一种HEVC标准中8×8IDCT变换的实现方法

申请号：201310241147.5
申请日：2013-06-18

申请(专利权)人　复旦大学
地址　200433 上海市杨浦区邯郸路220号
发明(设计)人　范益波　马天龙　刘聪　曾晓洋

主分类号　H04N7/26(2006.01)I
分类号　H04N7/26(2006.01)I　H04N7/30(2006.01)I　G06F17/14(2006.01)I
公开(公告)号　103327332A
公开(公告)日　2013-09-25
专利代理机构　上海正旦专利代理有限公司 31200
代理人　陆飞　盛志范

(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 103327332 B
(45)授权公告日 2016.04.13

(21)申请号 201310241147.5

(22)申请日 2013.06.18

(73)专利权人 复旦大学
地址 200433 上海市杨浦区邯郸路 220 号

(72)发明人 范益波 马天龙 刘聪 曾晓洋

(74)专利代理机构 上海正旦专利代理有限公司 31200
代理人 陆飞 盛志范

(51)Int.Cl.
H04N 19/625(2014.01)
H04N 19/42(2014.01)
H04N 19/176(2014.01)
G06F 17/14(2006.01)

审查员 荣芳

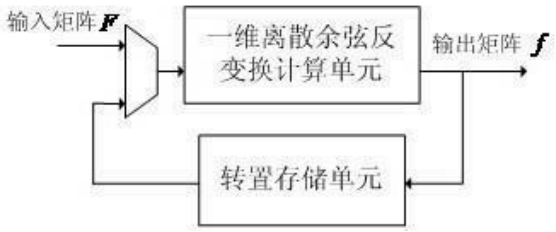
权利要求书2页 说明书5页 附图1页

(54)发明名称

一种 HEVC 标准中 8×8IDCT 变换的实现方法

(57)摘要

本发明属于数字视频信号编解码技术领域，具体为一种 HEVC 标准中 8×8IDCT 变换的实现方法。本发明通过将 8×8 的变换矩阵分解成稀疏矩阵相乘以及相加的形式，减小 8×8IDCT 变换中矩阵相乘的计算复杂度，从而大大降低系统的硬件开销。



CN 103327332 B

1.一种 HEVC 标准中 8×8IDCT 变换的实现方法,其采用两次相同的一维矩阵运算和两次转置操作来实现二维矩阵运算,其特征在于具体步骤如下:

(1) 输入 8×8 矩阵数据 F,以 F 的每一列为单位,进行式(1)的计算,进行 8 次这样的计算后,得到 8×8 输出矩阵 f_1:

$$f_1 = A^T \times F \qquad (1)$$

其中,A 为常数 8×8 矩阵,A 的具体数值如下:

$$A=\begin{bmatrix} 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 \\ 89 & 75 & 50 & 18 & -18 & -50 & -75 & -89 \\ 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 \\ 75 & -18 & -89 & -50 & 50 & 89 & 18 & 75 \\ 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 \\ 50 & -89 & 18 & 75 & -75 & -18 & 89 & -50 \\ 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 \\ 18 & -50 & 75 & -89 & 89 & -75 & 50 & -18 \end{bmatrix};$$

(2) 对矩阵 f_1 进行转置操作,得到矩阵 f_2;

(3) 以 f_2 的每一列为单位,进行式(3)的计算,进行 8 次这样的计算后,得到 8×8 输出矩阵 f_3:

$$f_3 = A^T \times f_2 \qquad (3)$$

其中,A 为常数 8×8 矩阵,A^T 为 A 的转置矩阵;

(4) 对矩阵 f_3 进行转置操作,得到最终的 8×8 输出矩阵 f;其中:

步骤(1)和步骤(3)中的常数矩阵分解为稀疏矩阵相乘和相加的形式,具体如下:

$$A^T = B_8 \times \begin{bmatrix} M & 0 \\ 0 & N \end{bmatrix} \times P_8$$

其中:

$$P_8=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad B_8=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

$$M=\begin{bmatrix}64 & 83 & 64 & 36\\ 64 & 36 & -64 & -83\\ 64 & -36 & -64 & 83\\ 64 & -83 & 64 & -36\end{bmatrix} N=\begin{bmatrix}18 & -50 & 75 & -89\\ 50 & -89 & 18 & 75\\ 75 & -18 & -89 & -50\\ 89 & 75 & 50 & 18\end{bmatrix}$$

M 和 N 进行进一步分解，M 分解为：

$$M=B_4\times\alpha\times P_4=\begin{bmatrix}1 & 0 & 0 & 1\\ 0 & 1 & 1 & 0\\ 0 & 1 & -1 & 0\\ 1 & 0 & 0 & -1\end{bmatrix}\times\begin{bmatrix}64 & 64 & 0 & 0\\ 64 & -64 & 0 & 0\\ 0 & 0 & 36 & -83\\ 0 & 0 & 83 & 36\end{bmatrix}\times\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

N 分解为：

$$N=\mu+\lambda$$

其中：

$$\mu=\begin{bmatrix}18 & -50 & 75 & -90\\ 50 & -90 & 18 & 75\\ 75 & -18 & -90 & -50\\ 90 & 75 & 50 & 18\end{bmatrix} \lambda=\begin{bmatrix}0 & 0 & 0 & 1\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ -1 & 0 & 0 & 0\end{bmatrix}$$

μ 进一步分解为：

$$\mu=\begin{bmatrix}0 & 0 & 0 & 1\\ 0 & 1 & -1 & 0\\ 0 & 1 & 1 & 0\\ 1 & 0 & 0 & 0\end{bmatrix}\times\begin{bmatrix}0 & 0 & 25/2 & 18\\ 0 & 0 & -9 & 25/2\\ 25/2 & -9 & 0 & 0\\ 18 & 25/2 & 0 & 0\end{bmatrix}\times\begin{bmatrix}1 & 0 & 0 & -5\\ 0 & -4 & 6 & 0\\ 0 & 6 & 4 & 0\\ 5 & 0 & 0 & 1\end{bmatrix}。$$

一种 HEVC 标准中 8×8IDCT 变换的实现方法

技术领域

[0001] 本发明属于数字视频信号编解码技术领域，针对 HEVC 视频标准，具体涉及一种 HEVC(High Efficiency Video Coding) 标准中 8×8IDCT(inverse discrete cosine transform) 变换的实现方法。

背景技术

[0002] HEVC(High efficiency video coding) 是由国际电信组织(ITU)和运动图像专家组(MPEG)联合制定而成的最新国际视频编码标准。相对于旧的 H.264 标准，HEVC 具有更高的压缩效率，更适合超高分辨率视频的编码，但其计算量与复杂度也剧增，以标准中的 IDCT 变换为例，H.264 标准采用 4×4 以及 8×8 的 IDCT 变换矩阵，而 HEVC 则采用 4×4、8×8、16×16 以及 32×32 的变换矩阵，这大大增加了硬件实现的复杂度；然而通过对矩阵进行适当分解，可以有效降低计算复杂度，提高计算速度以及减小硬件开销。

发明内容

[0003] 为了克服现有技术的不足，本发明的目的在于提出一种 HEVC 标准中 8×8IDCT 变换的实现方法，其可以有效降低计算复杂度，提高计算速度以及减小硬件开销。

[0004] 本发明提出的方法具体描述如下：

[0005] HEVC 中 8×8IDCT 的变换过程如下式所示：$f = A^T \times F \times A$。这是一个二维离散整数余弦变换，其中，$F$为输入 8×8 矩阵，$f$为输出 8×8 矩阵，$A$为常数 8×8 矩阵，$A^T$为 A的转置矩阵。A的具体数值如下：

[0006]

$$A=\begin{bmatrix} 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 \\ 89 & 75 & 50 & 18 & -18 & -50 & -75 & -89 \\ 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 \\ 75 & -18 & -89 & -50 & 50 & 89 & 18 & 75 \\ 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 \\ 50 & -89 & 18 & 75 & -75 & -18 & 89 & -50 \\ 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 \\ 18 & -50 & 75 & -89 & 89 & -75 & 50 & -18 \end{bmatrix}$$

[0007] 通过进行两次一维离散整数余弦变换来实现二维离散整数余弦变换，过程如下。

[0008] $f = A^T \times F \times A$ 又可以写成另一种形式：$f = \left[A^T \times \left(A^T \times F \right)^T \right]^T$，那么可以用如下过程来计算 $f = A^T \times F \times A$：

[0009] $f_1 = A^T \times F$ (1)

[0010] $$f_2 = f_1^T \quad (2)$$

[0011] $$f_3 = A^T \times f_2 \quad (3)$$

[0012] $$f = f_3^T \quad (4)$$

[0013] 上述计算中，式(1)和式(3)计算过程相同，式(2)和式(4)对矩阵进行了转置操作。图1描述了上述计算过程的整体框图。

[0014] 对于式(1)和式(3)所述的一维矩阵运算，通过将常数矩阵分解为稀疏矩阵相乘和相加的形式，得到一种实现方法，可以提高一维矩阵运算的计算速度和减低其计算复杂度。

[0015] 为方便进行说明，现将式(1)、式(3)写成$Y = A^T \times X$的形式，X为8×8矩阵，包含64个元素，记为$x_{i,j}$，$i,j = 0,1,\_7$，同样，Y也为8×8矩阵，包含64个元素，记为$y_{i,j}$，$i,j = 0,1,\_7$。

[0016] $Y = A^T \times X$的计算可以以X的列为单位进行，即

[0017] $$[y_{0,i}, y_{1,i}, y_{2,i}, y_{3,i}, y_{4,i}, y_{5,i}, y_{6,i}, y_{7,i}]^T = A^T \times [x_{0,i}, x_{1,i}, x_{2,i}, x_{3,i}, x_{4,i}, x_{5,i}, x_{6,i}, x_{7,i}]^T \quad i = 0,1,\_7 \quad (5)$$

[0018] 本发明中，A^T可以进行如下分解：

[0019] $$A^T = B_8 \times \begin{bmatrix} M & 0 \\ 0 & N \end{bmatrix} \times P_8 \quad (6)$$

[0020] 其中

[0021] $$P_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad B_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

[0022] $$M = \begin{bmatrix} 64 & 83 & 64 & 36 \\ 64 & 36 & -64 & -83 \\ 64 & -36 & -64 & 83 \\ 64 & -83 & 64 & -36 \end{bmatrix} \quad N = \begin{bmatrix} 18 & -50 & 75 & -89 \\ 50 & -89 & 18 & 75 \\ 75 & -18 & -89 & -50 \\ 89 & 75 & 50 & 18 \end{bmatrix}$$

[0023] M和N还可以进行进一步分解：

[0024] $$M=B_4\times\alpha\times P_4=\begin{bmatrix}1&0&0&1\\0&1&1&0\\0&1&-1&0\\1&0&0&-1\end{bmatrix}\times\begin{bmatrix}64&64&0&0\\64&-64&0&0\\0&0&36&-83\\0&0&83&36\end{bmatrix}\times\begin{bmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{bmatrix}\quad(7)$$

[0025] N 可以分解为

[0026] $$N=\mu+\lambda\quad(8)$$

[0027] 其中：

[0028] $$\mu=\begin{bmatrix}18&-50&75&-90\\50&-90&18&75\\75&-18&-90&-50\\90&75&50&18\end{bmatrix}\quad(9)\qquad\lambda=\begin{bmatrix}0&0&0&1\\0&1&0&0\\0&0&1&0\\-1&0&0&0\end{bmatrix}\quad(10)$$

[0029] 而 N 可以进一步分解，

[0030] $$N=\begin{bmatrix}0&0&0&1\\0&1&-1&0\\0&1&1&0\\1&0&0&0\end{bmatrix}\times\begin{bmatrix}0&0&25/2&18\\0&0&-9&25/2\\25/2&-9&0&0\\18&25/2&0&0\end{bmatrix}\times\begin{bmatrix}1&0&0&-5\\0&-4&6&0\\0&6&4&0\\5&0&0&1\end{bmatrix}\quad(11)$$

[0031] 本发明的有益效果在于：从而能够以更快的速度，更小的硬件开销来计算 HEVC 中的 8×8 二维整数离散余弦反变换。

附图说明

[0032] 图 1. 二维离散余弦反变换（IDCT）的整体框图。

[0033] 图 2. 式（5）的一种实现流程图。

具体实施方式

[0034] 图 1 所示为二维离散余弦反变换（IDCT）的整体框图。

[0035] 本发明所述的 HEVC 中 8×8IDCT 实现方法，具体实施方式如下：

[0036] （1）输入 8×8 矩阵数据 F，以 F 的每一列为单位，进行（1）式的计算，计算流程如图 2 所示，进行 8 次这样的计算后，得到 8×8 输出矩阵 f_1。

[0037] （2）对矩阵 f_1 进行转置操作，得到矩阵 f_2。

[0038] （3）以 f_2 的每一列为单位，进行（1）式的计算，计算流程如图 2 所示，进行 8 次这样的计算后，得到 8×8 输出矩阵 f_3。

[0039] （4）对矩阵 f_3 进行转置操作，得到最终的 8×8 输出矩阵 f。

[0040] 图 2 所示为根据上述分解原理所得到的关于式（5）的实现流程图，下面对其进行详细描述：

[0041] a) 输入为$x_{0,j}, x_{1,j}, x_{2,j}, x_{3,j}, x_{4,j}, x_{5,j}, x_{6,j}, x_{7,j}$。

[0042] b) 先进行与P_8的相乘，对应图中步骤 1 的操作，可以看到，与P_8相乘只是

[0043] 对输入进行了重新排序而已，并没有任何的计算操作。

[0044] c) 然后将前一步骤算出的结果分成上下两部分，分别与矩阵 *M*和 *N*进行相

[0045] 乘。步骤 2 到步骤 5 的上半部分对应与 *M*的相乘，步骤 2 到步骤 5 的下半部分对应与 *N*的相乘；其中，步骤 5 的上半部分没有操作，下半部分对应 $N=\mu+\lambda$ 的操作。

[0046] d) 步骤 6 对应的操作为将步骤 5 中得到的结果和B_8相乘。

[0047] e) 输出为$y_{0,j}, y_{1,j}, y_{2,j}, y_{3,j}, y_{4,j}, y_{5,j}, y_{6,j}, y_{7,i}$。

[0048] 本发明的这种实现方法与直接进行矩阵乘法相比，可以有效的减少乘法和加法的次数，乘法次数可以减少 66%，加法次数可以减少 46%，对比如下：

[0049]

	乘法	加法
直接计算	64	56
本发明	22	30

[0050] 整个实施过程可以方便的用软件或是硬件实现。

[0051] 用软件实现时，如下：

[0052] (1) 输入 8×8 矩阵数据F，以F的每一列为单位，进行(1)式的计算，计算流程如图 2 所

[0053] 示，用相应的软件语言依次描述每一步骤的计算过程，进行 8 次这样的计算后，得到 8×8 输出矩阵f_1。

[0054] (2) 对矩阵f_1进行转置操作，用软件实现时只需在读出时改变相应的行列号即可，得到f_1的转置矩阵f_2。

[0055] (3) 以f_2的每一列为单位，进行(1)式的计算，计算流程如图 2 所示，用相应的软件语言依次描述每一步骤的计算过程，进行 8 次这样的计算后，得到 8×8 输出矩阵f_3。

[0056] (4) 对矩阵f_3进行转置操作，用软件实现时只需在读出时改变相应的行列号即可，得到最终的 8×8 输出矩阵f。

[0057] 用硬件实现时，如下：

[0058] (1) 输入 8×8 矩阵数据F，以F的每一列为单位，进行(1)式的计算，计算流程如图 2 所示，用硬件实现时，可采用多级流水线的方式实现，流程图中的每一步骤可以对应于一级流水线，流程图中的小圆圈可以看做一个寄存器，用来保存每一级的计算结果。但由于步骤 1 只是对输入数据的位置进行变换，并没有计算操作，因此可以省去步骤 1 这级流水线，直接对输入进行相应位置的调整即可，进行 8 次这样的计算后，得到 8×8 输出矩阵f_1。

[0059] (2) 对矩阵f_1进行转置操作，用硬件实现时需要有相应的存储器来存储f_1的数

据，对存储器的写操作读操作进行相应处理即可得到f_1的转置矩阵f_2。

[0060] (3) 以f_2的每一列为单位，进行(1)式的计算，计算流程如图 2 所示，用硬件实现时，可采用多级流水线的方式实现，流程图中的每一步骤可以对应于一级流水线，流程图中的小圆圈可以看做一个寄存器，用来保存每一级的计算结果。但由于步骤 1 只是对输入数据的位置进行变换，并没有计算操作，因此可以省去步骤 1 这级流水线，直接对输入进行相应位置的调整即可，进行 8 次这样的计算后，得到 8×8 输出矩阵f_3。

[0061] (4) 对矩阵f_3进行转置操作，用硬件实现时需要有相应的存储器来存储f_3的数据，对存储器的写操作读操作进行相应处理即可得到最终的 8×8 输出矩阵f。

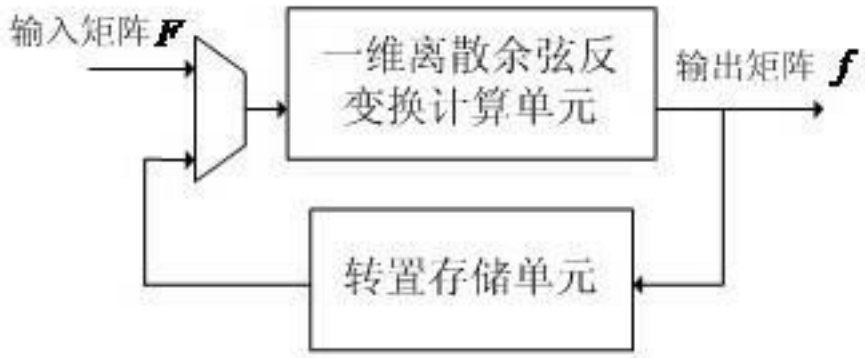

输入矩阵 F
一维离散余弦反变换计算单元
输出矩阵 f
转置存储单元

图 1

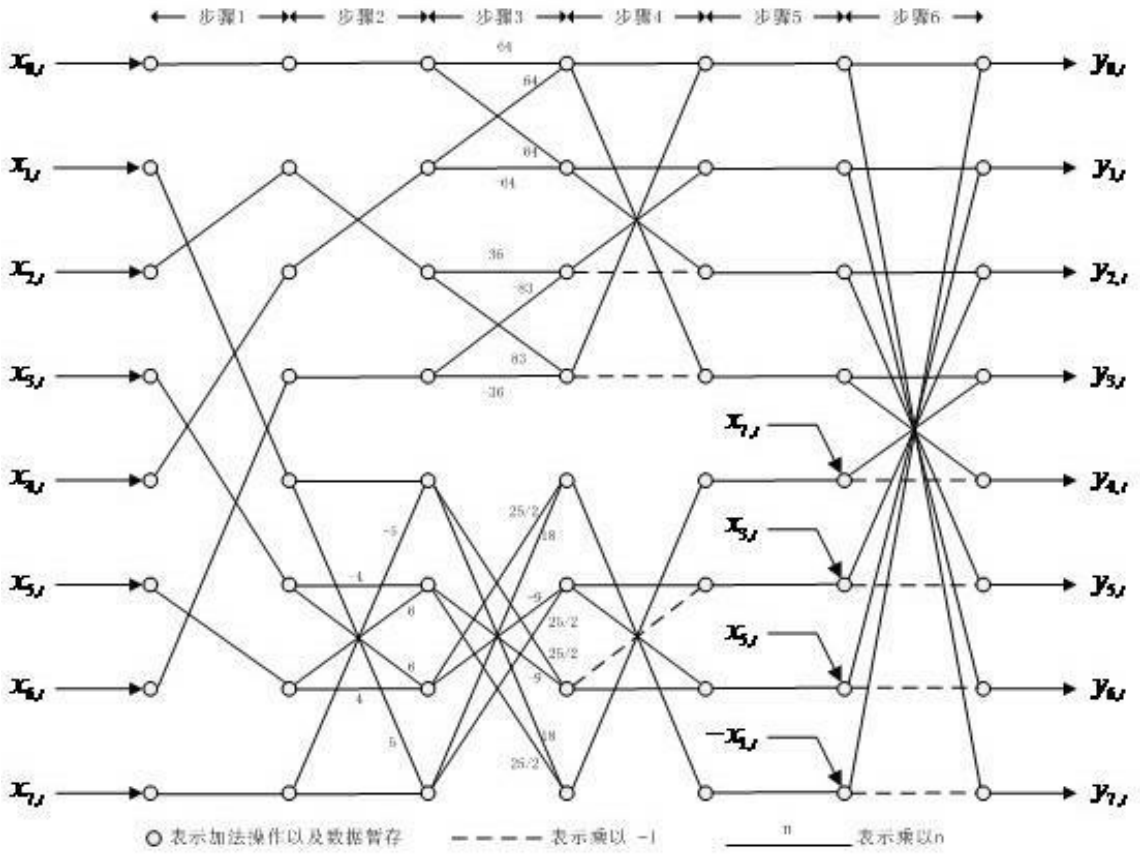

步骤1
步骤2
步骤3
步骤4
步骤5
步骤6
表示加法操作以及数据暂存
表示乘以 -1
表示乘以n

图 2