用了涉及数据的片区(而不是整个时间交织的块)传输的多阶段过程,其中片区大小是根据DRAM接口突发大小进行选择的。片区的配列特定于具体的实现,上面所描述的方法可以应用于片区的任何排列以及每一片区任意数量的数据项。

[0110] 例如,当在DVB-T2中使用该方法时,可以将一个列中的片区的数量(N)设置为等于前向纠错(FEC)块的数量,使得图7-10中所示的示例可以与存在两个FEC块的场景相对应。在其它示例中,可以存在三个FEC块,所以N=3,并且在一次片区工作中,从SRAM向DRAM传输三个片区,并将它们写入到DRAM中的连续地址。

[0111] 上面所描述的方法,解交织过程,被划分成几个阶段。使用所描述的方法,在开始解交织过程之前,不需要在片区缓冲区中存储整个的交织数据块。如参照图8所描述的,在该方法开始之前,只需要在片区缓冲区中存储有N个片区就可以。

[0112] 上面参照图5(示例506)和图7-10所描述的方法可以使用如图2中所示的地址生成元件210来实现。该地址生成元件210可以进行配置,或者可以包括专用硬件逻辑电路,其配置为生成所需要的(预定的)非线性地址序列,其中该非线性地址序列在所述方法的具体实现中使用(例如,用于特定的片区的排列)。

[0113] 上面所描述的方法可以用于对任何交织的数据块进行解交织。示例性应用包括OFDM信号,具体而言,是诸如DVB-T2之类的数字地面电视(DTT)信号;但是,这些方法并不限于OFDM、DTT或者DVB-T2。上面所描述的方法还可以用于对数据进行交织,以形成交织的数据块。为了将上面所描述的方法用于交织,而不是解交织,这些方法步骤保持不变,差别在于:输入数据(例如,如在方框802中所存储的)包括解交织的数据(而不是交织的数据),输出数据(例如,如在图8的结束时写回到SRAM的)包括交织的数据(而不是解交织的数据)。

[0114] 本文中使用术语“处理器”和“计算机”来表示具有处理能力使得其能够执行指令的任意设备。本领域技术人员将认识到这样的处理能力被并入到许多不同的设备,并且因此,术语“计算机”包括机顶盒、媒体播放器、数字电台、PC、服务器、移动电话、个人数字助理和许多其它设备。

[0115] 本领域技术人员将认识到用于存储程序指令的存储设备可以分布在网络上。例如,远程计算机可以将所描述的过程的例子存储成软件。本地或终端计算机可以访问远程

计算机，并下载所述软件的一部分或全部以运行程序。可替换地，本地计算机可以根据需要下载软件的片段，或者在本地终端执行一些软件指令并且在远程计算机（或计算机网络）执行一些软件指令。本领域技术人员还将认识到，通过使用本领域技术人员已知的常规技术，软件指令的全部或一部分可以由专用电路、可编程逻辑阵列等来执行。

[0116] 对于“逻辑”的具体引用指代执行某种功能或者一些功能的结构。逻辑的示例包括：用于执行这些功能的电路。例如，这种电路可以包括晶体管和/或在制造过程中可用的其它硬件元件。举例而言，这种晶体管和/或其它元件可以用于形成实现和/或包含存储器（例如，寄存器、触发器或锁存器）、逻辑运算器（例如，布尔运算）、算术运算器（例如，加法器、乘法器或移位器）和互连的电路或结构。可以将这些元件提供成定制电路或者标准单元库、宏或者位于其它层次的抽象。可以用特定的排列对这些元件进行互连。逻辑可以包括是固定功能的电路，可以对电路进行编程以执行某种功能或者一些功能；可以通过固件或者软件更新或者控制机制来提供这种编程。被标识为执行一种功能的逻辑还可以包括：实现组分功能或者子过程的逻辑。在一个示例中，硬件逻辑具有实现固定的功能操作或者一些操作、状态机或过程的电路。

[0117] 本文给出的任意范围或设备值可以被扩展或者被修改，而不丧失所请求的效果，这对于技术人员将是显而易见的。

[0118] 将理解的是，上面描述的益处和优点可以涉及一个实施例，或者可以涉及若干实施例。这些实施例并不限于解决所陈述问题中的任何一个或全部的实施例或者具有所陈述益处和优点中的任何一个或全部的实施例。

[0119] 对“一”项的任何提及是指这些项中的一个或多个。本文中使用术语“包括”来表示包含所标识的方法方框或元件，但是这样的方框或元件并不包括排他列表，并且装置可以包含额外的方框或元件，并且方法可以包含额外的操作或元素。

[0120] 本文描述的方法的步骤可以以任何适当的顺序执行，或者在适当的情况下同时执行。此外，可以从任意方法中删除单独的方框，而不脱离本文描述的主题的精神和范围。上面描述的任意例子的方案可以与所描述的任意其它例子的方案组合，以形成其它例子，而不丧失所请求的效果。

[0121] 将理解的是，仅仅通过例子的方式给出了优选实施例的以上描述，并且本领域技术人员可以做出各种修改。虽然上面以某种具体度或者参考一个或多个单独实施例描述了各个实施例，但是本领域技术人员可以对所公开的实施例做出大量的改变，而不脱离本发明的精神或范围。

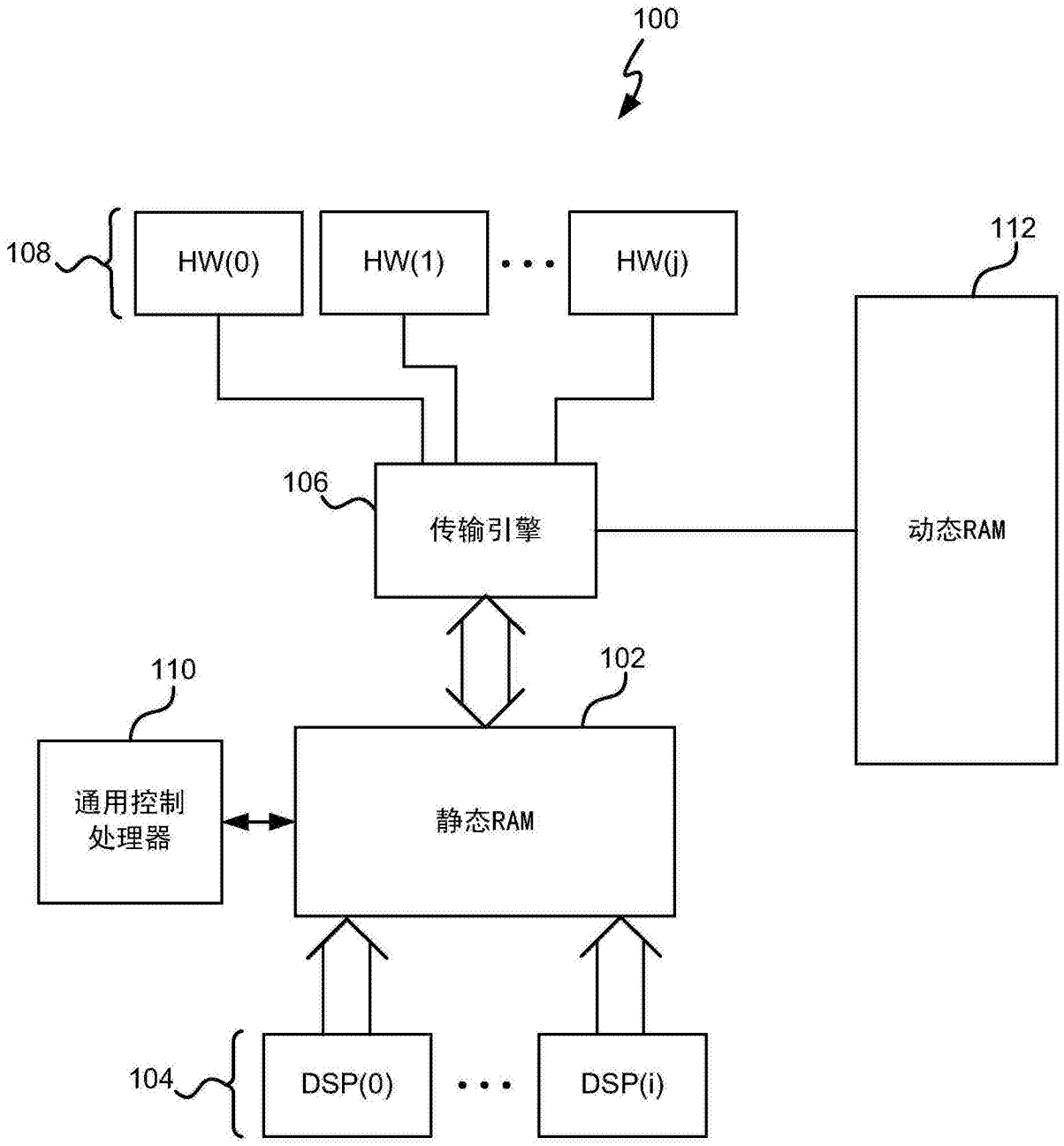

100
108
HW(0)
HW(1)
HW(j)
112
106
传输引擎
动态RAM
110
102
通用控制
处理器
静态RAM
104
DSP(0)
DSP(i)

图1

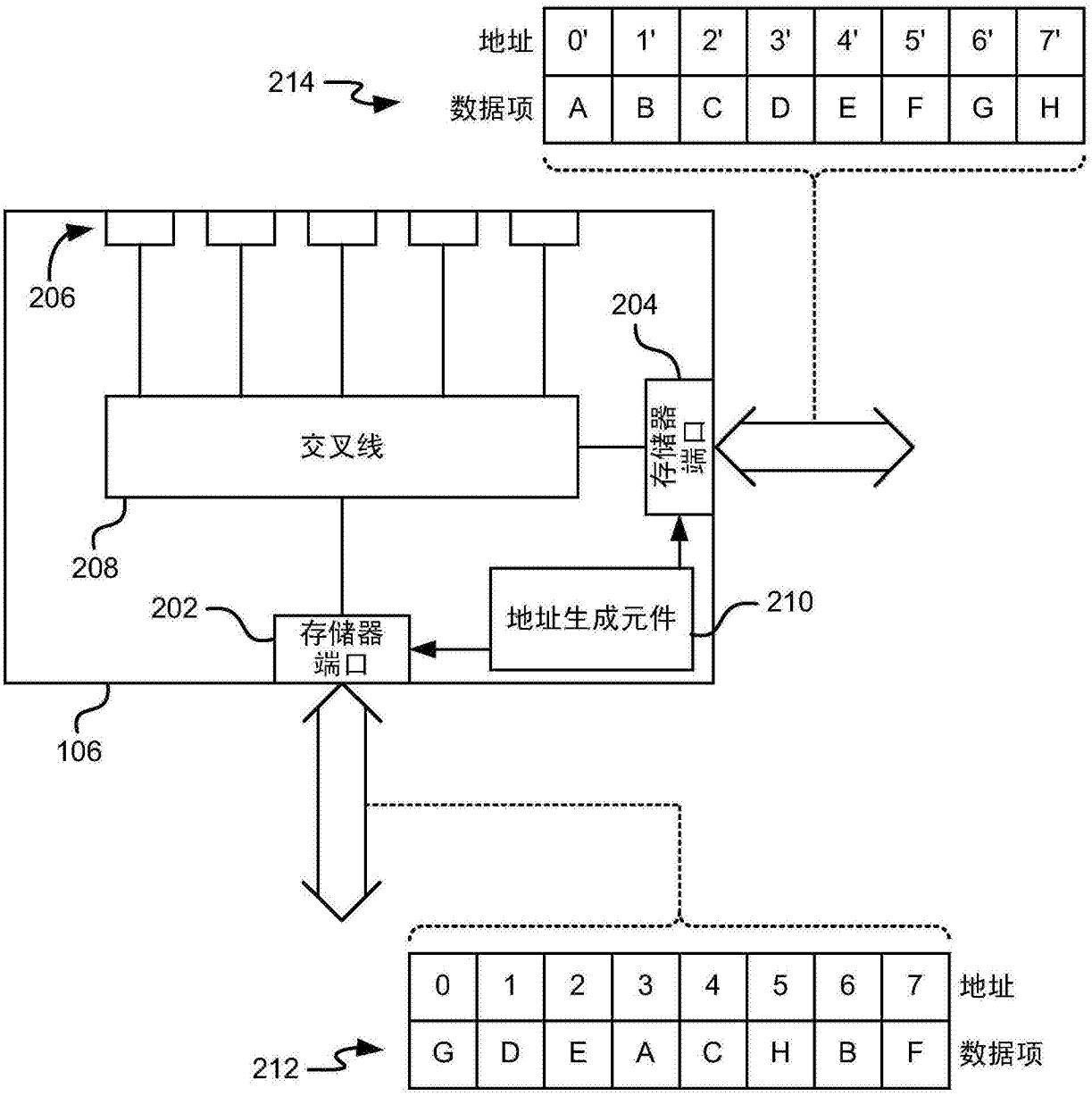

地址
0'
1'
2'
3'
4'
5'
6'
7'
214
数据项
A
B
C
D
E
F
G
H
206
204
交叉线
存储器端口
208
202
存储器端口
地址生成元件
210
106
0
1
2
3
4
5
6
7
地址
212
G
D
E
A
C
H
B
F
数据项

图2

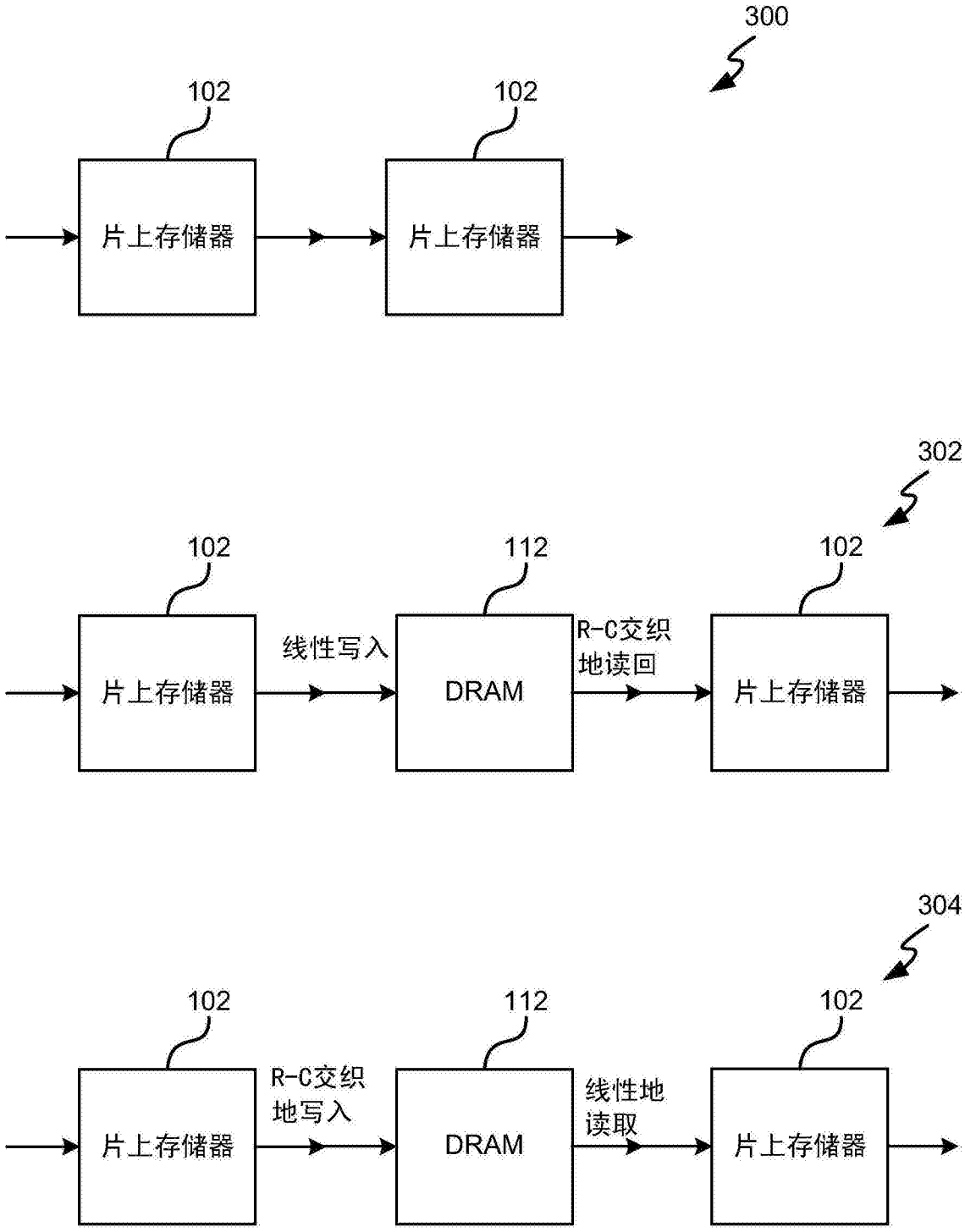

300
102
102
片上存储器
片上存储器
302
102
112
102
片上存储器
线性写入
DRAM
R-C交织
地读回
片上存储器
304
102
112
102
片上存储器
R-C交织
地写入
DRAM
线性地
读取
片上存储器

图3

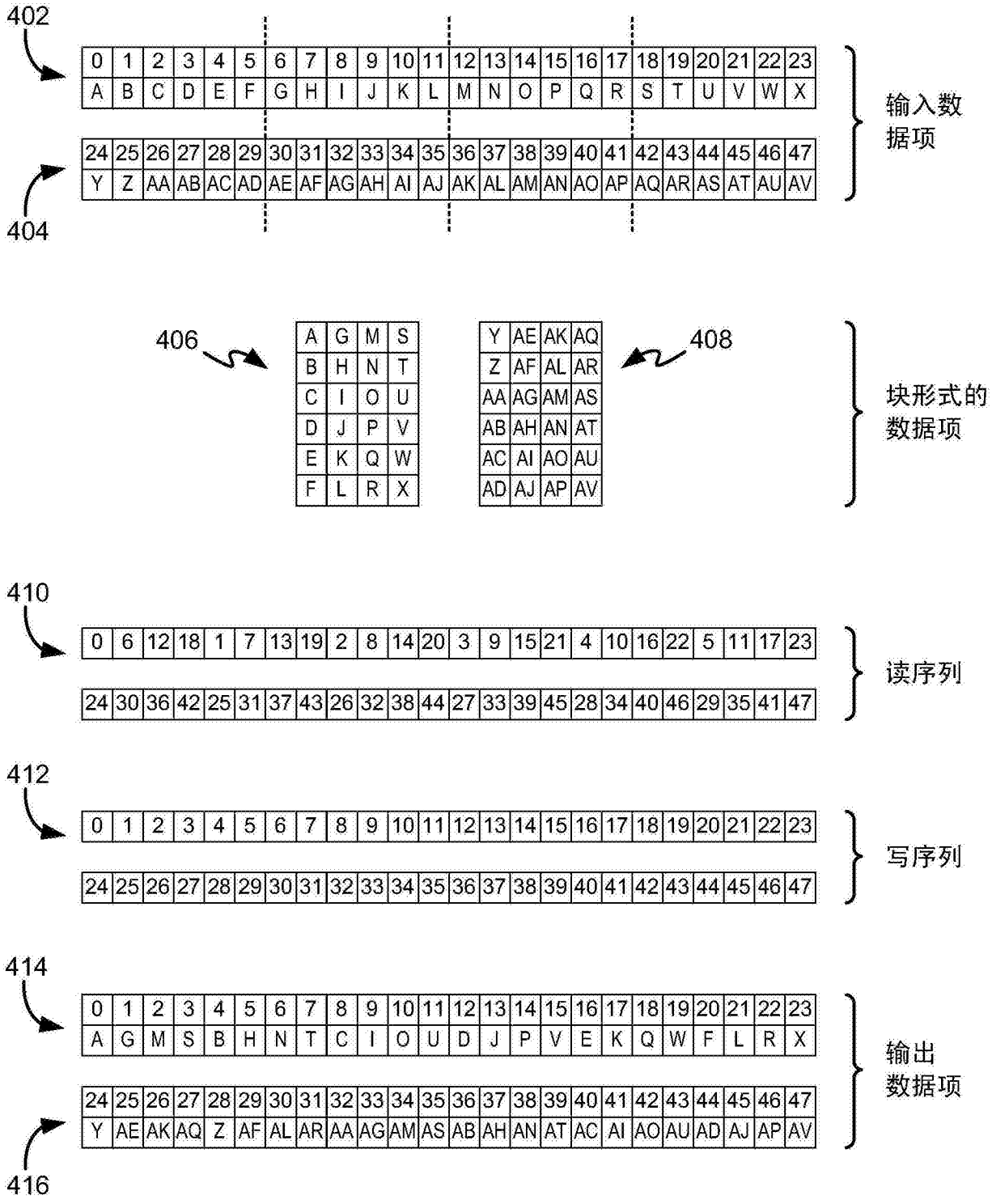
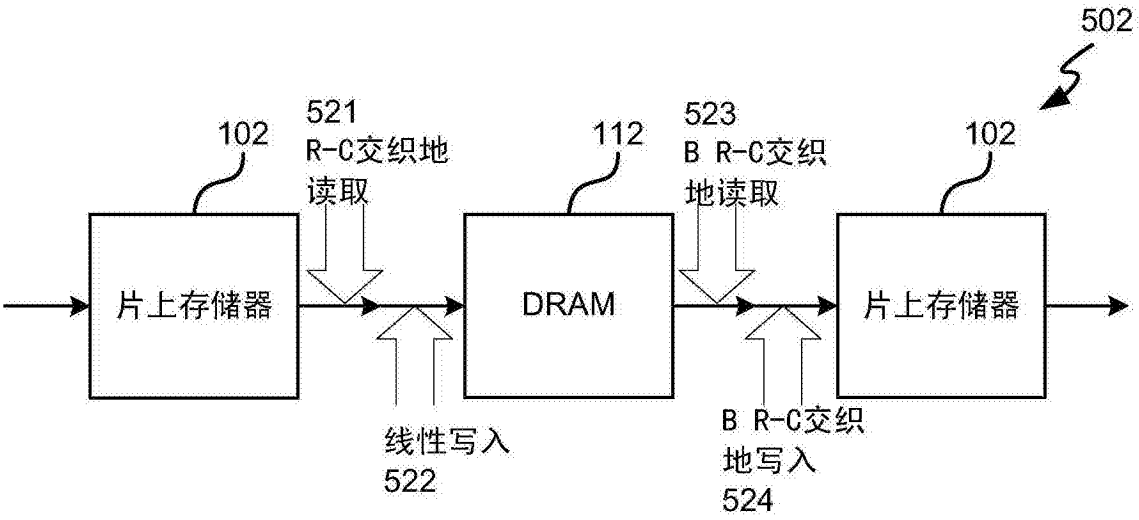

402
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
A B C D E F G H I J K L M N O P Q R S T U V W X
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
Y Z AA AB AC AD AE AF AG AH AI AJ AK AL AM AN AO AP AQ AR AS AT AU AV
404
输入数据项
406
A G M S
B H N T
C I O U
D J P V
E K Q W
F L R X
Y AE AK AQ
Z AF AL AR
AA AG AM AS
AB AH AN AT
AC AI AO AU
AD AJ AP AV
408
块形式的数据项
410
0 6 12 18 1 7 13 19 2 8 14 20 3 9 15 21 4 10 16 22 5 11 17 23
24 30 36 42 25 31 37 43 26 32 38 44 27 33 39 45 28 34 40 46 29 35 41 47
读序列
412
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
写序列
414
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
A G M S B H N T C I O U D J P V E K Q W F L R X
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
Y AE AK AQ Z AF AL AR AA AG AM AS AB AH AN AT AC AI AO AU AD AJ AP AV
416
输出数据项

图4


502
102
521
R-C交织地
读取
112
523
B R-C交织
地读取
102
片上存储器
DRAM
片上存储器
线性写入
522
B R-C交织
地写入
524

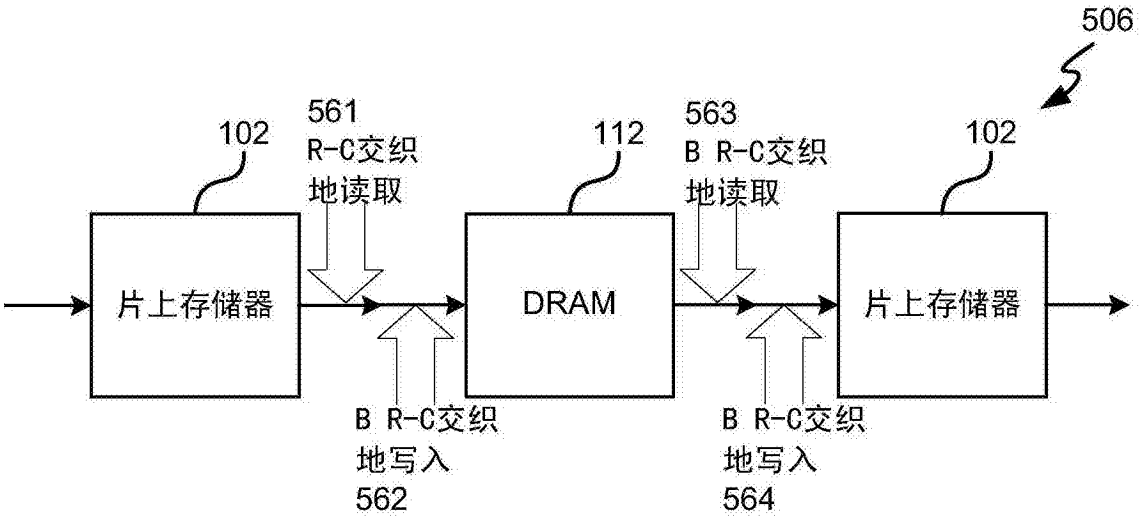

506
102
561
R-C交织
地读取
112
563
B R-C交织
地读取
102
片上存储器
DRAM
片上存储器
B R-C交织
地写入
562
B R-C交织
地写入
564

图5

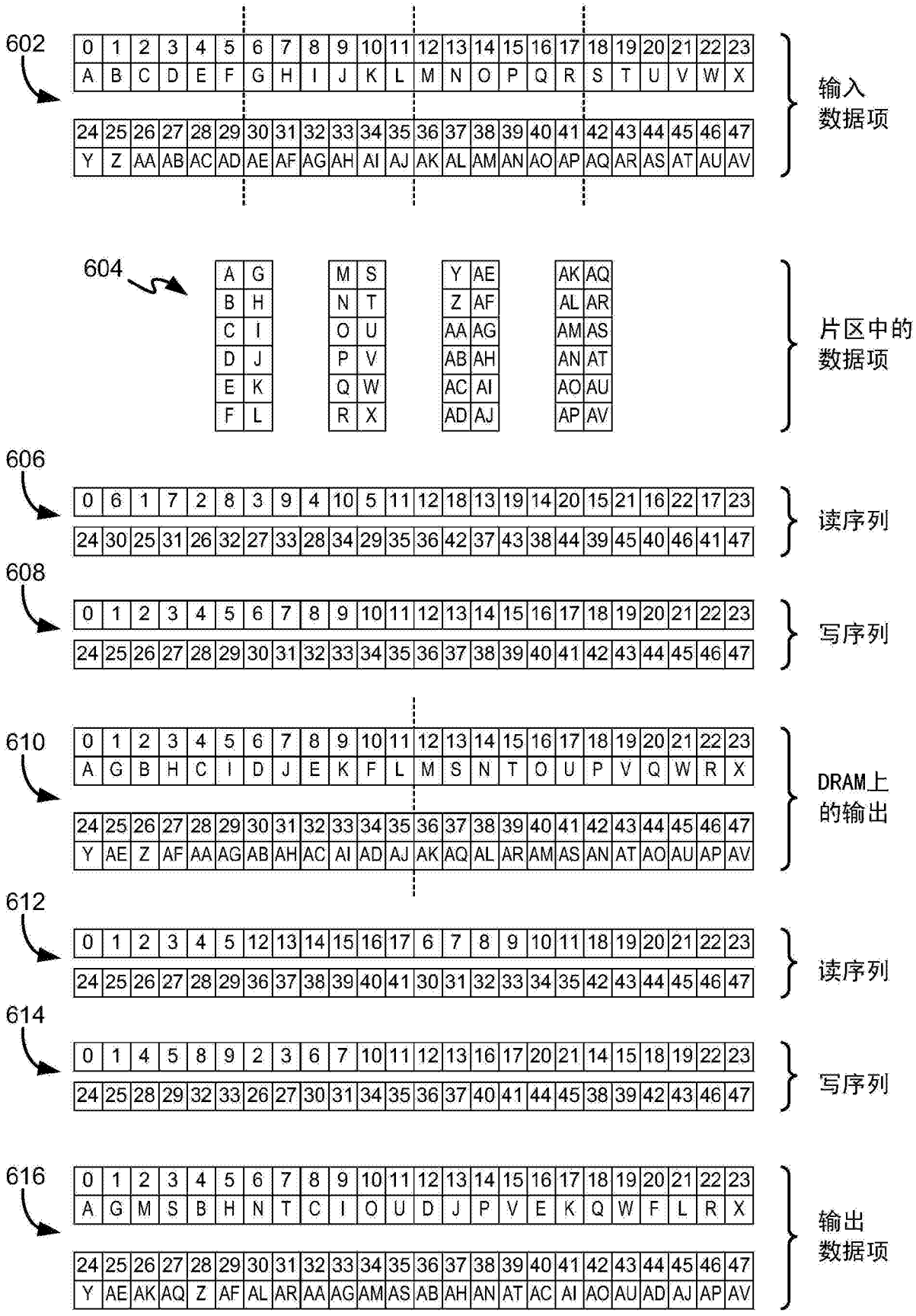

602
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
A B C D E F G H I J K L M N O P Q R S T U V W X
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
Y Z AA AB AC AD AE AF AG AH AI AJ AK AL AM AN AO AP AQ AR AS AT AU AV
输入数据项
604
A G
B H
C I
D J
E K
F L
M S
N T
O U
P V
Q W
R X
Y AE
Z AF
AA AG
AB AH
AC AI
AD AJ
AK AQ
AL AR
AM AS
AN AT
AO AU
AP AV
片区中的数据项
606
0 6 1 7 2 8 3 9 4 10 5 11 12 18 13 19 14 20 15 21 16 22 17 23
24 30 25 31 26 32 27 33 28 34 29 35 36 42 37 43 38 44 39 45 40 46 41 47
读序列
608
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
写序列
610
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
A G B H C I D J E K F L M S N T O U P V Q W R X
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
Y AE Z AF AA AG AB AH AC AI AD AJ AK AQ AL AR AM AS AN AT AO AU AP AV
DRAM上的输出
612
0 1 2 3 4 5 12 13 14 15 16 17 6 7 8 9 10 11 18 19 20 21 22 23
24 25 26 27 28 29 36 37 38 39 40 41 30 31 32 33 34 35 42 43 44 45 46 47
读序列
614
0 1 4 5 8 9 2 3 6 7 10 11 12 13 16 17 20 21 14 15 18 19 22 23
24 25 28 29 32 33 26 27 30 31 34 35 36 37 40 41 44 45 38 39 42 43 46 47
写序列
616
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
A G M S B H N T C I O U D J P V E K Q W F L R X
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
Y AE AK AQ Z AF AL AR AA AG AM AS AB AH AN AT AC AI AO AU AD AJ AP AV
输出数据项

图6

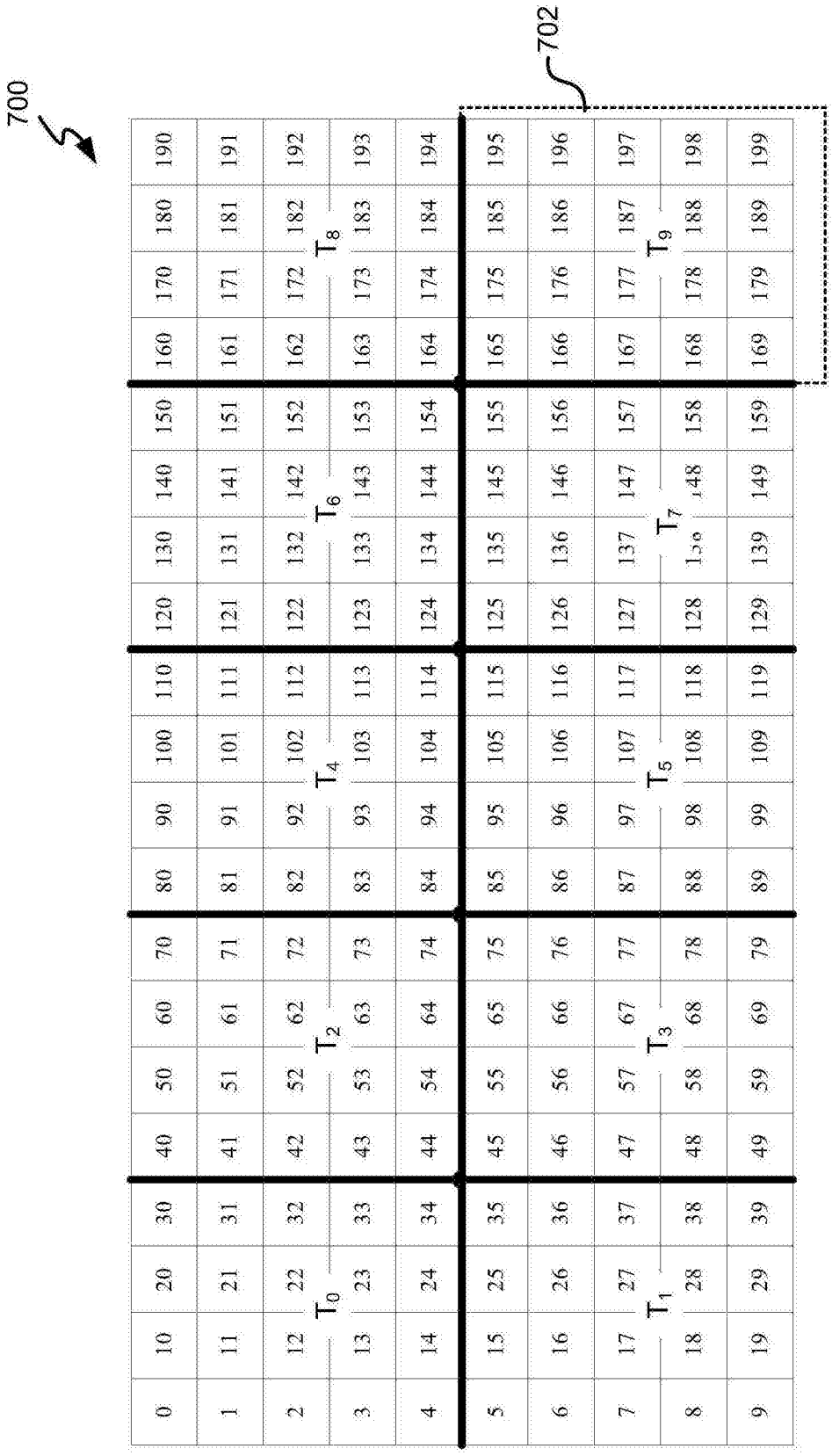


图7

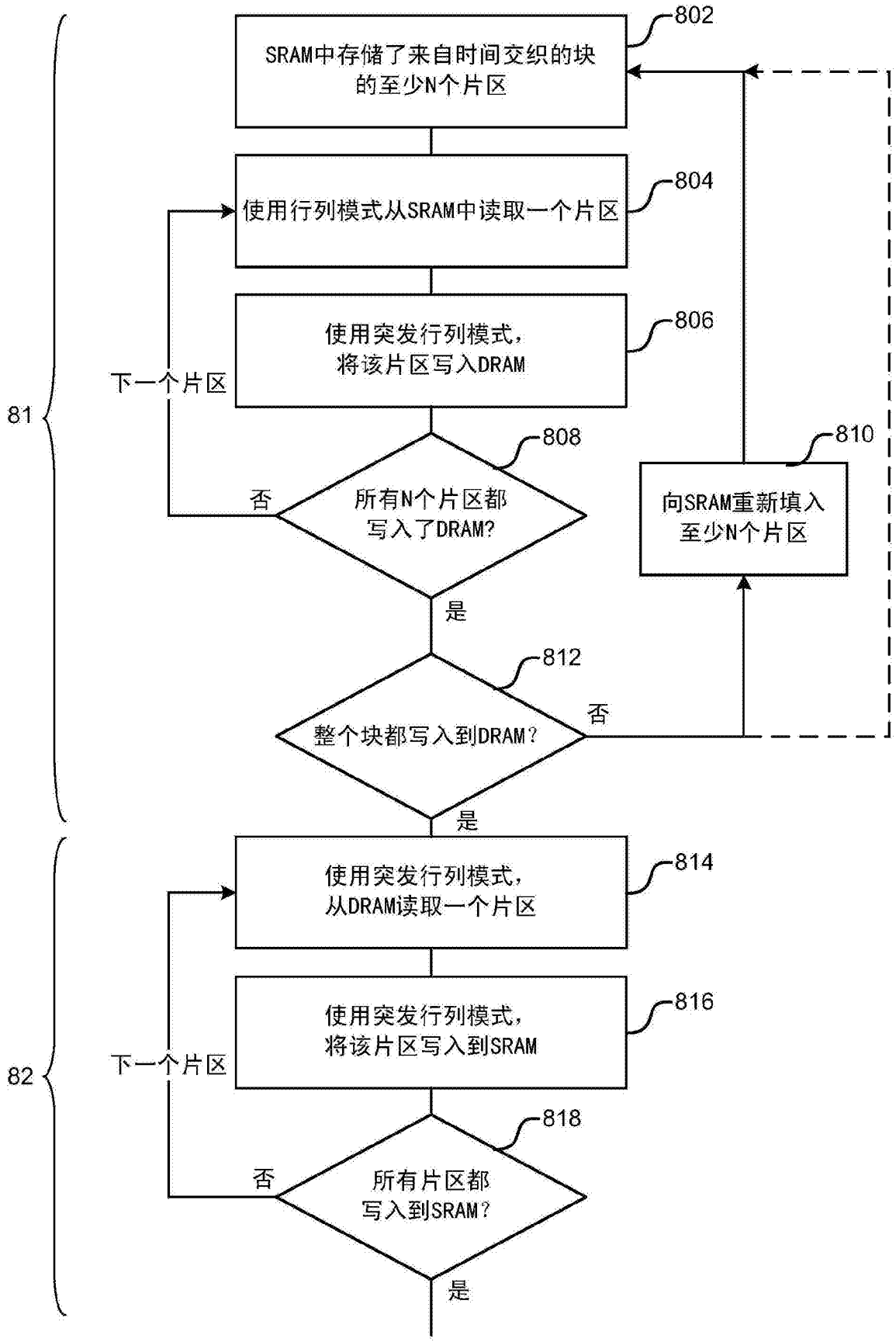
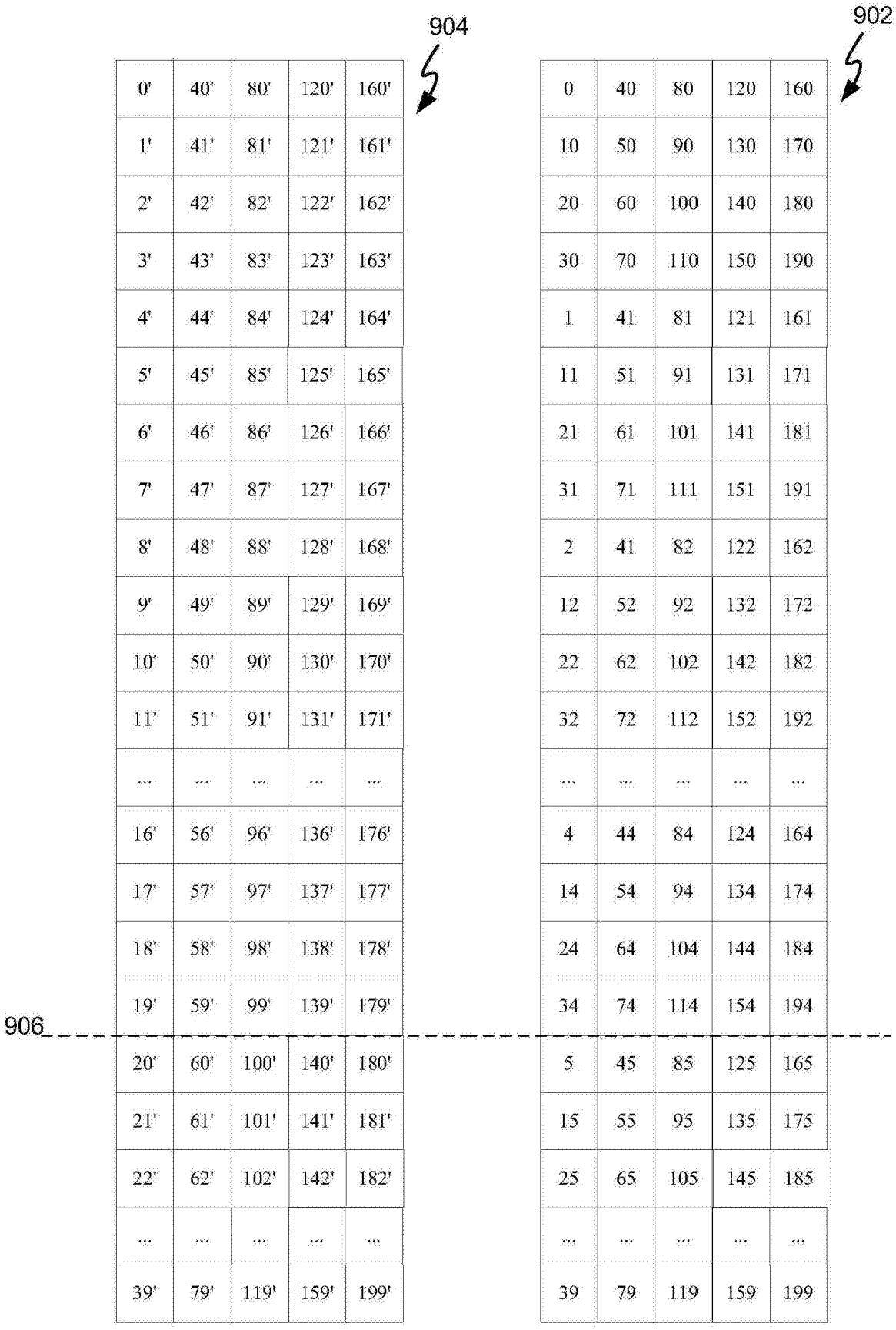

802
SRAM中存储了来自时间交织的块的至少N个片区
804
使用行列模式从SRAM中读取一个片区
806
使用突发行列模式，将该片区写入DRAM
下一个片区
81
808
否
所有N个片区都写入了DRAM?
是
810
向SRAM重新填入至少N个片区
812
否
整个块都写入到DRAM？
是
814
使用突发行列模式，从DRAM读取一个片区
816
使用突发行列模式，将该片区写入到SRAM
下一个片区
82
818
否
所有片区都写入到SRAM？
是

图8



0'	40'	80'	120'	160'
1'	41'	81'	121'	161'
2'	42'	82'	122'	162'
3'	43'	83'	123'	163'
4'	44'	84'	124'	164'
5'	45'	85'	125'	165'
6'	46'	86'	126'	166'
7'	47'	87'	127'	167'
8'	48'	88'	128'	168'
9'	49'	89'	129'	169'
10'	50'	90'	130'	170'
11'	51'	91'	131'	171'
...	...	...	...	...
16'	56'	96'	136'	176'
17'	57'	97'	137'	177'
18'	58'	98'	138'	178'
19'	59'	99'	139'	179'
20'	60'	100'	140'	180'
21'	61'	101'	141'	181'
22'	62'	102'	142'	182'
...	...	...	...	...
39'	79'	119'	159'	199'

0	40	80	120	160
10	50	90	130	170
20	60	100	140	180
30	70	110	150	190
1	41	81	121	161
11	51	91	131	171
21	61	101	141	181
31	71	111	151	191
2	41	82	122	162
12	52	92	132	172
22	62	102	142	182
32	72	112	152	192
...	...	...	...	...
4	44	84	124	164
14	54	94	134	174
24	64	104	144	184
34	74	114	154	194
5	45	85	125	165
15	55	95	135	175
25	65	105	145	185
...	...	...	...	...
39	79	119	159	199

图9

0"	20"	40"	60"	80"	100"	...	180"
1"	21"	41"	61"	81"	101"	...	181"
2"	22"	42"	62"	82"	102"	...	182"
3"	23"	43"	63"	83"	103"	...	183"
4"	24"	44"	64"	84"	104"	...	184"
5"	25"	45"	65"	85"	105"	...	185"
6"	26"	46"	66"	86"	106"	...	186"
7"	27"	47"	67"	87"	107"	...	187"
8"	28"	48"	68"	88"	108"	...	188"
9"	29"	49"	69"	89"	109"	...	189"
10"	30"	50"	70"	90"	110"	...	190"
11"	31"	51"	71"	91"	111"	...	191"
...	...	...	...	...	...	...	...
16"	36"	56"	76"	96"	116"	...	196"
17"	37"	57"	77"	97"	117"	...	197"
18"	38"	58"	78"	98"	118"	...	198"
19"	39"	69"	79"	99"	119"	...	199"

1004

1002

T_0

0	1	2	3	4	5	...	9
10	11	12	13	14	15	...	19
20	21	22	23	24	25	...	29
30	31	32	33	34	35	...	39
40	41	42	43	44	45	...	49
50	51	52	53	54	55	...	59
60	61	62	63	64	65	...	69
70	71	72	73	74	75	...	79
80	81	82	83	84	85	...	89
90	91	92	93	94	95	...	99
100	101	102	103	104	105	...	109
110	111	112	113	114	115	...	119
...	...	...	...	...	...	...	...
160	161	162	163	164	165	...	169
170	171	172	173	174	175	...	179
180	181	182	183	184	185	...	189
190	191	192	193	194	195	...	199

图10