



(12)发明专利

(10)授权公告号 CN 103888224 B

(45)授权公告日 2017.05.10

(21)申请号 201410136945.6

(22)申请日 2014.04.04

(65)同一申请的已公布的文献号

申请公布号 CN 103888224 A

(43)申请公布日 2014.06.25

(73)专利权人 中国科学院自动化研究所

地址 100190 北京市海淀区中关村东路95号

(72)发明人 吴军宁 王晓琴 张森 赵旭莹

林啸 郭晓龙 郭璟 王伟康

(74)专利代理机构 中科专利商标代理有限责任

公司 11021

代理人 宋焰琴

(51)Int.Cl.

H04L 1/00(2006.01)

(56)对比文件

CN 102412850 A,2012.04.11,

CN 102577205 A,2012.07.11,

US 2013/0311852 A1,2013.11.21,

审查员 刘梅

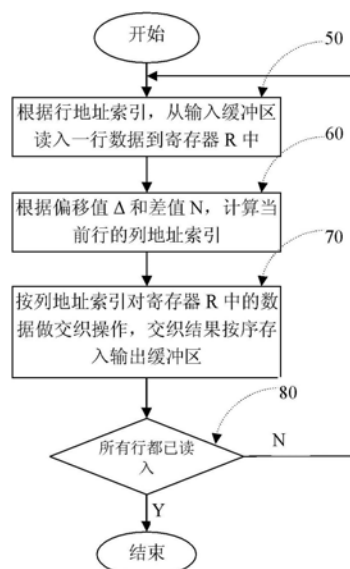
权利要求书1页 说明书6页 附图3页

(54)发明名称

一种用于LTE系统Turbo码内交织的并行实现方法及装置

(57)摘要

本发明提出并公开了一种通过查表以及简单的移位及算术运算操作,并行实现Turbo码内交织的方法及装置,将数据按列优先的方式存入输入缓冲区中,每次从输入缓冲区中按行地址索引读入一行数据,再根据列地址索引进行交织操作,实现Turbo码内交织。通过分析LTE物理层协议36.212中所规定的188种不同码块长度,总结出以下规律:行地址索引在不同码块长度的情况下前后均存在等差特性,同时在计算列地址索引时,每行数据的列地址索引均与首行相差偏移量 Δ 和差值N。本发明能够明显减少Turbo码内交织器的计算时间,实现较快地Turbo编译码,提高无线通信系统的数据吞吐率,可广泛应用于4G基站以及移动设备中。



1. 一种并行实现Turbo码内交织的方法,其特征在于,将数据按列优先的方式存入一个输入缓冲区中,然后进行多次置换操作,所述每次置换操作是从该输入缓冲区中按行地址索引读入一行数据,再根据列地址索引进行交织操作,实现Turbo码内交织。

2. 根据权利要求1所述的并行实现Turbo码内交织的方法,其特征在于,所述输入缓冲区的行数R和列数C根据以下规则进行计算:

$$\begin{cases} R=8, 40 \leq K \leq 512 \\ R=16, 528 \leq K \leq 1024 \\ R=32, 1056 \leq K \leq 2048 \\ R=64, 2112 \leq K \leq 6144 \end{cases}$$

$C=K/R$,其中K为码块长度。

3. 根据权利要求1所述的并行实现Turbo码内交织的方法,其特征在于,将不同码块长度K对应的列置换时所需的列地址索引存放在一张表格中,在进行第一次置换操作时,所述列地址索引是根据码块长度K并通过查表得到。

4. 根据权利要求3所述的并行实现Turbo码内交织的方法,其特征在于,除首行之外的行的列地址索引是由首行的列地址索引根据偏移量 Δ 和差值N计算得到。

5. 根据权利要求3所述的并行实现Turbo码内交织的方法,其特征在于,得到第一行的列地址索引T0以及每行相对于第一行的偏移量 Δ 和差值N后,根据偏移量 Δ 对T0进行循环移位操作,并根据差值N完成加法操作,得到除首行之外的行的列地址索引。

6. 根据权利要求4所述的并行实现Turbo码内交织的方法,其特征在于,除首行之外的行的列地址索引是根据不同码块长度K统计得到的。

7. 根据权利要求4或5所述的并行实现Turbo码内交织的方法,其特征在于,统计得到的针对与不同码块长度K的偏移量和差值。

8. 根据权利要求1所述的并行实现Turbo码内交织的方法,其特征在于,所述Turbo码内交织的实现方式的并行度由数据存入输入缓冲区的列数C和交织单元的位宽L决定,其并行度P的计算公式为 $P=C/\text{ceil}(C/L)$, $\text{ceil}()$ 表示对数据进行向上取整操作。

9. 一种并行实现Turbo码内交织的装置,其特征在于,包括行列地址生成单元、全交织单元、输入缓冲区,

所述输入缓冲区存放需要做交织运算的输入信息流;

所述行列地址生成单元完成两部分工作:其一,将不同码块长度K对应的列置换时所需的列地址索引存放在一张表格中,在进行第一次置换操作时,所述列地址索引是根据码块长度K并通过查表得到;其二,计算每行输入缓冲区中待译码的数据的列地址索引,完成第二次置换所需的交织索引;

所述全交织单元根据所述行列地址生成单元传递的列地址索引,对每一行数据进行全交织运算。

10. 如权利要求9所述的并行实现Turbo码内交织的装置,其特征在于,还包括输出缓冲区,其用于存储所述全交织单元的输出数据。

一种用于LTE系统Turbo码内交织的并行实现方法及装置

技术领域

[0001] 本发明涉及通信技术领域,特别是涉及一种用于LTE系统Turbo编译码内交织的并行实现方法及装置。

背景技术

[0002] 随着科技的日益发展,无线通信技术得到了广泛应用。但是用户数据在经过空中接口传输时,会受到各种噪声以及信号的干扰,致使信号出错或者丢失。为了提高信号传输质量,增加无线通信系统的可靠性,通常需要通信系统具有一定的检错纠错的能力。

[0003] 在长期演进(Long Term Evolution,LTE)系统中,广泛采用的一种纠错码就是Turbo码。Turbo码诞生于1993年,由Berrou等人在卷积码和级联码的基础上首次提出,其特征是编译码复杂度高,时延大,但误码性能优异,适合于大数据量的长码块及对时延要求不高的数据传输。它能很好地满足香农信道编码理论中的随机性条件,并且采用了迭代译码的方式来获得编码增益,因此能够获得逼近香农限的极限性能。

[0004] 同时,为了改善移动通信的传输特性,降低信号衰落带来的影响,均会采用交织编码技术。交织编码技术就是将造成数字信号传输的突发性差错,利用交织编码技术进行离散化处理并纠正该差错。图1所示即为3GPP LTE物理层协议36.212规定的Turbo编码器的结构图,由两个分量编码器和一个内交织器组成,其中内交织器在Turbo码中发挥着关键作用,直接影响Turbo码的译码性能。LTE采用的内交织器,是二次置换多项式(Quadratic Polynomial Permutation,QPP)交织器,具有“最大无争用”特性,其表达式为

[0005]
$$\Pi(i) = (f_1 \cdot i + f_2 \cdot i^2) \bmod K$$

[0006] 其中, $i=0,1,\dots,K-1$ 。K为码块长度, f_1 、 f_2 的值与码块大小K值有关,具体参数由LTE物理层协议36.212中5.1.3.2.3表5.1.3-3所定义。

[0007] 交织器的实现方式多种多样,最直接的方法就是按照公式依次计算,由于包括乘法操作和取模运算,这种计算方式效率非常低。现有的实现Turbo码内交织器的方式大体上可归类为根据递推公式计算,省去乘法操作,提高处理速度。如申请号为201210177005.2、发明名称为“一种用于LTE的并行Turbo码内交织器的实现”的中国发明专利公开说明书中提出了一种通过递推公式得到 $\Pi(i+8)$ 与 $\Pi(i)$ 之间的迭代关系,由当前的8种状态获得后续8种状态,实现一个时钟周期完成8个交织地址的输出,其8个交织地址分别读取8块内容相同的RAM并由主控模块处理得到8比特数据输出。又如申请号为201010293964.1、发明名称为“Turbo码并行交织器及并行交织方法”的中国发明专利公开说明书中提出了一种按列并行读取数据以及对数据进行行交叉的并行实现Turbo码内交织的方法,通过正向和逆向递推基本交织地址 $\Pi(i)$,并对其设定的L求模和求商分别获得列地址和首行的行地址,再根据临行之间行地址的递推关系,分别由前向和后向递推得到所有行的行地址,其所述控制器通过并行地读取一行数据并将读取的数据按照交叉单元产生的各行的行地址进行行交叉,实现数据的行内交织和行间交织,完成整个交织过程。

[0008] 在实际工作中,发明人发现现有Turbo码内交织器的实现方式均存在相应的问题:

如所述“一种用于LTE的并行Turbo码内交织器的实现”，其效率尽管可以提高8倍，但对于LTE物理层协议36.212规定的188种交织器均采用8位并行实现，显然有其局限性，且生成交织地址的过程依然是串行执行；而如所述“Turbo码并行交织器及并行交织方法”，其所提出的二次置换思想值得肯定，但其每次置换所需的行列地址均是由递推公式依次计算得来，其效率提升依然有局限性。

发明内容

[0009] (一) 要解决的技术问题

[0010] 为了有效减少LTE系统中Turbo码内交织器的实现时间，提高Turbo码编码器和译码器的效率，提升现有4G基站以及用户终端的数据吞吐率。

[0011] (二) 技术方案

[0012] 为解决上述技术问题，本发明提出一种并行实现Turbo码内交织的方法，将数据按列优先的方式存入一个输入缓冲区中，然后进行多次置换操作，所述每次置换操作是从该输入缓冲区中按行地址索引读入一行数据，再根据列地址索引进行交织操作，实现Turbo码内交织。

[0013] 根据本发明的一种具体实施方式，所述输入缓冲区的行数R和列数C根据以下规则进行计算：

$$[0014] \quad \begin{cases} R = 8, 40 \leq K \leq 512 \\ R = 16, 528 \leq K \leq 1024 \\ R = 32, 1056 \leq K \leq 2048 \\ R = 64, 2112 \leq K \leq 6144 \end{cases}。$$

[0015] $C = K/R$

[0016] 根据本发明的一种具体实施方式，将不同码块长度K对应的列置换时所需的列地址索引存放在一张表格中，在进行第一次置换操作时，所述列地址索引是根据码块长度K并通过查表得到。

[0017] 根据本发明的一种具体实施方式，所述列地址索引是由首行的列地址索引根据偏移量 Δ 和差值N计算得到。

[0018] 根据本发明的一种具体实施方式，得到第一行的列地址索引T0以及每行相对于第一行的偏移量 Δ 和差值N后，根据偏移量 Δ 对T0进行循环移位操作，并根据差值N完成加法操作，得到相应行的列地址索引。

[0019] 根据本发明的一种具体实施方式，列地址索引是根据不同码块长度K统计得到的。

[0020] 根据本发明的一种具体实施方式，统计得到的针对与不同码块长度K的偏移量和差值。

[0021] 根据本发明的一种具体实施方式，所述Turbo码内交织的实现方式的并行度由数据存入输入缓冲区的列数C和交织单元的位宽L决定，其并行度P的计算公式为 $P = C / \text{ceil}(C/L)$ ， $\text{ceil}()$ 表示对数据进行向上取整操作。

[0022] 本发明还提出一种并行实现Turbo码内交织的装置，包括行列地址生成单元、全交织单元、输入缓冲区，所述输入缓冲区存放需要做交织运算的输入信息流；所述行列地址生

成单元完成两部分工作：其一，将不同码块长度 K 对应的列置换时所需的列地址索引存放在一张表格中，在进行第一次置换操作时，所述列地址索引是根据码块长度 K 并通过查表得到；其二，计算每行输入缓冲区中待译码的数据的列地址索引，完成第二次置换所需的交织索引；所述全交织单元根据所述行列地址生成单元传递的列地址索引，对每一行数据进行全交织运算。

[0023] 根据本发明的一种具体实施方式，还包括输出缓冲区，其用于存储所述全交织单的输出数据。

[0024] (三) 有益效果

[0025] 本发明通过查表操作以及移位、加法等简单的算术计算，不需计算索引地址直接Turbo码内交织器的功能，具有以下有益之处：

[0026] 1) 查表操作实现简单，且表项存储所占空间较少，易于实现。

[0027] 2) 按行地址索引从输入缓冲区读入数据，不需额外计算即完成了第一次置换操作。

[0028] 3) 仅通过移位、加法以及交织操作就可以并行完成多个数据的交织，提高Turbo内交织器实现的并行度，降低执行时间。

附图说明

[0029] 图1是应用在LTE技术中的Turbo编码器结构图；

[0030] 图2是根据本发明的Turbo码内交织器的一种具体实施例结构图；

[0031] 图3是根据本发明的一种Turbo码内交织的实现流程图；

[0032] 图4是根据本发明的一种具体实施方式全交织操作示意图；

[0033] 图5是根据本发明的一种具体实施例中 $K=216$ 时输入数据缓冲区示意图；

[0034] 图6是根据本发明的一种具体实施例中 $K=216$ 时某时刻输出数据缓冲区示意图。

具体实施方式

[0035] 本发明提出了一种快速高效完成Turbo码内交织计算的并行实现方法，该方法仅包括查表以及移位、加法及交织等简单算术操作，利于嵌入式移动处理器实现，缩短执行时间，提高4G基站以及终端设备的数据吞吐率。

[0036] 为使本发明的目的、技术方案和优点更加清楚明白，以下结合具体实施例，并参照附图，对本发明作进一步的详细说明。

[0037] 图1是应用于LTE物理层协议中Turbo编码器的结构示意图。其编码码率为 $1/3$ ，对于输入的信息比特流 c_k ，分别得到系统信息比特流 x_k 、经过分量编码器1后得到的校验比特流 z_k 和经过分量编码器2后得到的校验比特流 z_k 。从图中可以看出，分量编码器2的输入比特流并不是原始输入信息，而是输入信息流经过交织器后得到的比特流 c'_k 。如前所述，交织器的加入对于改善移动通信的传输特性，降低信道衰落带来的影响是极为必要的。

[0038] 图2是根据本发明的Turbo码内交织器的一种具体实施例的结构示意图，主要包括4个模块：行列地址生成单元10、全交织单元20、输入缓冲区30和输出缓冲区40。其中输出缓冲区是可选的，在实际实现时，全交织单元20的输出数据可以直接输出到后续计算单元，这样可以节省内存资源的消耗。

[0039] 输入缓冲区20存放需要做交织运算的输入信息流。为了配合本发明的具体实现，输入信息比特流按照列优先的方式存入到输入缓冲区中。同时，根据LTE物理层协议中规定的码块长度K的统计特性，输入缓冲区的行数R和列数C是动态变化的，其取值根据以下规则进行计算：

$$[0040] \quad \begin{cases} R = 8, 40 \leq K \leq 512 \\ R = 16, 528 \leq K \leq 1024 \\ R = 32, 1056 \leq K \leq 2048 \\ R = 64, 2112 \leq K \leq 6144 \end{cases}$$

[0041] $C = K/R$

[0042] 也就是说输入缓冲区的大小根据具体输入数据块的大小而改变。例如K=40的情况下，行数R=8，列数C=5；而K=3136时，行数R=64，列数C=49。

[0043] 行列地址生成单元10主要完成两部分工作。其一，根据码块长度K的值，通过查表得到该码块进行第一次置换所需的行地址索引。显然地，该地址索引的长度与输入缓冲区的行数R相同。本发明人在具体工程实践中，对LTE物理层协议36.212中规定的188种不同的码块长度所对应的行地址索引进行了统计。为了减少专利说明书的篇幅，这里仅举例做一说明。如码块长度K=528，其输入缓冲区的行数和列数分别为16和33，对应的行地址索引为

[0044] $\{0, 3, 10, 5, 4, 7, 14, 9, 8, 11, 2, 13, 12, 15, 6, 1\}$ ；

[0045] 又如码块长度K=1056时，其输入缓冲区的行数和列数分别是32和33，对应的行地址索引为

[0046] $\{0, 19, 10, 5, 4, 7, 14, 25, 8, 27, 18, 13, 12, 15, 22, 1, 16, 3, 26, 21, 20, 23, 30, 9, 24, 11, 2, 29, 28, 31, 6, 17\}$ ；

[0047] 又如码块长度K=5760时，其输入缓冲区的行数和列数分别是64和90，对应的行地址索引为

[0048] $\{0, 25, 34, 27, 4, 29, 38, 31, 8, 33, 42, 35, 12, 37, 46, 39, 16, 41, 50, 43, 20, 45, 54, 47, 24, 49, 58, 51, 28, 53, 62, 55, 32, 57, 2, 59, 36, 61, 6, 63, 40, 1, 10, 3, 44, 5, 14, 7, 48, 9, 18, 11, 52, 13, 22, 15, 56, 17, 26, 19, 60, 21, 30, 23\}$ 。

[0049] 同时，本发明人在具体工程实践中也发现，对于LTE物理层协议36.212中所规定的188种码块长度，在不同码块长度下，其行地址索引也有一定规律，即行地址索引前后相差为定值。如上所述，码块长度K=528时，其周期为4，差值也为4；码块长度K=1056时，其周期为8，差值也为8；码块长度为5760时，其周期为4/8/16/32，其差值也相应地取4/8/16/32。本发明人在具体工程实践中发现，对于LTE物理层协议36.212所规定的188种不同码块长度，都有类似规律。这样，一方面在存储行地址索引时可以只存储一部分的索引值，节省内存开销；同时，在访问存储器时也可以根据此规律进行访存操作而非间接寻址，提高访存效率。

[0050] 行列地址生成单元10的另外一个工作就是计算每行数据的列地址索引，即完成第二次置换所需的交织索引。根据本发明人在具体工程实践中的经验，对于LTE物理层协议36.212中所规定的188种码块长度，可以得出如下规律：每行的列地址索引相对与第一行来说，仅存在偏移量 Δ 和差值N两点区别。具体说来就是，在已知第一行的列地址索引T0后，可以根据偏移量 Δ 对T0所循环移位操作，然后再根据差值N做加法运算，即可得到该行的列地

址索引。同样地,为了减少专利说明书的篇幅,这里仅举例做一说明。如码块长度 $K=400$,其输入缓冲区的行数和列数分别是8和50,其完成第二次置换首行的列地址索引

[0051] $T0 = \{0, 21, 32, 33, 24, 5, 26, 37, 38, 29, 10, 31, 42, 43, 34, 15, 36, 47, 48, 39, 20, 41, 2, 3, 44, 25, 46, 7, 8, 49, 30, 1, 12, 13, 4, 35, 6, 17, 18, 9, 40, 11, 22, 23, 14, 45, 16, 27, 28, 19\}$ 。

[0052] 每行的偏移量 Δ 和差值 N 分别为 $\{2, 4, 6, 8, 10, 12, 14\}$ 和 $\{41, 33, 25, 17, 9, 1, 43\}$ 。为了计算第二行的列地址索引,则通过对 $T0$ 做循环左移2之后,得到

[0053] $T1 (=T0 \ll 2) = \{32, 33, 24, 5, 26, 37, 38, 29, 10, 31, 42, 43, 34, 15, 36, 47, 48, 39, 20, 41, 2, 3, 44, 25, 46, 7, 8, 49, 30, 1, 12, 13, 4, 35, 6, 17, 18, 9, 40, 11, 22, 23, 14, 45, 16, 27, 28, 19, 0, 21\}$;

[0054] 然后在对 $T1$ 做加法操作(超过列数 C 的索引值需做模运算),得到该行最终的列地址索引

[0055] $T2 = \{23, 24, 15, 46, 17, 28, 29, 20, 1, 22, 33, 34, 25, 6, 27, 38, 39, 30, 11, 32, 43, 44, 35, 16, 37, 48, 49, 40, 21, 42, 3, 4, 45, 26, 47, 8, 9, 0, 31, 2, 13, 14, 5, 36, 7, 18, 19, 10, 41, 12\}$ 。

[0056] 同理,其他各行的列地址索引也可以由此得来。需要说明的是,这里为了叙述方便,设了三个变量 $T0$ 、 $T1$ 和 $T2$,在实际实现时,只需一个变量 $T0$ 即可,每次计算得到的列地址索引可直接传递到交织单元20。

[0057] 全交织单元20主要根据行列地址生成单元10传递的列地址索引,对每一行数据进行全交织运算,得到的结果送到输出缓冲区40。其原理如图4所示。列地址索引101即为行列地址生成单元10传递的某一行数据的列地址索引,全交织单元20根据列地址索引101,对输入数据寄存器201做全交织操作,得到交织后的数据存入结果寄存器202中。

[0058] 图3是根据本发明的一种实现Turbo码内交织的流程图,其计算过程分为一下几个步骤:步骤50根据行地址索引从输入缓冲区30中读取一行数据送入全交织单元20的输入数据寄存器201中。当然,如前所述,每次读入的数据多少与具体实现方式有关,可以读入一行,也可以根据规律,一次搬运多行数据。步骤60行列地址生成单元10根据偏移值 Δ 和差值 N ,在首行列地址索引的基础上,计算当前行的列地址索引101,送入全交织单元20中。步骤70全交织单元20根据得到的列地址索引101,对输入数据寄存器201做全交织操作,交织结果存入输出数据寄存器202中,或者直接存入输出缓冲区40中。步骤80则判断所有数据是否已全部完成交织运算,如果已全部完成交织运算则跳出循环,否则返回到步骤50继续上述操作。

[0059] 为了使本发明的思想进一步得到阐述,下面以码块长度 $K=216$ 为例进行说明。

[0060] 图5所示是输入的216个数据按列优先的方式存入输入缓冲区之后的示意图。如前所述,其行数和列数分别为8和27。按照前面所述的方法,根据行地址索引,从输入缓冲区30中读入每行数据完成第一次置换,再按照该行的列地址索引由全交织单元20完成第二次置换。图6所示即为读入三行数据并进行交织后输出缓冲区40的存储状态。需要说明的是,输出缓冲区40的存在与否由具体实现方式决定。

[0061] 本实施例所介绍的计算列地址索引的过程,是相对于首行的列地址索引做移位和加法操作等算术操作,其他等同替换如相对于前一行的列地址索引所做的操作也在本发明的保护范围之内。

[0062] 以上所述的具体实施例,对本发明的目的、技术方案和有益效果进行了进一步详

细说明,应理解的是,以上所述仅为本发明的具体实施例而已,并不用于限制本发明,凡在本发明的精神和原则之内,所做的任何修改、等同替换、改进等,均应包含在本发明的保护范围之内。

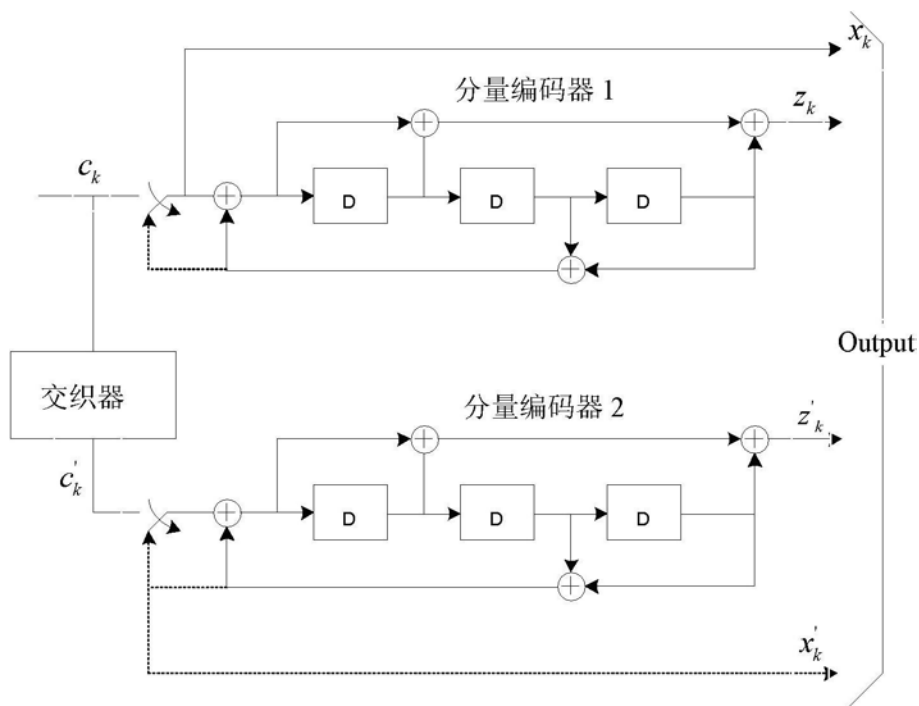


图1

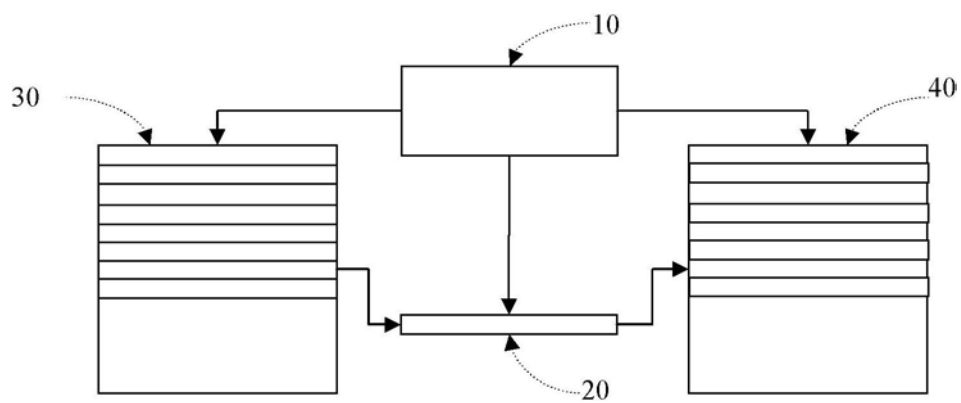


图2

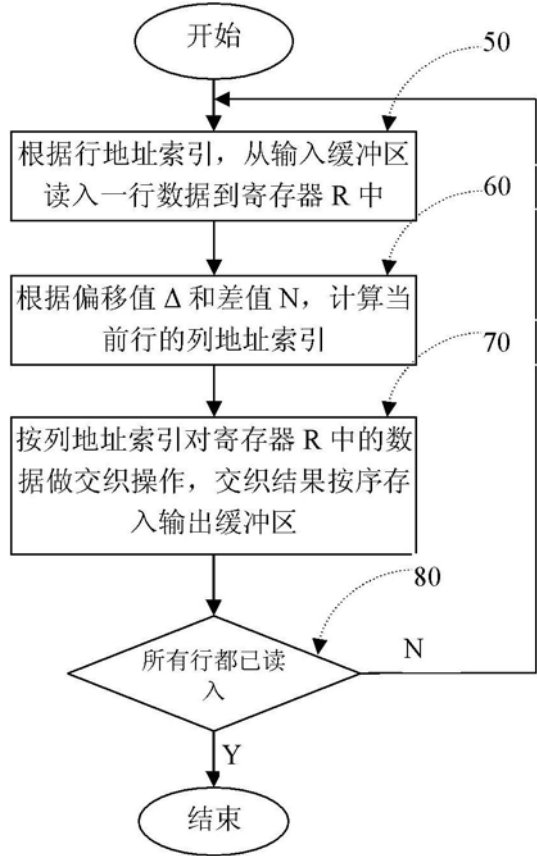


图3

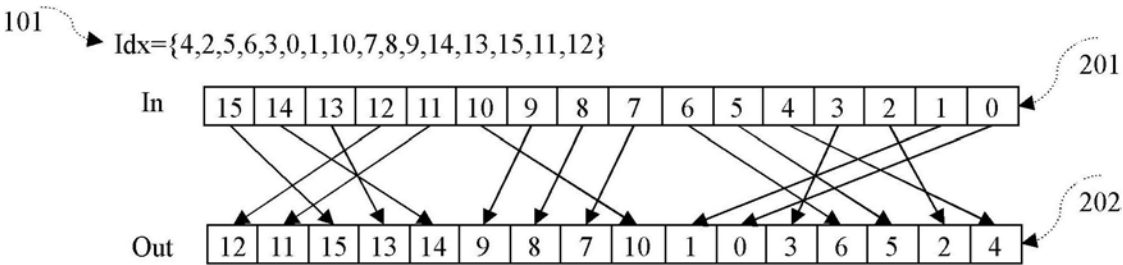
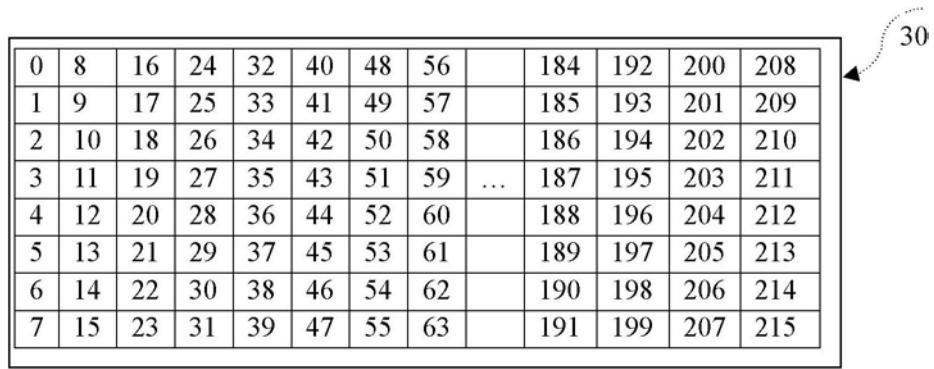
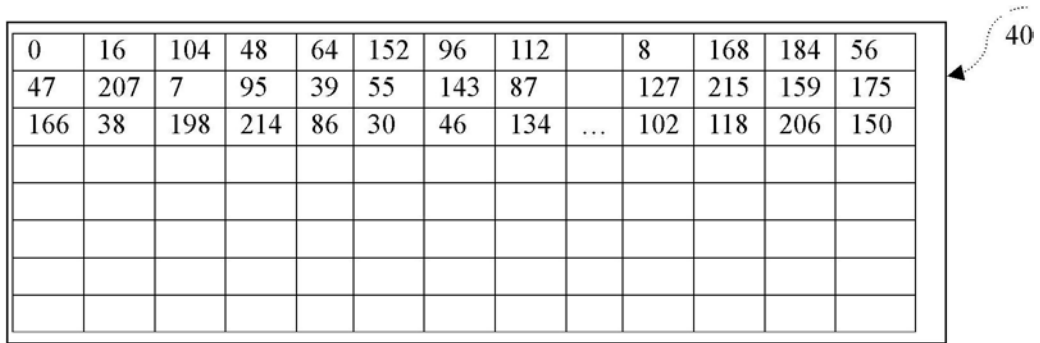


图4



0	8	16	24	32	40	48	56		184	192	200	208
1	9	17	25	33	41	49	57		185	193	201	209
2	10	18	26	34	42	50	58		186	194	202	210
3	11	19	27	35	43	51	59	...	187	195	203	211
4	12	20	28	36	44	52	60		188	196	204	212
5	13	21	29	37	45	53	61		189	197	205	213
6	14	22	30	38	46	54	62		190	198	206	214
7	15	23	31	39	47	55	63		191	199	207	215

图5



0	16	104	48	64	152	96	112		8	168	184	56
47	207	7	95	39	55	143	87		127	215	159	175
166	38	198	214	86	30	46	134	...	102	118	206	150

图6