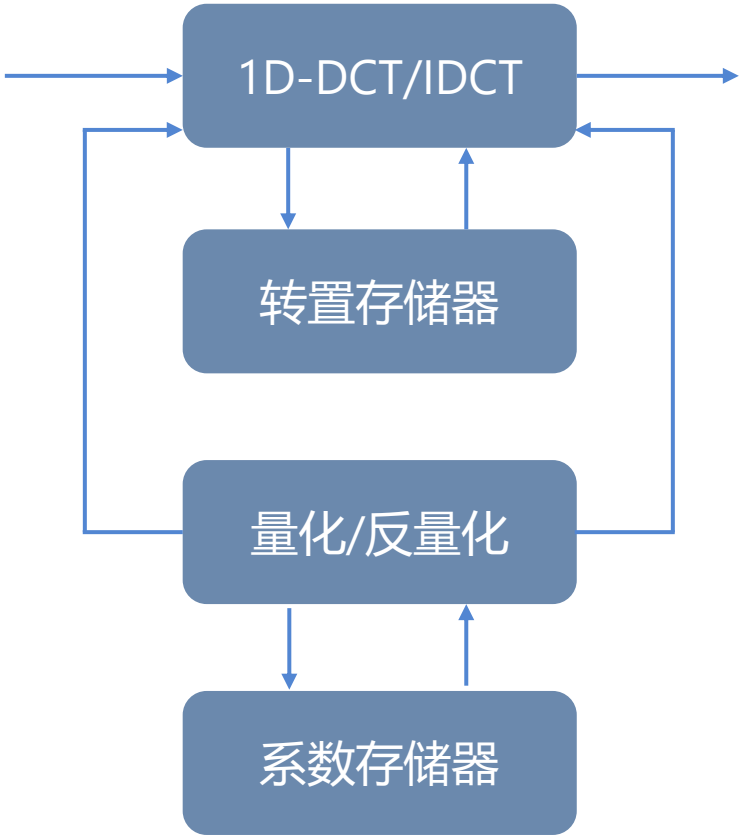
ASIC | 专注开源硬件 IP Core
www.openasic.org

DCT与量化的基本结构

作者 魏家聪

日期 2017.01.13

DCT与量化的基本结构



DCT所使用的算法

DCT $K_N = A_N * F_N * A_N^T$

IDCT $\mathrm{F}_N = A_N^T * K_N * A_N$

$$\begin{bmatrix} 64 & 64 & 64 & 64 \\ 83 & 36 & -36 & -83 \\ 64 & -64 & -64 & 64 \\ 36 & -83 & 83 & -36 \end{bmatrix}$$

4x4 的A_N矩阵

DCT所使用的算法

$$\begin{bmatrix} 64 & 64 & 64 & 64 \\ 83 & 36 & -36 & -83 \\ 64 & -64 & -64 & 64 \\ 36 & -83 & 83 & -36 \end{bmatrix}$$

A_N矩阵的左右两侧有很强的对称性

DCT所使用的算法

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

$$\begin{bmatrix} 64 & 64 & 64 & 64 \\ 83 & 36 & -36 & -83 \\ 64 & -64 & -64 & 64 \\ 36 & -83 & 83 & -36 \end{bmatrix}$$

A_N矩阵的偶数行可以生成$A_{N/2}$矩阵

1D-DCT所使用的算法

$$Y_N = A_N \times X_N^T$$

$$= P_N \times \begin{bmatrix} A_{N/2} & 0 \\ 0 & R_{N/2} \end{bmatrix} \times B_N \times X_N^T$$

$$= P_N \times \begin{bmatrix} A_{N/2} \times Q_A \\ R_{N/2} \times Q_S \end{bmatrix}$$

$$= P_N \times H_N$$

$B_N \times X_N^T$

N点蝶形运算

$A_{N/2} \times Q_A$

$R_{N/2} \times Q_S$

N/2点DCT
N/2点矩阵向量积

$P_N \times H_N$

N点排序

1D-DCT所使用的算法：系数缩放

ASIC
专注开源硬件 IP Core
www.openasic.org

DCT

$$Y = (X + offset) >> \left[(M-1)+(B-8)\right]$$
$$offset = 1 << \left[(M-2)+(B-8)\right]$$

第一次变换

$$Y = (X + offset) >> (M+6)$$
$$offset = 1 << (M+5)$$

第二次变换

IDCT

$$Y = (X + offset) >> 7$$
$$offset = 1 << 6$$

第一次变换

$$Y = (X + offset) >> (20\text{-}B)$$
$$offset = 1 << (19\text{-}B)$$

第二次变换

B: Bit Depth

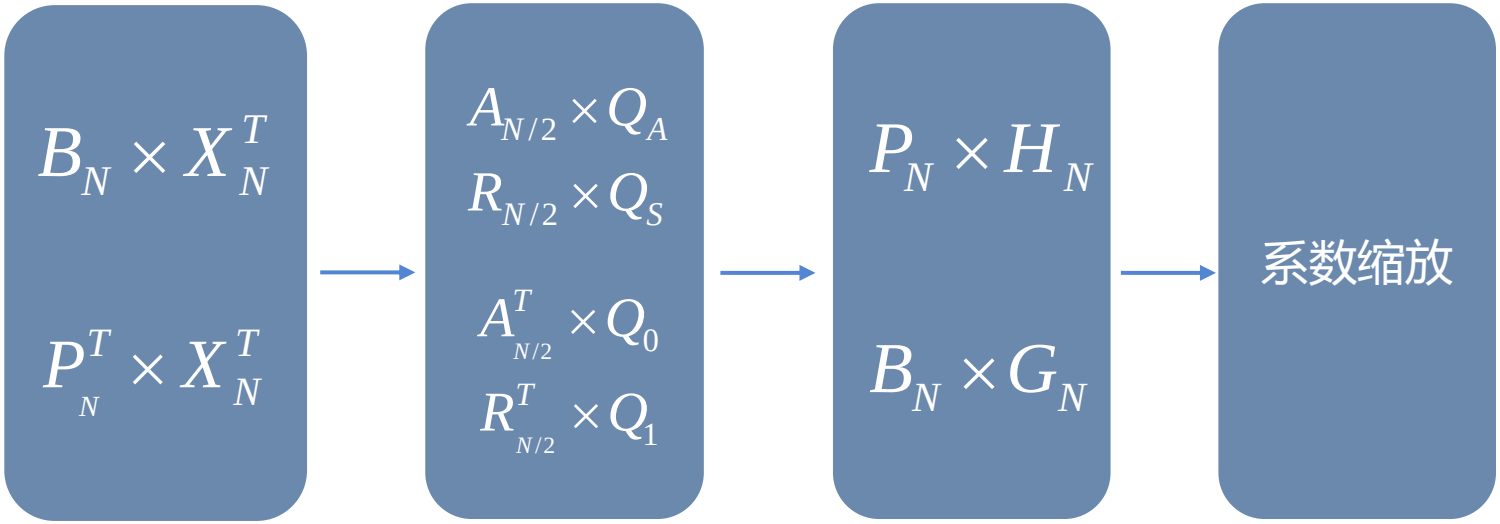
N: 变换尺寸

M: $\log_2(N)$

保证 DCT 的最终输出为 16bit

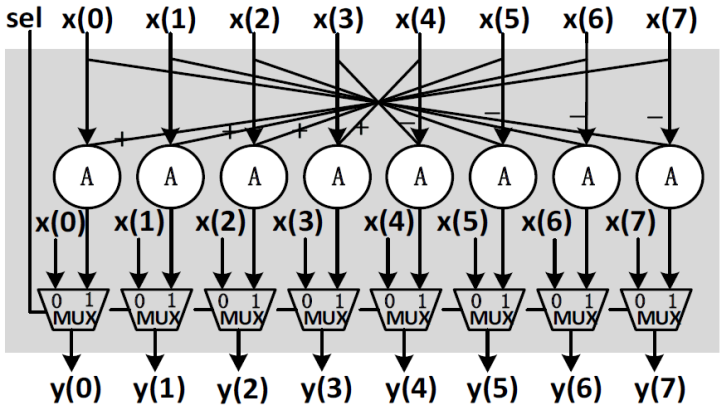
1D-DCT的结构

（以DCT为例）



1D-DCT的结构

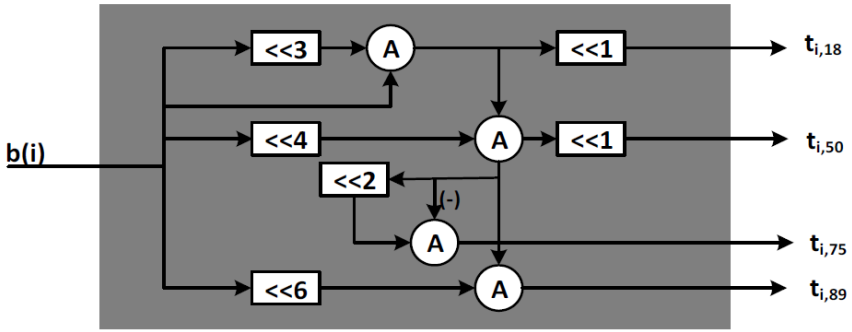
$$B_8 * X_8^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} x_0 + x_7 \\ x_1 + x_6 \\ x_2 + x_5 \\ x_3 + x_4 \\ x_3 - x_4 \\ x_2 - x_5 \\ x_1 - x_6 \\ x_0 - x_7 \end{bmatrix}$$



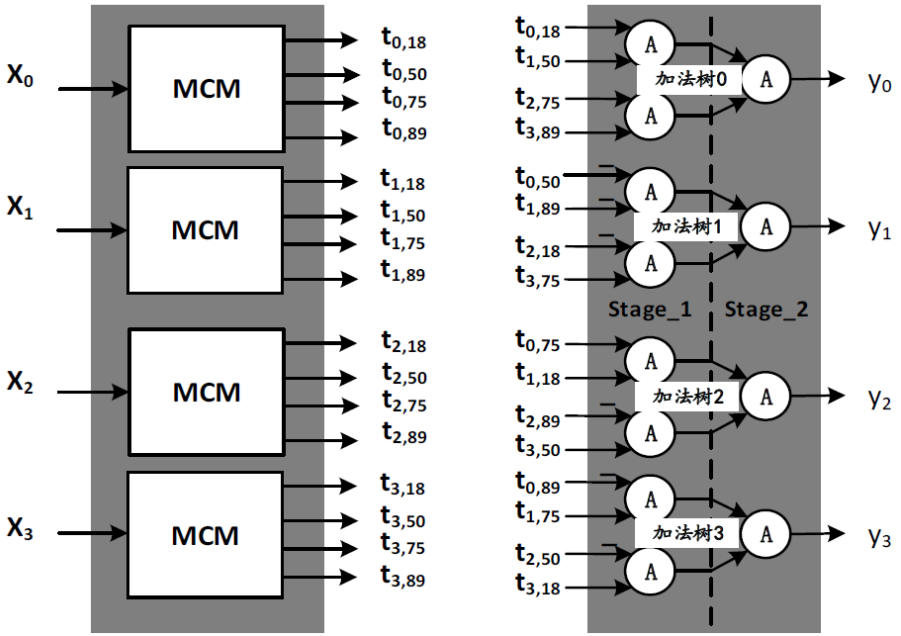
$$B_N \times X_N^T$$

1D-DCT的结构

$$R_4 * X_4^T = \begin{bmatrix} 18 & 50 & 75 & 89 \\ -50 & -89 & -18 & 75 \\ 75 & 18 & -89 & 50 \\ -89 & 75 & -50 & 18 \end{bmatrix} * \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$



R_4的MCM模块

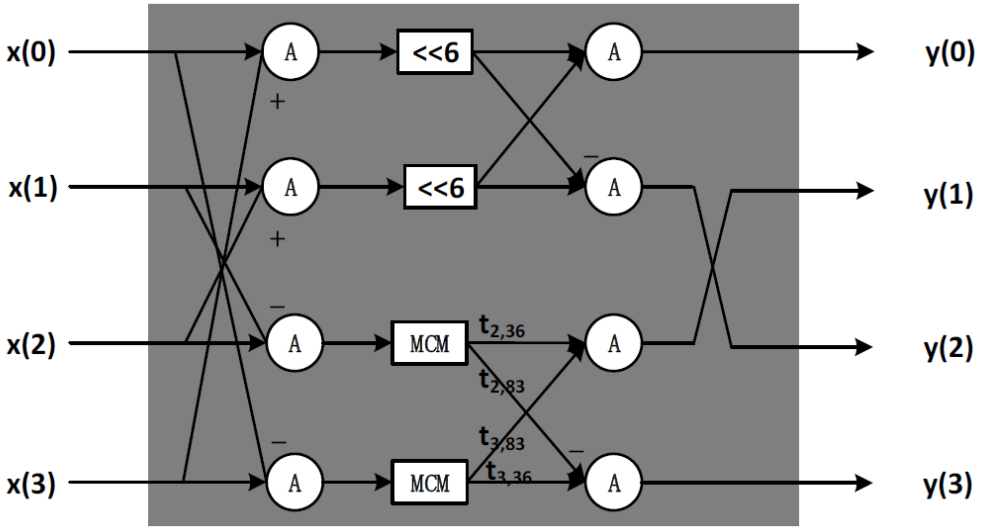


R_4乘法模块

$$R_{N/2} \times Q_S$$

1D-DCT的结构

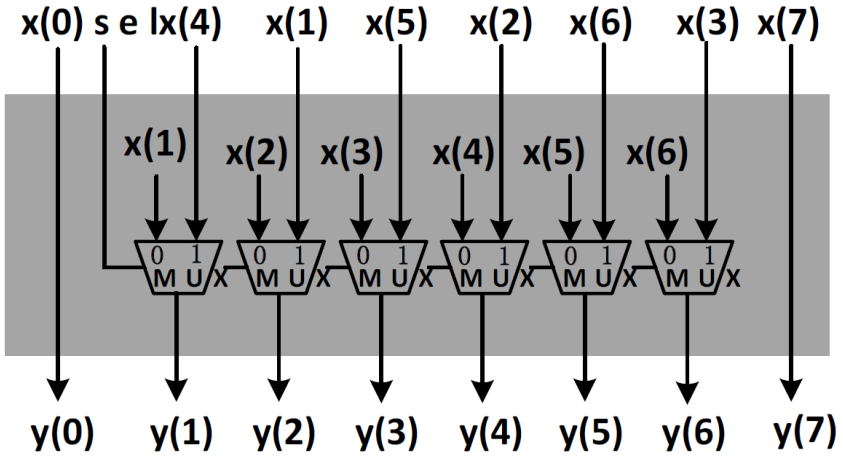
$$A_4 = P_4 \times \begin{bmatrix} A_2 & 0 \\ 0 & R_2 \end{bmatrix} \times B_4, \quad A_2 = \begin{bmatrix} 64 & 64 \\ 64 & -64 \end{bmatrix}, \quad R_2 = \begin{bmatrix} 36 & 83 \\ -83 & 36 \end{bmatrix}$$



$$A_4 \times Q_A$$

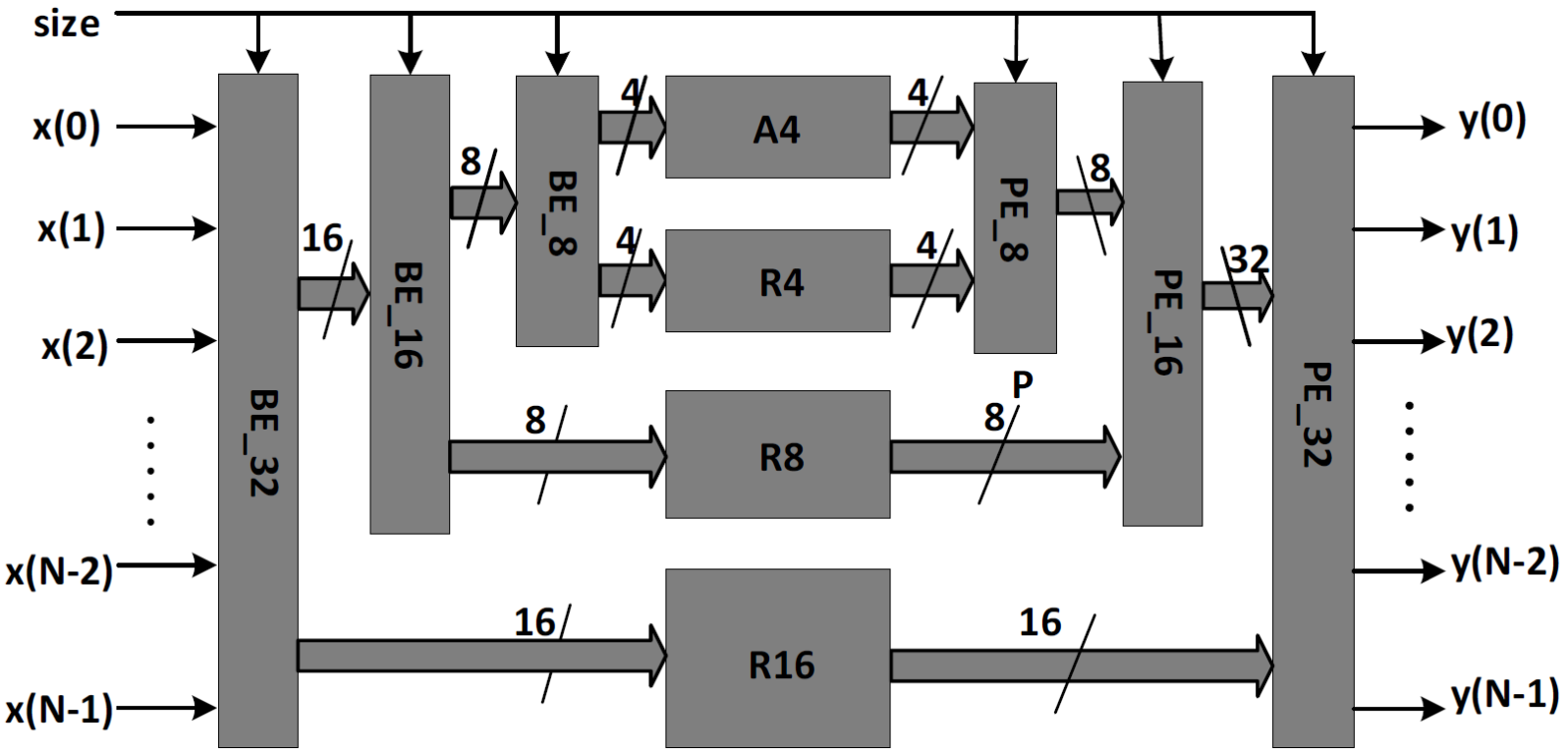
1D-DCT的结构

$$P_8 * X_8^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} x_0 \\ x_4 \\ x_1 \\ x_5 \\ x_2 \\ x_6 \\ x_3 \\ x_7 \end{bmatrix}$$



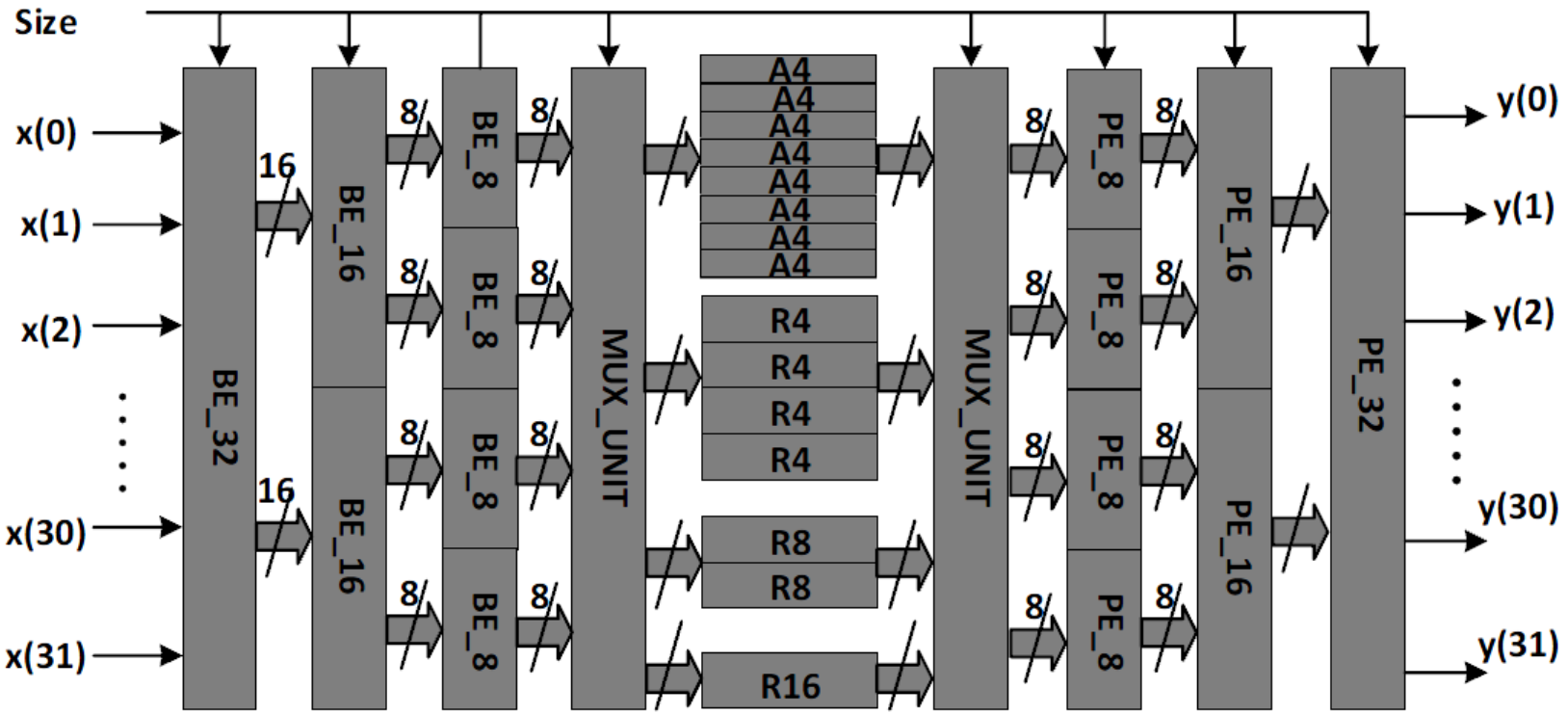
$$P_N \times H_N$$

1D-DCT的结构



若简单连接各个模块，一次只能进行一次矩阵变换

1D-DCT的结构



为提高吞吐量，可计算多个尺寸小于 32 的向量

1D-DCT的复用

$$Y_N = A_N \times X_N^T$$
$$= P_N \times \begin{bmatrix} A_{N/2} & 0 \\ 0 & R_{N/2} \end{bmatrix} \times B_N \times X_N^T$$
$$= P_N \times \begin{bmatrix} A_{N/2} \times Q_A \\ R_{N/2} \times Q_S \end{bmatrix}$$
$$= P_N \times H_N$$

DCT的算法

$$Y_N = A_N^T \times X_N^T \qquad *B_N^T = B_N$$
$$= B_N \times \begin{bmatrix} A_{N/2}^T & 0 \\ 0 & R_{N/2}^T \end{bmatrix} \times P_N^T \times X_N^T$$
$$= B_N \times \begin{bmatrix} A_{N/2}^T \times Q_0 \\ R_{N/2}^T \times Q_1 \end{bmatrix}$$
$$= B_N \times G_N$$

IDCT的算法

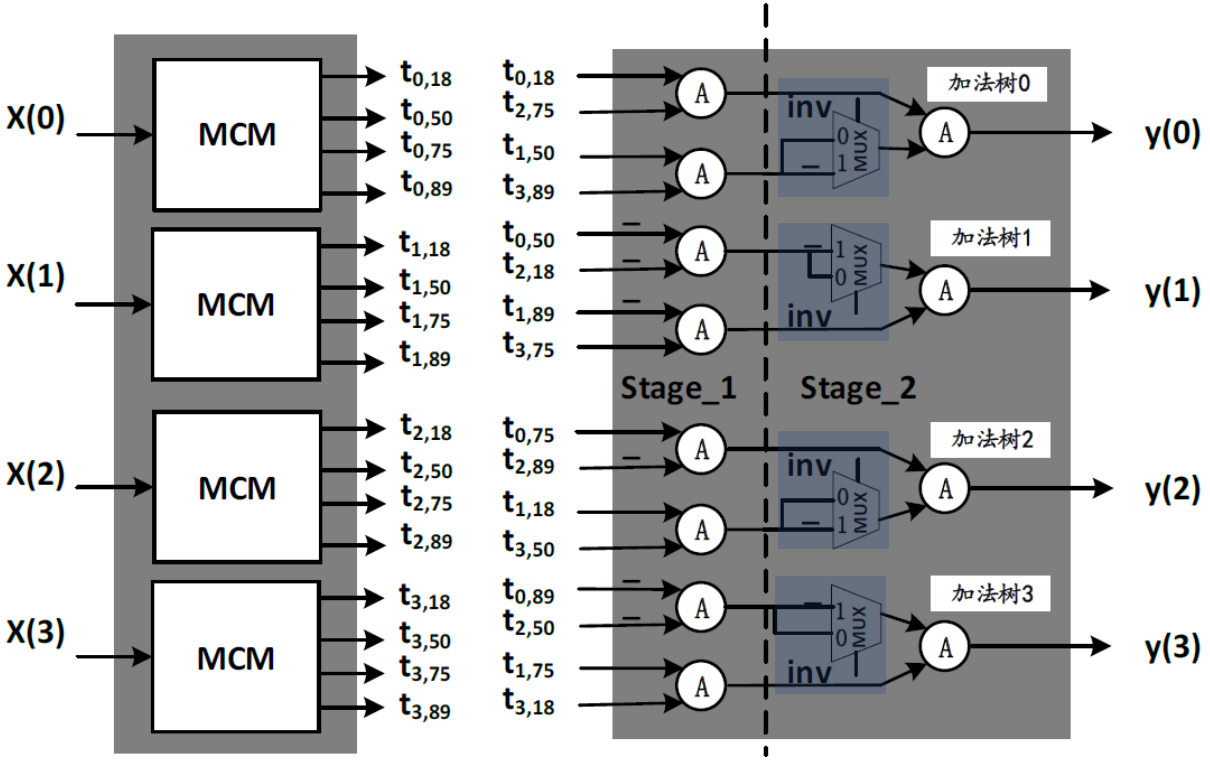
1D-DCT的复用

$$\begin{bmatrix} 18 & 50 & 75 & 89 \\ -50 & -89 & -18 & 75 \\ 75 & 18 & -89 & 50 \\ -89 & 75 & -50 & 18 \end{bmatrix}$$

$$R_4$$

$$\begin{bmatrix} 18 & -50 & 75 & -89 \\ 50 & -89 & 18 & 75 \\ 75 & -18 & -89 & -50 \\ 89 & 75 & 50 & 18 \end{bmatrix}$$

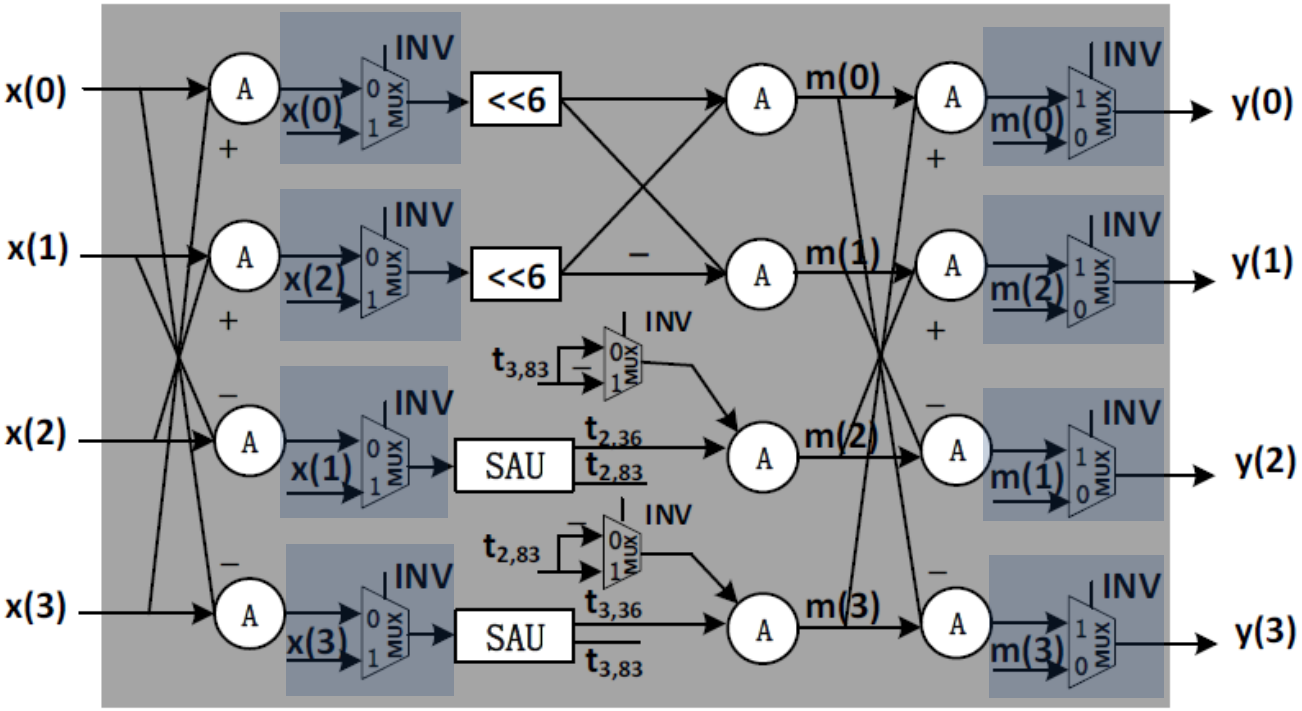
$$R_4^T$$



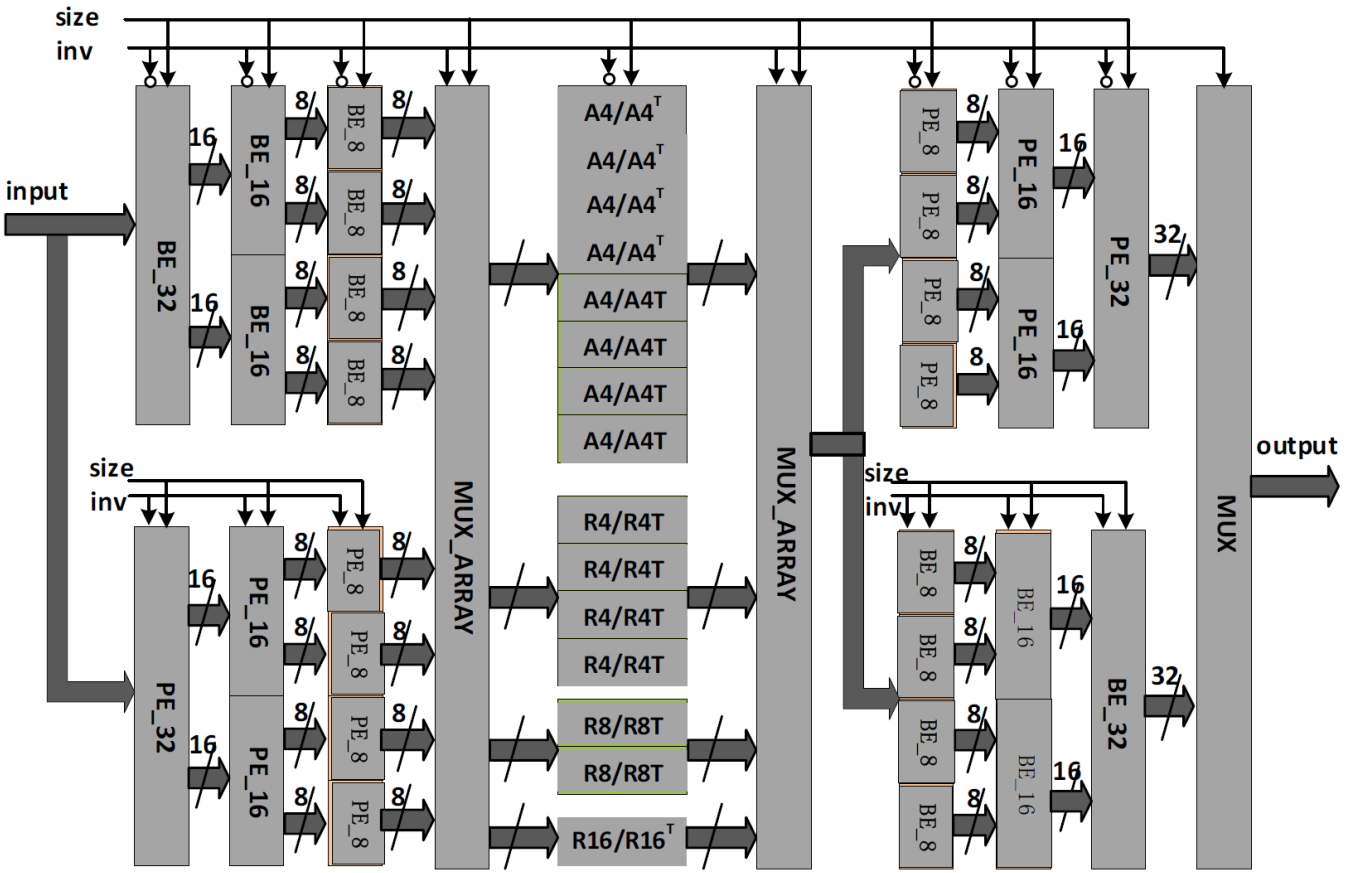
$R_{2/N}$乘法的复用

1D-DCT的复用

$$A_4^T = B_4 \times \begin{bmatrix} A_2^T & 0 \\ 0 & R_2^T \end{bmatrix} \times P_4^T, \quad A_2^T = \begin{bmatrix} 64 & 64 \\ 64 & -64 \end{bmatrix}, \quad R_2^T = \begin{bmatrix} 36 & -83 \\ 83 & 36 \end{bmatrix}$$

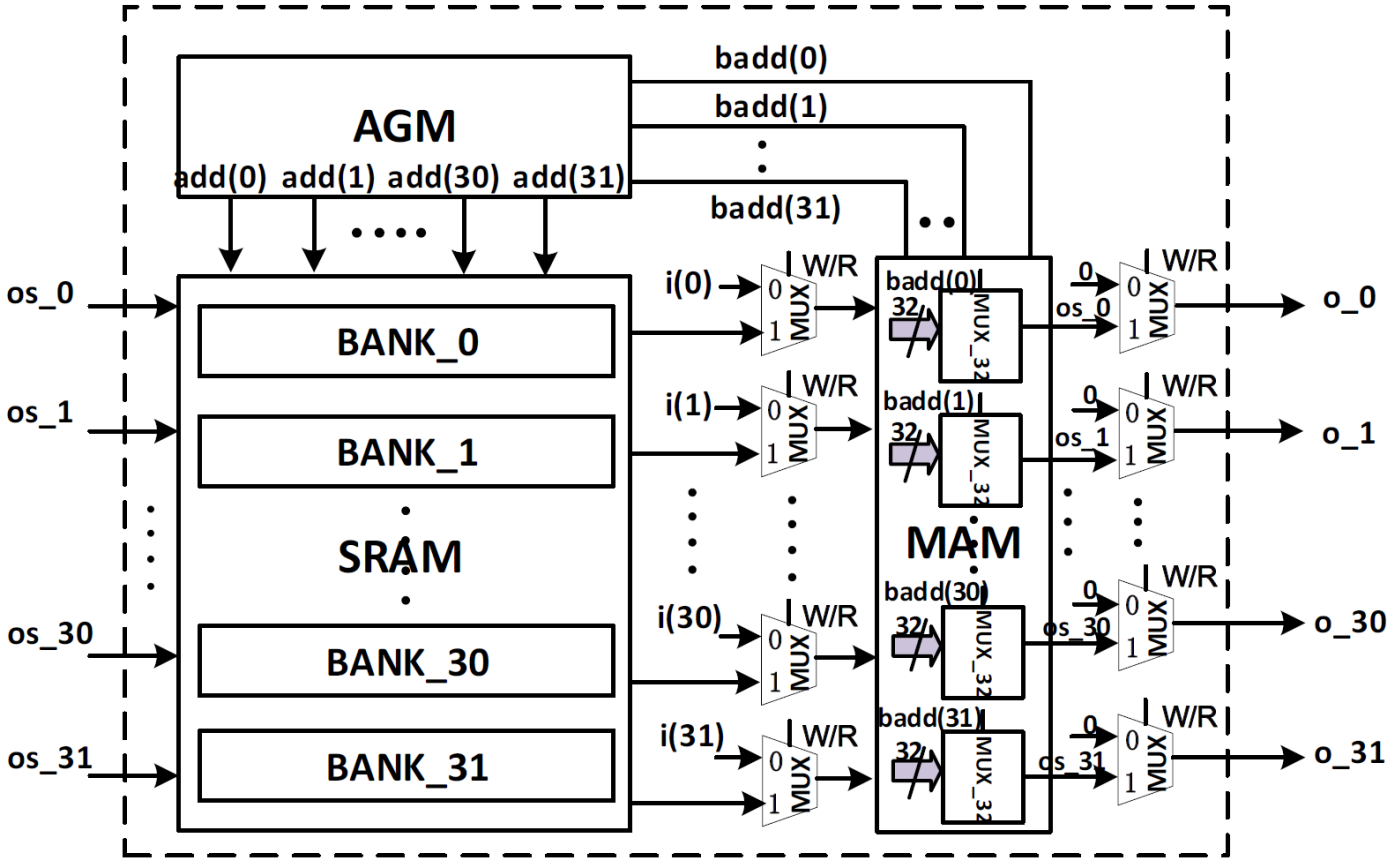


1D-DCT的复用



1D-DCT/IDCT 的高吞吐量复用结构

转置存储器结构



SRAM：存储矩阵数据

AGM：生成映射地址，完成矩阵转置

MAM：对数据排序

量化的算法

量化

$$output = (input \times Q + offset) >> (29 + QP/6 - M - B)$$

$$Q = f(QP \ \% \ 6),\ f(x) = \begin{cases} 26214, x = 0 \\ 23302, x = 1 \\ 20560, x = 2 \\ 18396, x = 3 \\ 16384, x = 4 \\ 14564, x = 5 \end{cases} \qquad offset = A_1 << A_2$$

B: Bit Depth

N: 变换尺寸

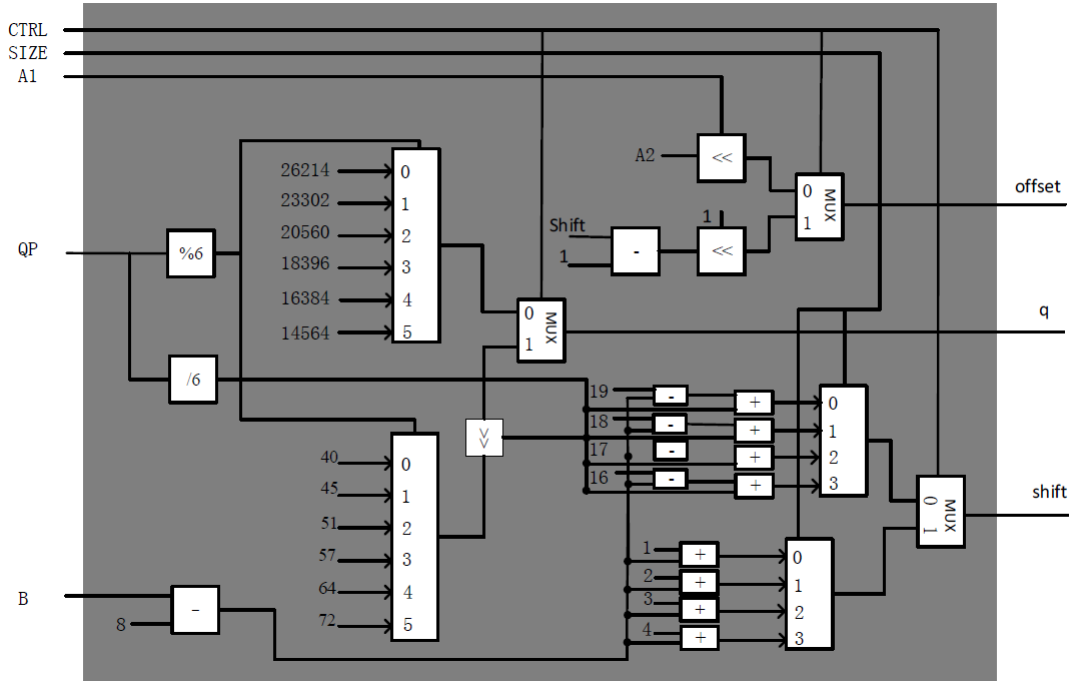
M: $\log_2(N)$

反量化

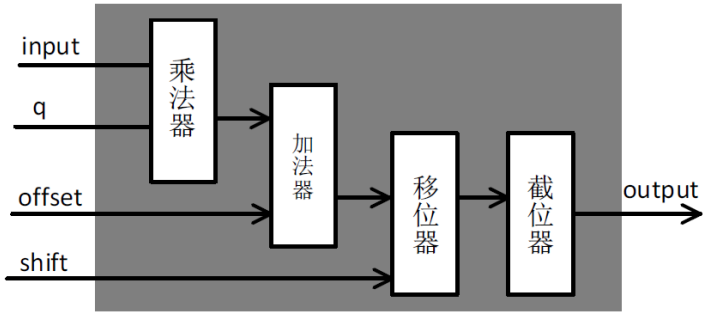
$$output = \min\left[32767, \max(-32768, \left\{\left[(input \times Q) << (QP/6) + offset\right] >> (M + B - 9)\right\})\right]$$

$$offset = 1 << (M + B - 10)$$

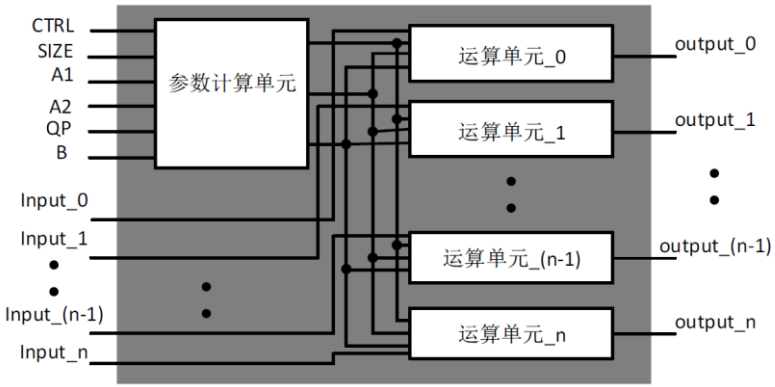
量化的结构



量化的参数计算单元



量化的运算单元



量化的整体结构

DCT与量化部分的外部接口

type_i: 0为intra模式，1为inter模式

tq_sel_i: 0x为亮度块，10为C_b块，11为C_r块

tq_size_i: TU尺寸，00为4x4，01为8x8，10为16x16，11为32x32

tq_idx_i: 残差输入计数器，最高计数值32

tq_cef_i: 输入的残差系数，每周期32点

cef_data_i: 编码数据输入

rec_idx_o: 输出计数器，最高计数值32

rec_data_o: 重建回路参差系数输出，每周期32点

cef_widx_o: 输出计数器，最高计数值32

cef_data_o: 编码码流输出，每周期32点

Testbench

输入数据: line 1: QP
line 2: TU_size
line 3-line NxN: 残差矩阵

经过TQ模块处理后，与check文件进行比对。输入数据与check数据均可由HM生成

参考资料

1.《HEVC 视频编码器中重建环路模块的VLSI 实现研究》，谢峥，范益波
2.《High Efficiency Video Coding: Algorithms and Architectures》，V. Sze, etc.
3.《High Efficiency Video Coding: Coding Tools and Specification》，M. Wien
4.《Perceptual Image Coding with Discrete Cosine Transform》, E. Tan, etc.

ASIC

专注开源硬件 IP Core

www.openasic.org

谢谢！