一种针对VC1自适应大小块的反变换结构

方 健[1], 张 丁[2], 王 匡[2]
(1. 浙江大学城市学院, 浙江杭州 310015; 2. 浙江大学信息学院, 浙江杭州 310027)

摘 要: 针对VC1的四种自适应反变换模式, 文章提出了一种基于8×8块的反变换结构. 利用VC1变换矩阵的对称性, 通过数据块重构, 四种反变换模式统一为相同的结构, 大大简化了硬件设计. 文章同时提出了硬件实现结构, 在满足应用要求的同时, 有效减小了硬件规模. 实验仿真表明, 在108MHz工作频率下, 能够有效支持标清和高清图像实时解码的反变换运算.

关键词: 视频解码; VC1; 反变换
中图分类号: TN919.81 文献标识码: A 文章编号: 0372-2112 (2009) 02-0419-05

Inverse Transform Architecture for VC1 Adaptive Block Size

FANG Jian[1], ZHANG Ding[2], WANG Kuang[2]
(1. *Zhejiang University City College, Hangzhou, Zhejiang* 310015, *China*;
2. *College of Information Science and Engineering, Zhejiang University, Hangzhou, Zhejiang* 310027, *China*)

Abstract: A unified inverse transform architecture based on 8×8block for Video Codec One standard is proposed in this paper. With the data reconstruction, four different inverse transforms mode could use the same architecture, which made the hardware design easy. At the same time, an inverse transform hardware architecture is proposed with less hardware resource. The experiment showed that when clocked at 108MHz, the proposed design could perform real-time inverse transform for standard definition and high definition video decoder.

Key words: video decode; VC1; inverse transform

1 引言

视频编码方案一(Video Codec One, 简称VC1)[1]的前身是微软公司的WMV9, 2004年经美国电影与电视工程师协会(SMPTE)审批并命名为SMPTE标准VC1, 成为视频压缩编码的产业标准之一. 它主要应用于数字电视广播, IPTV, 高清DVD等领域.

VC1是继H.264/AVC后的又一种新的视频编码标准, 相比于后者有两方面的优势. 一方面由于VC1是微软公司支持的技术标准, 在互联网具有广泛的应用前景. 另一方面, 由于采用新的压缩技术[2], 相同的压缩率下, VC1的运算复杂度相对较低, 更易于应用实现. 因此由于市场和技术这两方面的原因, 新一代的高清视频解码器需要考虑兼容VC1标准. VC1仍然属于传统的基于块的运动补偿混合编码技术, 因此可以采用和现有标准类似的解码结构. 但由于采用了新的算法和技术, 在模块功能实现时需要重新设计, 本文主要研究VC1解码器的反变换部分.

传统的视频压缩通常采用8×8的分块进行变换, 压缩频域冗余度. VC1采用自适应分块变换技术, 同时支持8×8、8×4、4×8和4×4四种分块的整数变换. 这不仅能提高图像压缩效率, 同时提高了重构图像的主观质量[3]. 但是, 自适应分块变换技术增加了解码器的复杂度, 本文在传统8×8DCT变换[4]的基础上, 首次提出了一种适用于VC1的整数反变换结构. 通过对自适应块的重构处理, 统一了整数反变换运算单元, 从而有利于解码器的规则化设计, 节省硬件资源. 在此基础上, 提出了适用于VC1反变换的硬件实现结构.

2 VC1反变换运算

视频编码通过变换和量化技术, 压缩视频图像数据的频域冗余. 一般的说, 大尺寸变换有利于压缩周期性纹理变化和运动趋势小的图像, 小尺寸有利于提高压缩率和减小块边缘的不连续性. 以往的视频编码标准广泛采用8×8的变换块. H.264标准增加了4×4块大小的变换, 对于直流系数还采用了2x2Hardmad变换. VC1标准以8×8块为变换基本块, 8×8块可进一步划分为8×4, 4×8和4×4的子块进行变换, 如图1. 通过四种变

收稿日期: 2007-08-08; 修回日期: 2008-05-04
基金项目: 国家自然科学基金(No. 90307002)

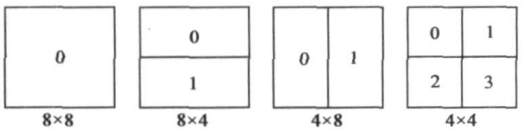

图1 VC1整数变换尺寸

换尺寸的灵活选取,增强了对图像细节的自适应性,提高了重构图像的主观质量.

VC1 标准[1] 附录 A 规定的反变换运算定义如式(1);其中 $\boldsymbol{T}_4$, $\boldsymbol{T}_8$ 为反变换常数矩阵,如式(2);$\boldsymbol{C}_8$ 为取整调节常数矩阵,如式(3);$\boldsymbol{C}_4$ 为长度 4 的零列向量;$\mathbf{1}_M$ 为长度 M 的单位行向量;运算"·"为矩阵乘法运算,"′"为矩阵转置运算,运算"≫"表示每个矩阵像素右移运算.

$$\boldsymbol{E}_{M\times N}=(\boldsymbol{D}_{M\times N}\cdot\boldsymbol{T}_M+4)\gg 3$$

$$\boldsymbol{R}_{M\times N}=(\boldsymbol{T}_N'\cdot\boldsymbol{E}_{M\times N}+\boldsymbol{C}_N\cdot\mathbf{1}_M+64)\gg 7,\ M,N\in\{4,8\} \tag{1}$$

$$\boldsymbol{T}_4=\begin{bmatrix}17&17&17&17\\22&10&-10&-22\\17&-17&-17&17\\10&-22&22&-10\end{bmatrix}$$

$$\boldsymbol{T}_8=\begin{bmatrix}12&12&12&12&12&12&12&12\\16&15&9&4&-4&-9&-15&-16\\16&6&-6&-16&-16&-6&6&16\\15&-4&-16&-9&9&16&4&-15\\12&-12&-12&12&12&-12&-12&12\\9&-16&4&15&-15&-4&16&-9\\6&-16&16&-6&-6&16&-16&6\\4&-9&15&-16&16&-15&9&-4\end{bmatrix} \tag{2}$$

$$\boldsymbol{C}_R=\begin{pmatrix}0&0&0&0&1&1&1&1\end{pmatrix}' \tag{3}$$

$\boldsymbol{D}_{M\times N}$是反量化系数矩阵,即反变换运算的输入系数矩阵,矩阵的元素为 12bits 带符号数,取值范围为-2048 到 2047 之间;$\boldsymbol{E}_{M\times N}$为反变换中间结果矩阵,其值在-4096 到 4095 之间;$\boldsymbol{R}_{M\times N}$为反变换结果矩阵,其值在-512 到 511 之间. 这里 M 表示变换分块的行像素个数,N 表示分块的列像素个数,故 $\boldsymbol{D}_{M\times N}$, $\boldsymbol{E}_{M\times N}$和 $\boldsymbol{R}_{M\times N}$ 际表示一个 $N\times M$ 矩阵.

3 基于 8×8 的统一反变换形式

VC1 支持四种 8×8, 8×4, 4×8 和 4×4 四种块大小的整数反变换,这不利于硬件的规则化设计. 如果简单地同时使用四种反变换模式,势必造成资源的浪费,本文提出了一种基于重构 8×8 块的统一反变换形式.

首先分析式(1)可以发现,VC1 二维反变换是依次进过行列一维反变换得到的,行列一维变换可以使用相同的结构,如式(4). 式中为表述方便起见,只表示了矩阵运算,不包括取整和移位操作,这些运算可以通过对矩阵运算结果的简单处理实现.

$$\boldsymbol{R}_{M\times N}=\boldsymbol{T}_N'\cdot(\boldsymbol{D}_{M\times N}\cdot\boldsymbol{T}_M)=\boldsymbol{T}_N'\cdot(\boldsymbol{T}_M'\cdot\boldsymbol{D}'_{M\times N})' \tag{4}$$

对于 8×8 变换,反变换如式(5). 为了获得一致的结构,对划分为更小子块的变换进行 8×8 重构,如式(6)~(8). 其中 $\boldsymbol{D}$ 表示反量化矩阵,$\boldsymbol{R}$ 表示反变换系数矩阵,$\mathbf{0}_4$ 表述 4×4 零矩阵. 下标 $M\times N$ 表示行像素个数 M,列像素个数 N 的分块大小;在 8×8 子块中,下标 $t8\times4$, $b8\times4$, $l4\times8$, $r4\times8$, $lt4\times4$, $rt4\times4$, $lb4\times4$ 和 $rb4\times4$ 分别表示的 8×4 上子块,8×4 下子块,4×8 左子块和 4×8 右子块,4×4 左上子块,4×4 右上子块,4×4 左下子块和 4×4 右下子块.

8×8 分块:

$$\boldsymbol{R}_{8\times8}=\boldsymbol{T}_8'\cdot(\boldsymbol{D}_{8\times8}\cdot\boldsymbol{T}_8)=\boldsymbol{T}_8'\cdot(\boldsymbol{T}_8'\cdot\boldsymbol{D}'_{8\times8})' \tag{5}$$

8×4 子块分割:

$$\begin{aligned}\begin{bmatrix}\boldsymbol{R}_{t8\times4}\\\boldsymbol{R}_{b8\times4}\end{bmatrix}&=\begin{bmatrix}\boldsymbol{T}_4'\cdot(\boldsymbol{D}_{t8\times4}\cdot\boldsymbol{T}_8)\\\boldsymbol{T}_4'\cdot(\boldsymbol{D}_{b8\times4}\cdot\boldsymbol{T}_8)\end{bmatrix}\\&=\begin{bmatrix}\boldsymbol{T}_4'&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4'\end{bmatrix}\begin{bmatrix}\boldsymbol{D}_{t8\times4}\\\boldsymbol{D}_{b8\times4}\end{bmatrix}[\boldsymbol{T}_8]\\&=\begin{bmatrix}\boldsymbol{T}_4'&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4'\end{bmatrix}\left(\boldsymbol{T}_8'\cdot\begin{bmatrix}\boldsymbol{D}_{t8\times4}\\\boldsymbol{D}_{b8\times4}\end{bmatrix}'\right)'\end{aligned} \tag{6}$$

4×8 子块分割:

$$\begin{aligned}[\boldsymbol{R}_{l4\times8}\ \boldsymbol{R}_{r4\times8}]&=[\boldsymbol{T}_8'\cdot(\boldsymbol{D}_{l4\times8}\cdot\boldsymbol{T}_4)\ \ \boldsymbol{T}_8'\cdot(\boldsymbol{D}_{r4\times8}\cdot\boldsymbol{T}_4)]\\&=\boldsymbol{T}_8'\cdot[\boldsymbol{D}_{l4\times8}\ \boldsymbol{D}_{l4\times8}]\begin{bmatrix}\boldsymbol{T}_4&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4\end{bmatrix}\\&=\boldsymbol{T}_8'\cdot\left(\begin{bmatrix}\boldsymbol{T}_4'&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4'\end{bmatrix}\cdot[\boldsymbol{D}_{l4\times8}\ \boldsymbol{D}_{l4\times8}]'\right)'\end{aligned} \tag{7}$$

4×4 子块分割:

$$\begin{aligned}\begin{bmatrix}\boldsymbol{R}_{lt4\times4}&\boldsymbol{R}_{rt4\times4}\\\boldsymbol{R}_{lb4\times4}&\boldsymbol{R}_{rb4\times4}\end{bmatrix}&=\begin{bmatrix}\boldsymbol{T}_4'\cdot(\boldsymbol{D}_{lt4\times4}\cdot\boldsymbol{T}_4)&\boldsymbol{T}_4'\cdot(\boldsymbol{D}_{rt4\times4}\cdot\boldsymbol{T}_4)\\\boldsymbol{T}_4'\cdot(\boldsymbol{D}_{lb4\times4}\cdot\boldsymbol{T}_4)&\boldsymbol{T}_4'\cdot(\boldsymbol{D}_{rb4\times4}\cdot\boldsymbol{T}_4)\end{bmatrix}\\&=\begin{bmatrix}\boldsymbol{T}_4'&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4'\end{bmatrix}\cdot\begin{bmatrix}\boldsymbol{D}_{lt4\times4}&\boldsymbol{D}_{rt4\times4}\\\boldsymbol{D}_{lb4\times4}&\boldsymbol{D}_{rb4\times4}\end{bmatrix}\cdot\begin{bmatrix}\boldsymbol{T}_4&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4\end{bmatrix}\\&=\begin{bmatrix}\boldsymbol{T}_4'&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4'\end{bmatrix}\cdot\left(\begin{bmatrix}\boldsymbol{T}_4'&\mathbf{0}_4\\\mathbf{0}_4&\boldsymbol{T}_4'\end{bmatrix}\cdot\begin{bmatrix}\boldsymbol{D}_{lt4\times4}&\boldsymbol{D}_{rt4\times4}\\\boldsymbol{D}_{lb4\times4}&\boldsymbol{D}_{rb4\times4}\end{bmatrix}'\right)'\end{aligned} \tag{8}$$

根据式(5)~(8),VC1 反变换可以统一成 8×8 结构,如式(9),含取整和移位运算. 其中 $\boldsymbol{D}_{8\times8}$、$\boldsymbol{R}_{8\times8}$、$\boldsymbol{A}_{8\times8}$、$\boldsymbol{B}_{8\times8}$和 $\boldsymbol{K}_{8\times8}$分别为基于 8×8 块重构的反量化系数矩阵,反变换系数矩阵,行反变换常数系数矩阵,列反变换常数系数矩阵和取整调节常数矩阵. 这样,VC1 二维整数反变换都可以通过 8×8 反量化系数依次执行 8×8 行变换和 8×8 列变换得到. 行列变换可以使用相同的结构,只需要根据变换分割类型选取合适的反变换常数.

$$\boldsymbol{R}_{8\times8}=(\boldsymbol{A}_{8\times8}\cdot((\boldsymbol{B}_{8\times8}\cdot\boldsymbol{D}'_{8\times8}+4)\gg3)'+\boldsymbol{K}_{8\times8}+64)\gg7$$

$$A_{8\times8}=\begin{cases} T'_8, & 8\times8 \text{ or } 4\times8 \text{ Inverse Transform} \\ \begin{bmatrix} T'_4 & 0_4 \\ 0_4 & T'_4 \end{bmatrix}, & 8\times4 \text{ or } 4\times4 \text{ Inverse Transform} \end{cases}$$

$$B_{8\times8}=\begin{cases} T'_8, & 8\times8 \text{ or } 4\times4 \text{ Inverse Transform} \\ \begin{bmatrix} T'_4 & 0_4 \\ 0_4 & T'_4 \end{bmatrix}, & 4\times8 \text{ or } 4\times4 \text{ Inverse Transform} \end{cases}$$

$$K_{8\times8}=\begin{cases} C_{8\times8}, & 8\times8 \text{ or } 4\times8 \text{ Inverse Transform} \\ [0]_{8\times8}, & 8\times4 \text{ or } 4\times4 \text{ Inverse Transform} \end{cases}$$

其次可以发现, T'_8 和 T'_4 具有这样的特性: 偶数列偶对称, 奇数列奇对称. 因此可以参考文献[4]的方法, 通过奇偶分解进一步简化运算. 式(9)的行列一维 8×8 反变换统一表示为 $Y=H\cdot X$, 利用变换矩阵的奇偶特性分解为两个 4×4 矩阵乘法, 如式(10)和(11). 其中 $X0, X1, \cdots, X7$ 为反变换输入矩阵 X 的行向量, $Y0, Y1, \cdots, Y7$ 为反变换输出矩阵 Y 的行向量.

$$\begin{bmatrix} Y0 \\ Y1 \\ Y2 \\ Y3 \end{bmatrix}=\begin{bmatrix} a & c & a & f \\ a & f & -a & -c \\ a & -f & -a & c \\ a & -c & a & -f \end{bmatrix}+\begin{bmatrix} X0 \\ X2 \\ X4 \\ X6 \end{bmatrix}\begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix}\begin{bmatrix} X1 \\ X3 \\ X5 \\ X7 \end{bmatrix}$$

$$\begin{bmatrix} Y7 \\ Y6 \\ Y5 \\ Y4 \end{bmatrix}=\begin{bmatrix} a & c & a & f \\ a & f & -a & -c \\ a & -f & -a & c \\ a & -c & a & -f \end{bmatrix}+\begin{bmatrix} X0 \\ X2 \\ X4 \\ X6 \end{bmatrix}\begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix}\begin{bmatrix} X1 \\ X3 \\ X5 \\ X7 \end{bmatrix}$$

$$[a, b, c, d, e, f, g]=[12, 16, 16, 15, 9, 6, 4], \quad \text{when } H=T'_8 \tag{10}$$

$$\begin{bmatrix} Y0 \\ Y1 \\ Y4 \\ Y5 \end{bmatrix}=\begin{bmatrix} h & h & 0 & 0 \\ h & -h & 0 & 0 \\ 0 & 0 & h & h \\ 0 & 0 & h & -h \end{bmatrix}+\begin{bmatrix} X0 \\ X2 \\ X4 \\ X6 \end{bmatrix}\begin{bmatrix} i & j & 0 & 0 \\ j & -i & 0 & 0 \\ 0 & 0 & i & j \\ 0 & 0 & j & -i \end{bmatrix}\begin{bmatrix} X1 \\ X3 \\ X5 \\ X7 \end{bmatrix}$$

$$\begin{bmatrix} Y3 \\ Y2 \\ Y7 \\ Y6 \end{bmatrix}=\begin{bmatrix} h & h & 0 & 0 \\ h & -h & 0 & 0 \\ 0 & 0 & h & h \\ 0 & 0 & h & -h \end{bmatrix}+\begin{bmatrix} X0 \\ X2 \\ X4 \\ X6 \end{bmatrix}\begin{bmatrix} i & j & 0 & 0 \\ j & -i & 0 & 0 \\ 0 & 0 & i & j \\ 0 & 0 & j & -i \end{bmatrix}\begin{bmatrix} X1 \\ X3 \\ X5 \\ X7 \end{bmatrix}$$

$$[h, i, j]=[17, 22, 10], \text{ when } H=\begin{bmatrix} T'_4 & 0_4 \\ 0_4 & T'_4 \end{bmatrix} \tag{11}$$

VC1 反变换无论采用哪种模式, 8×8 块都可以表示成式(9)的形式, 从而获得统一的结构. 然后根据式(10)和(11)进行奇偶分解, 8×8 矩阵运算转化为奇偶部分的 4×4 变换, 进一步简化结构. 不同的反变换模式, 只需要在选择变换常数和存储结果时做相应变化就可以, 这样 VC1 四种反变换模式, 就可以采用相同的运算结构. 本文提出的反变换结构就是基于式(9)、(10)和(11)的硬件实现方案.

4　硬件实现结构

通过 8×8 重构和奇偶分解, 统一的 VC1 反变换可以采用传统的 8×8IDCT 硬件结构[5], 如图(2). 该结构包括行列选择 RCM(Row Column Mux), 偶序数据处理单元 EODP(Even Order Data Process), 奇序数据处理单元 OODP(Odd Order Data Process), 数据后处理单元 DPBU(Data Process Behind Unit) 和转置存储器 TM(Transpose Memory). Wc 表示反量化系数矩阵的列数据, Wr 是列变换后的中间系数矩阵的行数据, De 为偶序数据($W0$, $W2$, $W4$, $W6$), Do 为奇序数据($W1$, $W3$, $W5$, $W7$), Ye 为偶序运算结果, Yo 是奇序运算结果, Z 为一维反变换运算结果, X 为二维反变换运算结果. EODP, OODP 和 DPBU 构成一维反变换的运算部分. 二维反变换先进行列变换, 其中反量化数据 Wc 按照列结构存储, 每个地址两个列数据. 按照(Wc_{0n}, Wc_{1n}), (Wc_{2n}, Wc_{3n}), (Wc_{4n}, Wc_{5n}), (Wc_{6n}, Wc_{7n}), n 代表列, 每拍两个数据, 保证 EODP 和 OODP 同时工作. 列变换结构通过 TM 转置后, 再进行一次行反变换, 二维反变换结果按照奇偶输出到后端缓冲区.

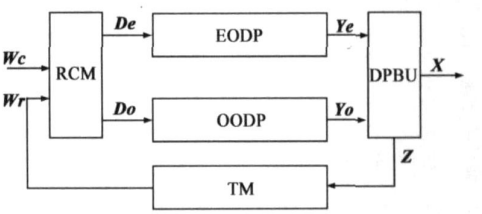

图2　传统IDCT

RCM 是选择器组, 选择行变换或列变换的数据. EODP 和 OODP 根据式(10)和(11)分别实现偶序数据和奇序数据的 4×1 矩阵运算, 如图 3, 选择器组根据反变换模式选择对应的运算结果, 累加器由加法器和寄存器构成, Ye0~ Ye3, Yo0~ Yo3 为四拍累加结果.

DPBU 结构如图 4. 寄存器组 1 由 6 个像素点寄存器构成, 寄存累加结果 Ye1、Ye2、Ye3、Yo1、Yo2 和 Yo3. 偶序和奇序结果通过两个加法器合成 Z0~ Z7, 再根据式(9)的取整和移位操作得到一维反变换结果, 输出到 TM 或缓存到寄存器组 2. 缓存器组 2 由 8 个像素点的寄存器构成, 它和转置存储器 TM 一起实现行列转置.

转置存储器采用一片 32×32bits 的 SRAM, 通过巧妙的数据存储结构, 实行每个时钟周期 2 点列反变换结果的写和 2 点行变换输入数据的读. 以列变换为例, Amn 代表列反变换结果, m(0~ 7)代表在 8×8 块中的行位置, n(0~ 7)代表在 8×8 块中的列位置. 第一列反变换四拍结果(A00, A70)、(A10, A60)、(A20, A50)和(A30, A40)寄存到寄存器组 2, 第二列结果和寄存器数据(A00, A01)、(A10, A11)、(A20, A21)和(A30, A31)4 拍写到转置存储器 TM, 同时 A71, A61, A51, A41 寄存到 A00, A10, A20, A30 位置. 第三列反变换时, (A70, A71)、(A60, A61)、(A50, A51)和(A40, A41)4 拍写到 TM, 同时

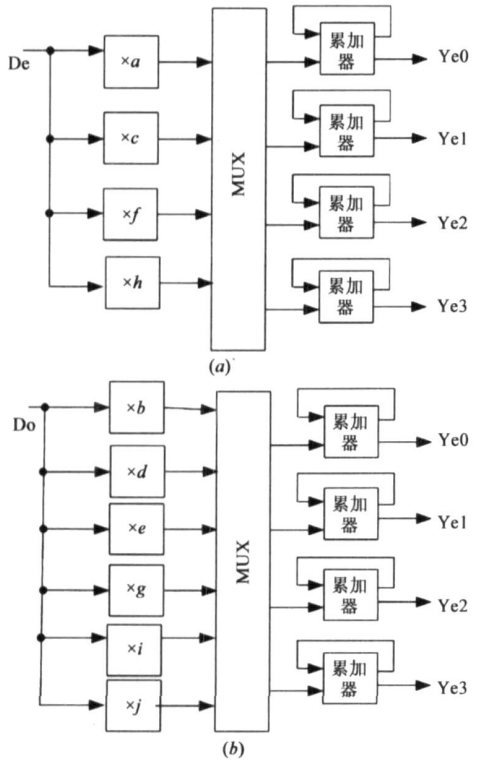

图3 矩阵计算结构

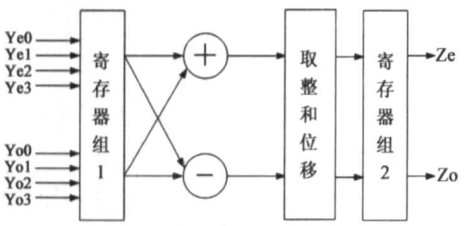

图4 数据后处理结构

第三列反变换结果寄存. 列反变换结束,TM 内数据如表 1. 行变换时按地址读取一行的两个数据. 行变换结果也通过寄存器组 2 调整次序后输出.

表 1 转置存储器中的数据

A00	A01	A02	A03	A04	A05	A06	A07
A10	A11	A12	A13	A14	A15	A16	A17
A20	A21	A22	A23	A24	A25	A26	A27
A30	A31	A32	A33	A34	A35	A36	A37
A40	A41	A42	A43	A44	A45	A46	A47
A50	A51	A52	A53	A54	A55	A56	A57
A60	A61	A62	A63	A64	A65	A66	A67
A70	A71	A72	A73	A74	A75	A76	A77

可以发现, 本文的行列转置结构增加了 4 拍流水深度用于调整行列数据, 如果 TM 采用双口 SRAM, , 8×8 内行列反变换和不同 8×8 块间的反变换可以完全流水,因此平均处理速度为每个 8×8 块 64 个时钟周期. 如果 TM 采用单口 SRAM, 不同 8×8 块间反变换可以流水, 但 8×8 内行列变换不能完全流水, 列变换结束需要 4 拍寄存和 4 拍写 SRAM, 平均处理速度为每个 8×8 块 72 个时钟周期.

5 性能分析

本文提出的设计结构使用 Verilog HDL 描述, VCS 仿真, 采用 Faraday 0.18μm 工艺库综合, 工作频率在 108MHz, 主要性能如表 2. 其中处理能力指每秒钟反变换模块能够运算的帧数, 以 fps 为单位. 分别使用单口和双口 SRAM 作为转置存储器 TM 进行仿真. 625 SD 指 720@576 标清; 720p HD 指 1024@768 高清; 1080 HD 指 1920@1088 高清.

表 2 反变换主要性能

	SRAM 容量 /bits	总面积 /μm^2	时延 /ns	平均处理速度 /cycles	625 SD 处理能力 /fps	720pHD 处理能力 /fps	1080HD 处理能力 /fps
单口 TM 结构	1P 32×32	115875	9.07	72	154.3	81.3	30.6
双口 TM 结构	2P 32×32	122755	9.07	64	173.6	91.5	34.43

6 结论

本文利用 VC1 变换矩阵的对称性, 通过数据块重构, 将 VC1 自适应反变换统一成以 8×8 块为单位的整数反变换结构, 简化了硬件设计. 通过奇偶分解, 进一步减小了硬件规模. 在此基础上, 文章提出了一种适用于 VC1 反变换的硬件实现结构, 实验表明在保证视频解码的要求下, 本文结构有效地控制了硬件规模. 本文提出的重构方法和硬件结构同样适用于 H.264 视频标准的反变换设计.

参考文献:

[1] Proposed SMPTE Standard for Television VC-1 Compressed Video Bitstream Format and Decoding Process[S]. SMPTE Technology Committee C24 on Video Compression Technology, 2005.

[2] Srinivansan S, Regunathan S. L. An Overview of VC-1[A]. Visual Communication and Image Processing2005[C]. Beijing: VCIP, 2005: 5960.

[3] Srinivasan S, Pohsiang J H Holcomb Tom, et al. Windows Media Video9: overview and applications[J]. Signal Processing: Image Communication. 2004, 19(9): 851- 874.

[4] Lee Y P, Chen T H, Chen L G, et al. A cost effective architecture for 8×8 two dimensional DCT/IDCT using direct method [J]. IEEE Trans Circuit Syst Video Technol, 1997, 7(3): 459 - 467.

[5] 傅宇卓, 王嘉芳, 胡铭曾. 一种新型 2-DCT/IDCT 结构的

设计与实行[J]. 电子学报, 2002, 30(12): 2126- 2129.
Fu Yur zhou, Wang Jiar fang, Hu Ming- zeng. The Design and Implementation of a Novel 2- DCT/IDCT Architecture[J]. Acta Electronica Sinica, 2002, 30(12): 2126- 2129. (in Chinese)

作者简介:

方　健　男, 1980 年生于浙江德清. 浙江大学信息学院信电系博士, 从事视频压缩, 数字电视和集成电路方面的研究.
E- mail: fangjian1980@163. com

张　丁　男, 1978 年生于浙江绍兴. 浙江大学信电系博士, 从事视频压缩, 数字电视和集成电路方面的研究.

王　匡　男, 1968 年生于浙江江山, 浙江大学信电系教授, 博导生导师, 主要从事数字通信、集成电路和数字电视的研究.

(上接第 418 页)

[16] 刘帘曦, 杨银堂, 朱樟明, 付永朝. 基于 Verilog- A 行为描述模型的 VCO 设计[J]. 电路与系统学报, 2005, 10(6): 25- 28.
Liu Liarn xi, Yang Yirn tang, Zhu Zhang- ming, Fu Yong- chao. Design of VCO based behavioral model uing Verilog- A[J]. Jounal of Circuits and Systems, 2005, 10(6): 25- 28. (in Chinese)

[17] Chilr de Hung, Wern shen Wuen, et al. A Unified Behavior Model of Low Noise Amplifier for Systenr Level Simulation [A]. Proceedings of the 9th European Conference on Wireless Technology[C]. Manchester UK: IEEE Press, 2006. 139- 142.

[18] Eskinat, E, S H Johnson, et al. Use of Hammerstein Models in Identification of Nonlinear Systems[J]. AiChE Journal, 1991, 37(2): 255- 268.

[19] Huar dong Wang, Song- bai He, et al. An Envelope Hammerstein Model for Power Amplifiers[J]. Journal of Electronic Science and Technology of China, 2007, 5(4): 362- 365.

[20] Kamal K Sabet, T Riad. A Generic VHDL- AMS Behavioral Model Physically Accounting For Typical Analog Norn Linear Output Behavior[A]. IEEE International Workshop on Behavioral Modeling and Simulation(BMAS' 2007)[C]. San Jose (USA) CA: IEEE Press, 2007. 105- 109.