Case Study: DCT

Shao-Yi Chien

Most slides are prepared by C.J.Lian,

DSP/IC Design Lab.

Outline

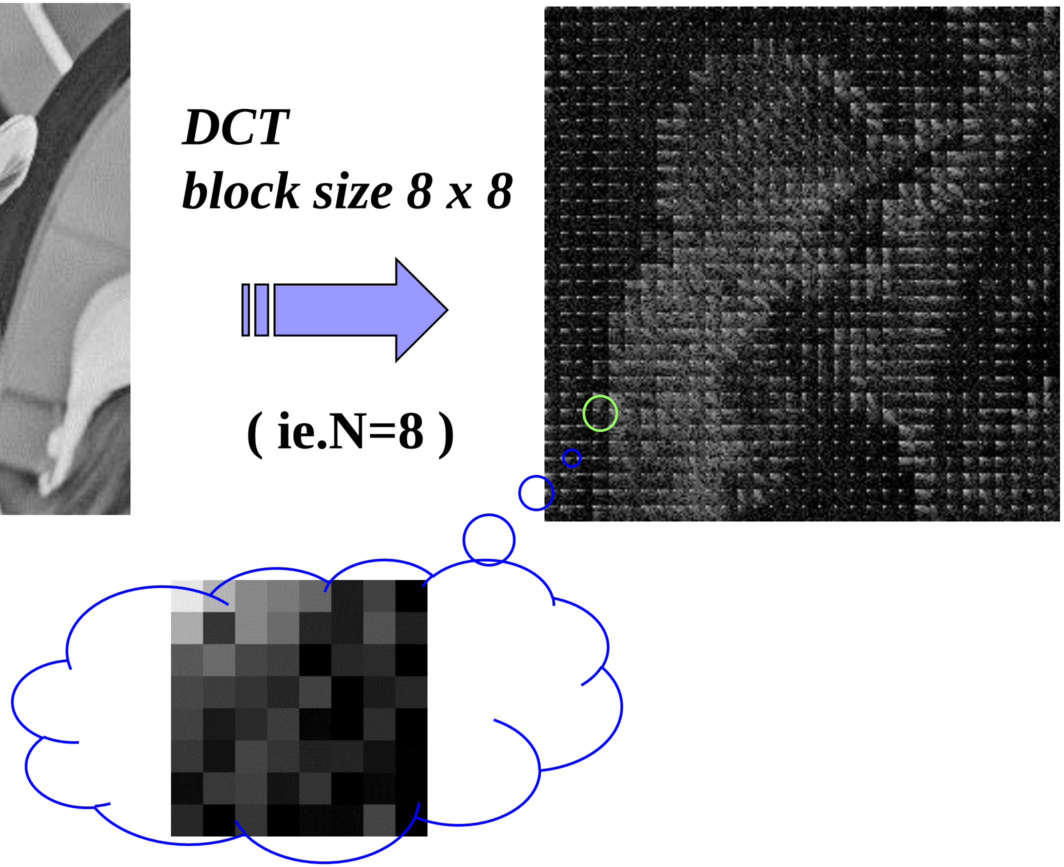
- DCT Algorithm
- 1-D DCT : Row-Column Method
- Direct 2-D Architecture

DCT Algorithm

Discrete Cosine Transform

DCT
block size 8 x 8

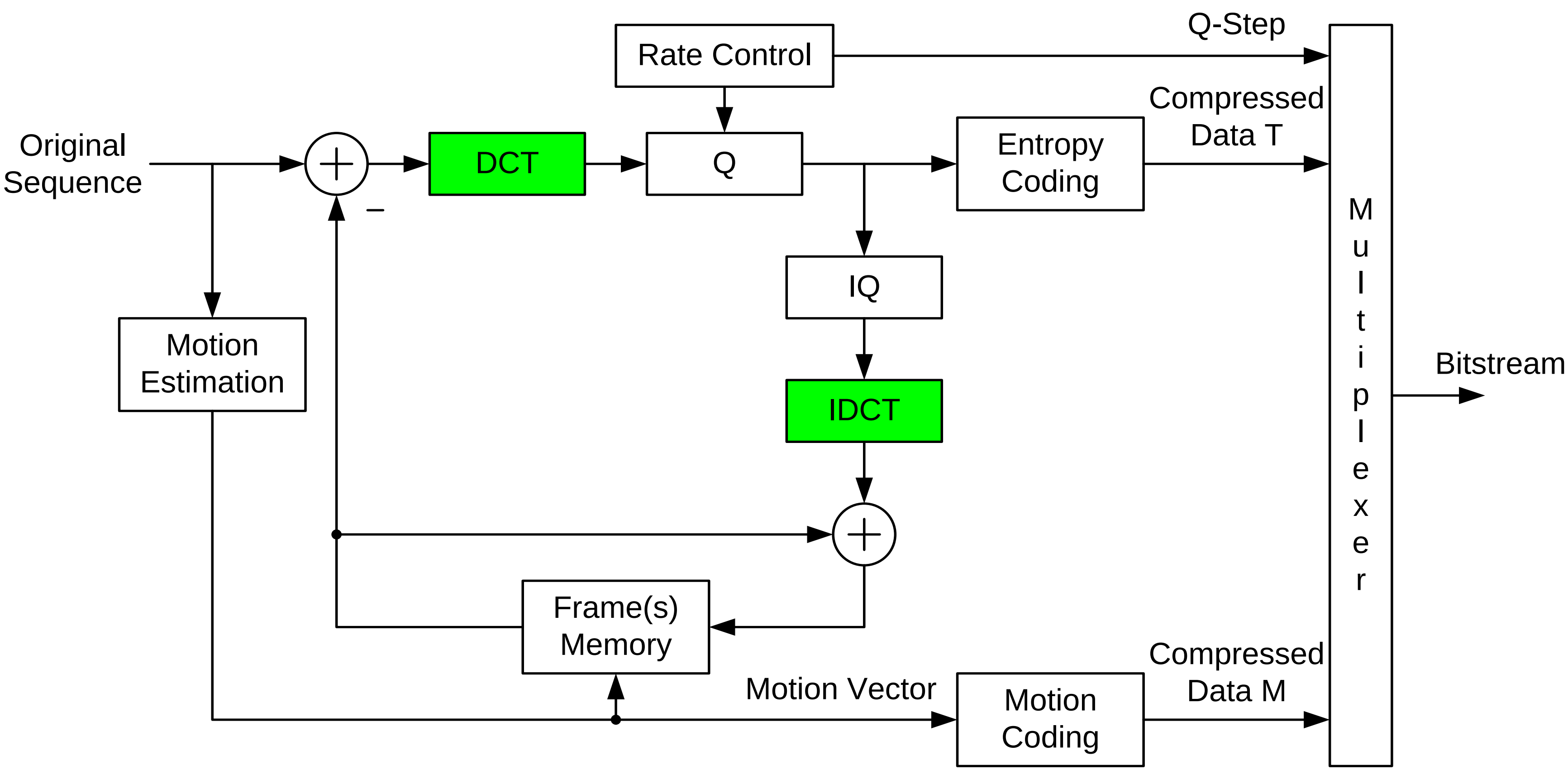
(ie.N=8)



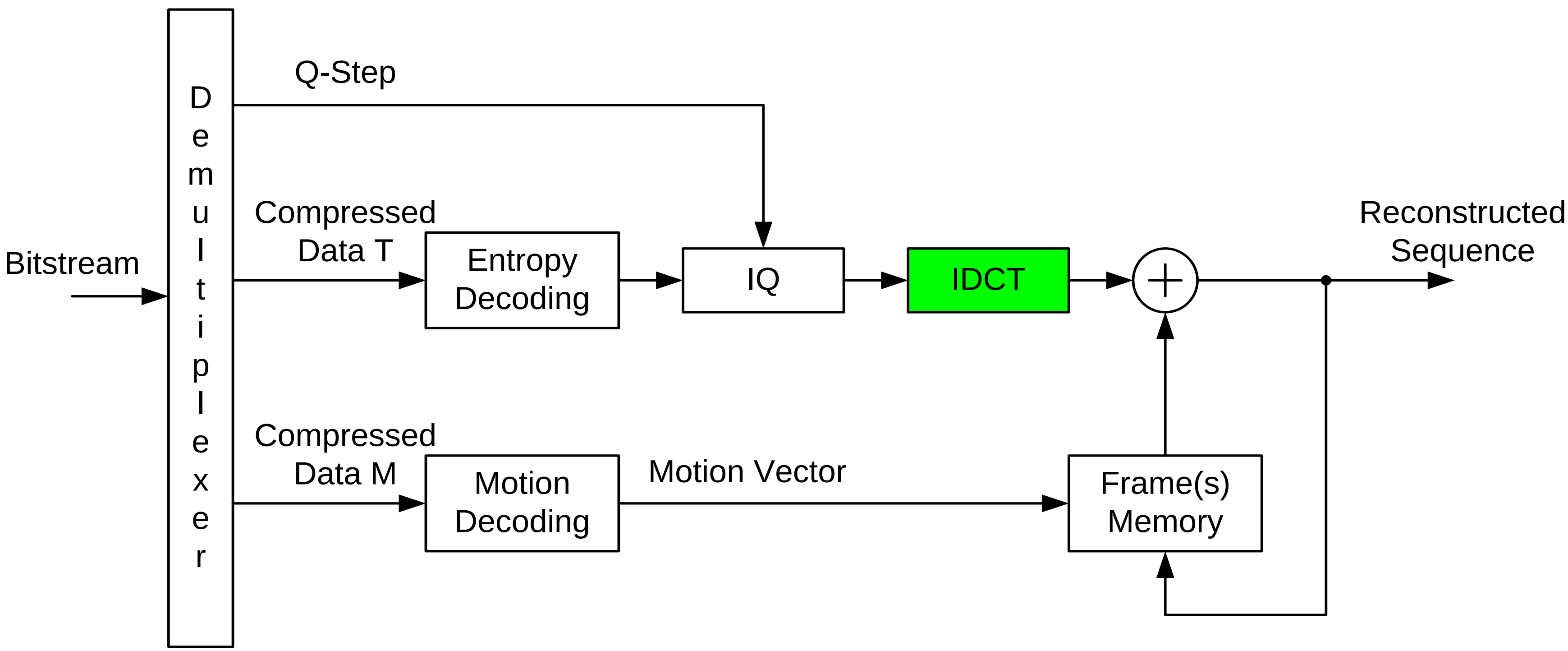
DCT-Based Coding

- Optimal transform is KLT, but
  - KLT is image dependent
  - High computing complexity
- DCT-based coding,
  - Image independent, unlike KLT for highly correlated image data
  - DCT compaction efficiency is close to KLT
  - Computations of DCT can be performed with fast algorithms which can be easily implemented on parallel architectures.

Roles in Video Encoder



Roles in Video Decoder



Hardware/Software Trade-Off

- For **low-end** applications, using software approach is powerful enough
- For **high-end** applications, must use hardware approach
- For **middle-end** applications, either software or hardware approach is possible, depending on the target design platform

Basic Transformation Forms

- 2-D forward transforms

$$T(u,v)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} g(x,y)\,f(x,y,u,v)$$

- 2-D inverse transforms

$$g(x,y)=\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} T(u,v)\,i(x,y,u,v)$$

1-D Discrete Cosine Transform

- Forward DCT

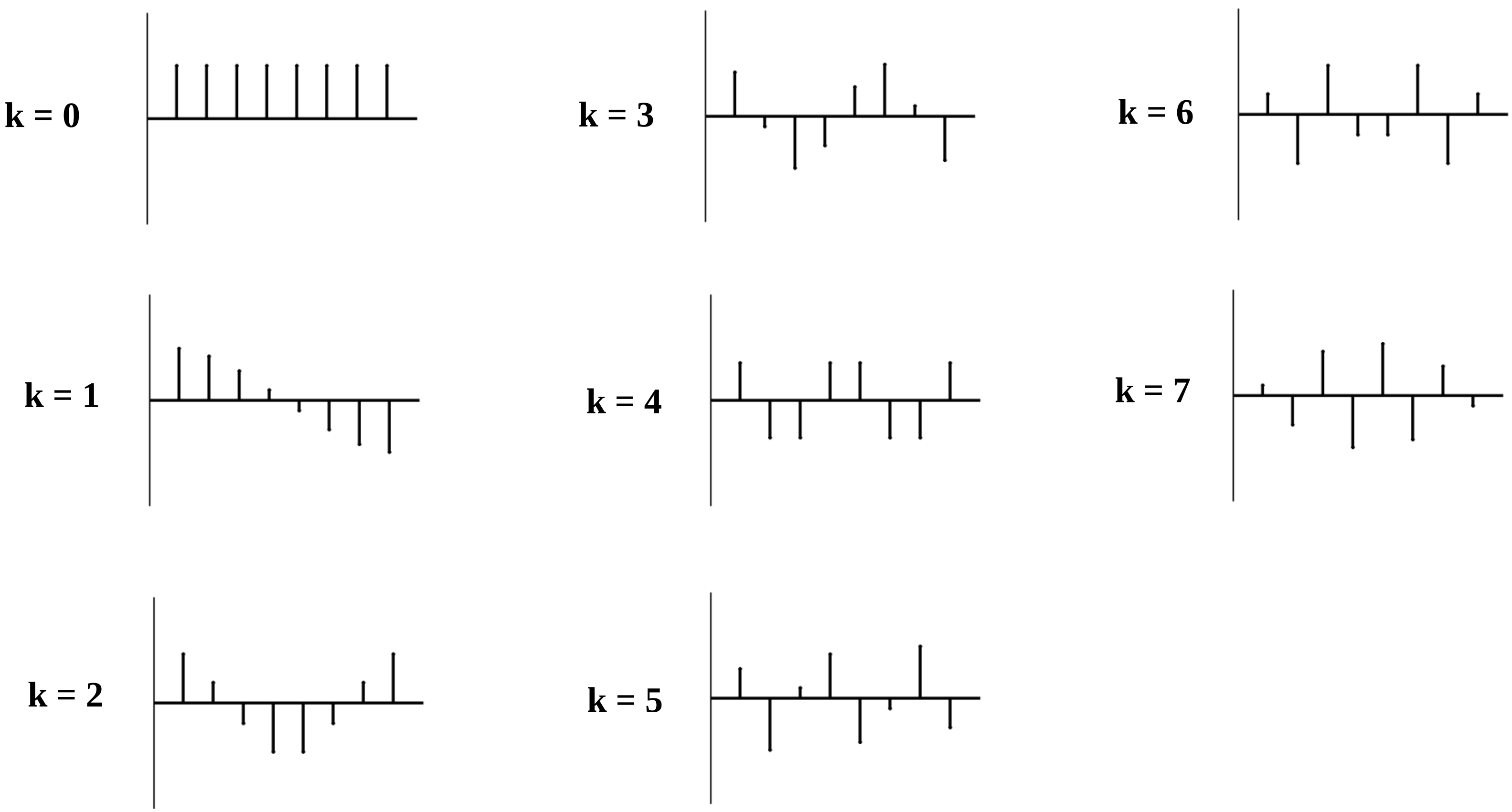
$$X(m) = u(m)\sqrt{\frac{2}{N}}\sum_{i=0}^{N-1} x(i)\cos\frac{(2i+1)m\pi}{2N}, \text{ for } m = 0, 1, \ldots, N-1,$$

$$\text{where } u(m) = \begin{cases} 1 & \text{for } m = 0; \\ \frac{1}{\sqrt{2}} & \text{otherwise.} \end{cases}$$

- Backward DCT

$$x(i) = \sqrt{\frac{2}{N}}\sum_{m=0}^{N-1} u(m)X(m)\cos\frac{(2i+1)m\pi}{2N}.$$

1D 8-Point DCT Basis Functions



2-D Discrete Cosine Transform

- Forward DCT

$$X(m,n) = u(m)u(n)\frac{2}{N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} x(i,j)\cos\frac{(2i+1)m\pi}{2M}\cos\frac{(2j+1)n\pi}{2N},$$

for $m = 0, 1, \ldots, M-1, n = 0, 1, \ldots, N-1.$

$$\text{where } u(m), u(n) = \begin{cases} 1/\sqrt{2}, & \text{for } m, n = 0 \\ 1, & \text{otherwise} \end{cases}$$

- Backward DCT

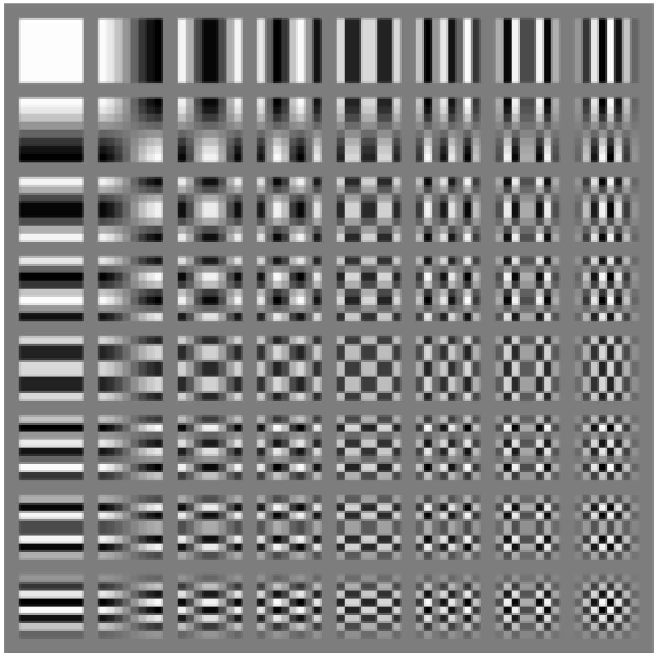
$$x(i,j) = \frac{2}{N}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} u(m)u(n)X(m,n)\cos\frac{(2i+1)m\pi}{2M}\cos\frac{(2j+1)n\pi}{2N},$$

for $i = 0, 1, \ldots, M-1, j = 0, 1, \ldots, N-1.$

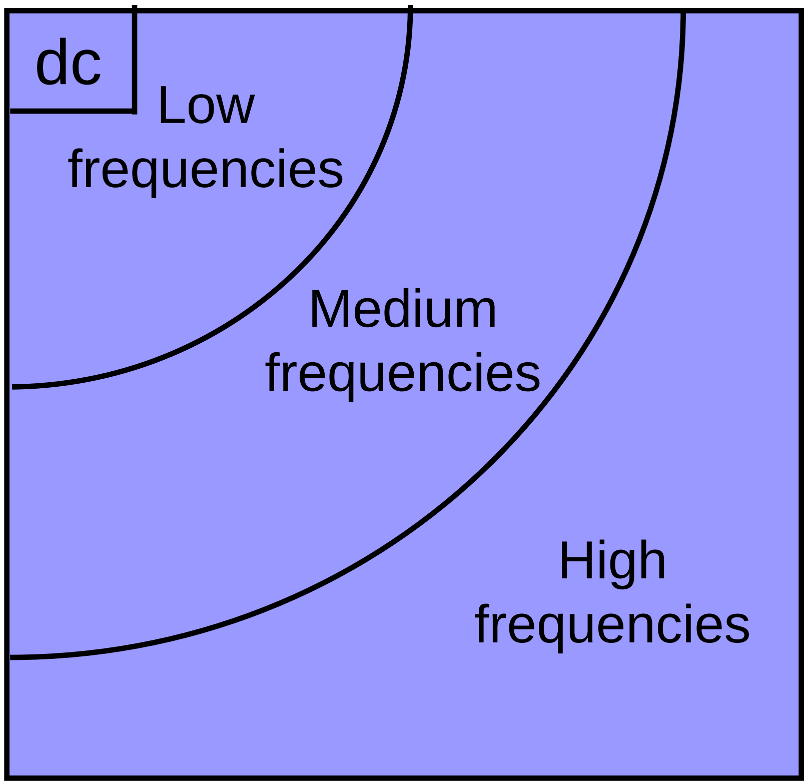
Block size = N x N
$X(m,n)$: *DCT coefficients*
$x(i,j)$: *image samples in block*

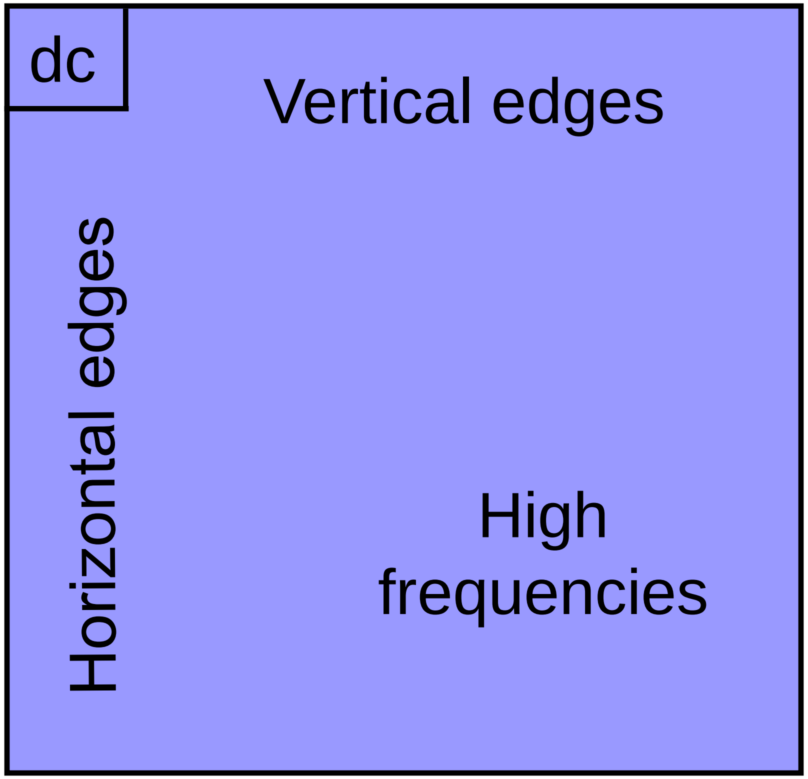
2-D 8x8 DCT Basis Functions

- The DCT represents each block of image samples as a weighted sum of 2-D cosine functions (basis functions)

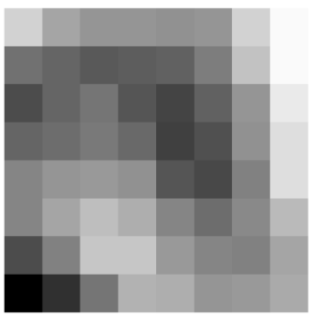


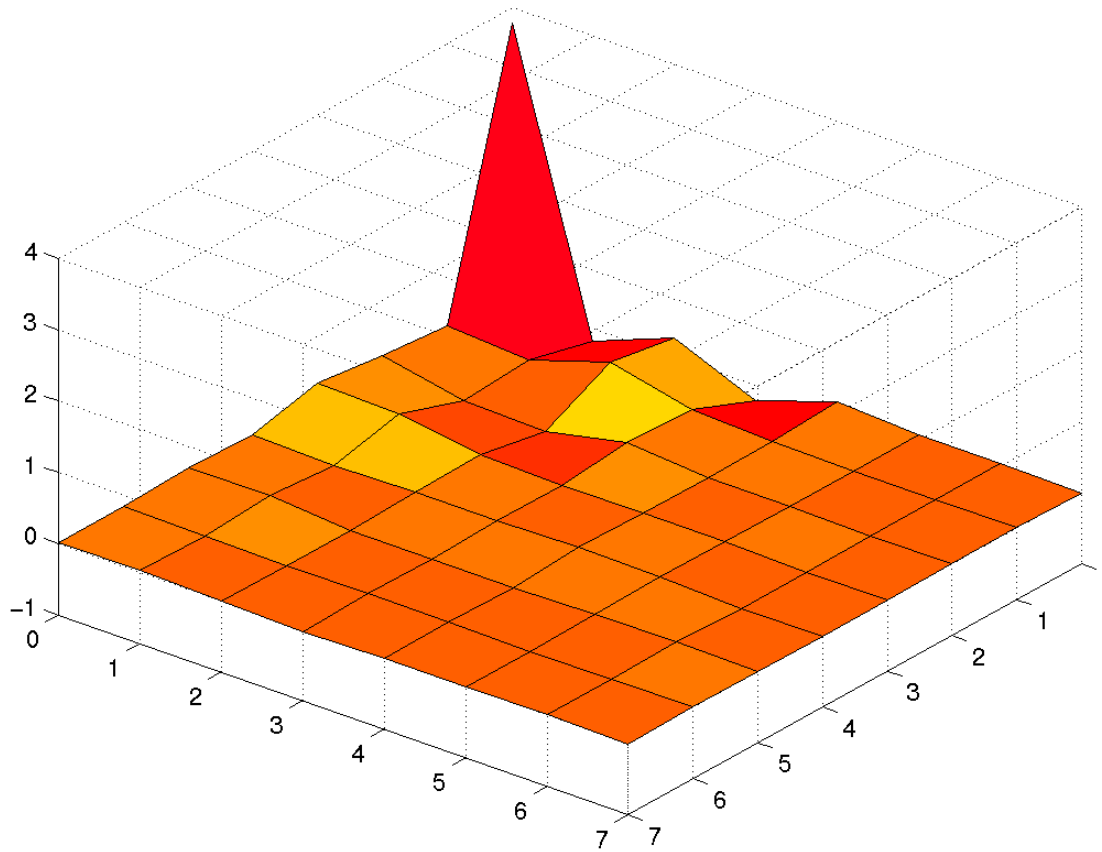
DCT Coefficients





An Example of Energy Compaction





An Example of DCT

52	55	61	66	71	61	64	73
63	59	66	90	109	585	69	72
62	59	68	113	144	104	66	73
63	58	71	122	154	106	70	69
67	61	68	104	126	88	68	70
79	65	60	70	77	68	58	75
85	71	64	59	55	61	65	83
87	79	69	58	65	76	78	94

Pixel Domain

DCT ↓ ↑ IDCT

DC →

AC →

609	-29	-62	25	55	-20	-1	3
7	-21	-62	9	11	-7	-6	6
-46	8	77	-25	-30	10	7	-5
-50	13	35	-15	-9	6	0	3
11	-8	-13	-2	-1	1	-4	1
-10	1	3	-3	-1	0	2	-1
-4	-1	2	-1	2	-3	1	-2
-1	-1	-1	-2	-1	-1	0	-1

Frequency Domain

DC: $F(0,0) = (1/8) \Sigma\Sigma f(m,n)$

related to the average value of the block

DCT Algorithm Classification

- **Direct 2-D Method**
  - The 2-D transforms, DCT and IDCT, to be applied directly on the N x N input data items
- **Row-Column Method**
  - The 2-D transform can be carried out with two passes of 1-D transforms
  - The separability property of 2-D DCT/IDCT allows the transform to be applied on one dimension (row) then on the other (column)
  - Requires 2N instances of N-point 1-D DCT to implement an N x N 2-D DCT

Row-Column Decomposition

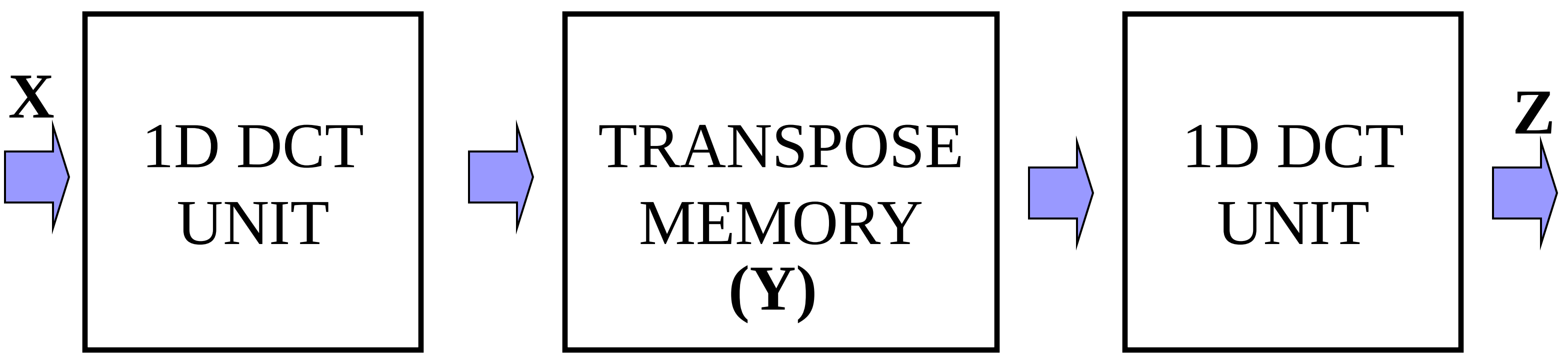
Separable,
row-column decomposition

$$Z = AXA^T$$

$$a(k,n) = \sqrt{\frac{2}{N}}\, c(k) \cos\left[\frac{2\pi(2n+1)k}{4N}\right]$$

$$k, n = 0, 1, \ldots, N-1$$

$$c(0) = \sqrt{\frac{1}{2}} \quad \text{and} \quad c(k) = 1 \quad \text{for} \quad k \neq 0$$



$\boldsymbol{Y=AX}$ $\qquad$ $\boldsymbol{Z=YA^T}$

Straightforward Approach

- Carry out the computation as full matrix-vector multiplications
  - 1-D transform requires N*N multiplications and N* (N-1) additions
  - 2-D transform requires N*N*N*N multiplications and N * N *(N * N-1) additions
  - Although requiring the most number of operations, this method is very regular
  - Most suitable for vector processors or deeply pipelined architectures for high PE utilization
  - **1-D fast algorithms => O(N*logN)**
  - **2-D fast algorithms => O(N*N*logN)**

DCT Definition

DCT

$$X(k) = e(k) \sum_{n=0}^{N-1} x(n) \cos\left[\frac{(2n+1)\ \pi\ k}{2N}\right], \ k = 0,1\ \ldots\ N-1$$

IDCT

$$x(n) = \frac{2}{N} \sum_{k=0}^{N-1} e(k) X(k) \cos\left[\frac{(2n+1)\ \pi\ k}{2N}\right], \ n = 0,1\ \ldots\ N-1$$

$$e(k) = \frac{1}{\sqrt{2}}\ \text{if}\ \ k = 0,\ e(k) = 1\ \ \text{othervise}$$

Example: 4-point DCT (N=4)

$$X(k)=e(k)\sum_{n=0}^{3}x(n)\cos\left[\frac{(2n+1)\ \pi\ \ k}{8}\right],\ k=0,1,2,3$$

$$X(0)=\frac{1}{\sqrt{2}}\sum_{n=0}^{3}x(n)=\frac{1}{\sqrt{2}}x(0)+\frac{1}{\sqrt{2}}x(1)+\frac{1}{\sqrt{2}}x(2)+\frac{1}{\sqrt{2}}x(3)$$

$$X(1)=\sum_{n=0}^{3}x(n)\cos\left[\frac{(2n+1)\ \pi}{8}\right]=x(0)\cos\frac{\pi}{8}+x(1)\cos\frac{3\pi}{8}+x(2)\cos\frac{5\pi}{8}+x(3)\cos\frac{7\pi}{8}$$

$$X(2)=\ldots\ldots$$

$$X(3)=\ldots\ldots$$

Coefficients

4-point DCT - Matrix Form

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \cos\frac{\pi}{8} & \cos\frac{3\pi}{8} & \cos\frac{5\pi}{8} & \cos\frac{7\pi}{8} \\ \cos\frac{2\pi}{8} & \cos\frac{6\pi}{8} & \cos\frac{10\pi}{8} & \cos\frac{14\pi}{8} \\ \cos\frac{3\pi}{8} & \cos\frac{9\pi}{8} & \cos\frac{15\pi}{8} & \cos\frac{21\pi}{8} \end{bmatrix} \times \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \cos\frac{\pi}{8} & \cos\frac{3\pi}{8} & -\cos\frac{3\pi}{8} & -\cos\frac{\pi}{8} \\ \cos\frac{2\pi}{8} & -\cos\frac{2\pi}{8} & -\cos\frac{2\pi}{8} & \cos\frac{2\pi}{8} \\ \cos\frac{3\pi}{8} & -\cos\frac{\pi}{8} & \cos\frac{\pi}{8} & -\cos\frac{3\pi}{8} \end{bmatrix} \times \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

Symmetric or anti-symmetric rows

4-point DCT - Matrix Form

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \cos\frac{\pi}{8} & \cos\frac{3\pi}{8} & -\cos\frac{3\pi}{8} & -\cos\frac{\pi}{8} \\ \cos\frac{2\pi}{8} & -\cos\frac{2\pi}{8} & -\cos\frac{2\pi}{8} & \cos\frac{2\pi}{8} \\ \cos\frac{3\pi}{8} & -\cos\frac{\pi}{8} & \cos\frac{\pi}{8} & -\cos\frac{3\pi}{8} \end{bmatrix} \times \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} c_2 & c_2 & c_2 & c_2 \\ c_1 & c_3 & -c_3 & -c_1 \\ c_2 & -c_2 & -c_2 & c_2 \\ c_3 & -c_1 & c_1 & -c_3 \end{bmatrix} \times \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

N-1 (3) Coefficients

C_1, C_2, C_3

4-point DCT

Define New Variables

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} c_2 & c_2 & c_2 & c_2 \\ c_1 & c_3 & -c_3 & -c_1 \\ c_2 & -c_2 & -c_2 & c_2 \\ c_3 & -c_1 & c_1 & -c_3 \end{bmatrix} \times \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$X(0) = \left[x(0) + x(3) + x(1) + x(2)\right] \times c_2$$

$$X(1) = \left[x(0) - x(3)\right] \times c_1 + \left[x(1) - x(2)\right] \times c_3$$

$$X(2) = \left[x(0) + x(3) - \left(x(1) + x(2)\right)\right] \times c_2$$

$$X(3) = \left[x(0) - x(3)\right] \times c_3 - \left[x(1) - x(2)\right] \times c_1$$

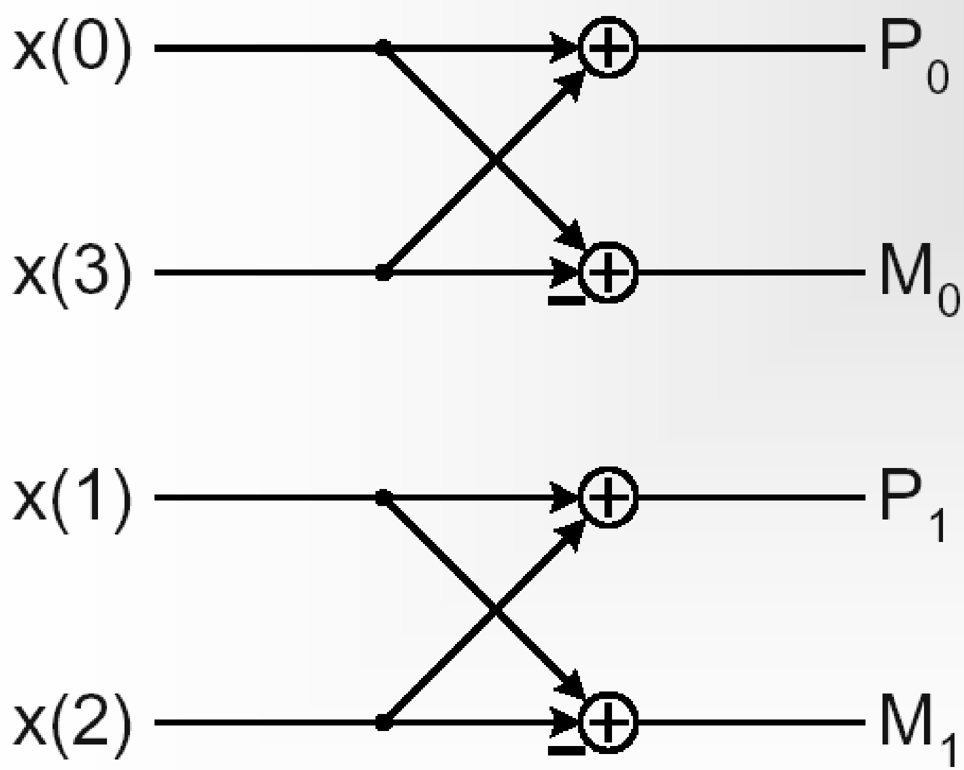
4-point DCT

16 Mult reduced to 6

$$\begin{cases} X(0)=[x(0)+x(3)+x(1)+x(2)]\times c_2 \\ X(1)=[x(0)-x(3)]\times c_1+[x(1)-x(2)]\times c_3 \\ X(2)=[x(0)+x(3)]\times c_2-[x(1)+x(2)]\times c_2 \\ X(3)=[x(0)-x(3)]\times c_3-[x(1)-x(2)]\times c_1 \end{cases}$$

$$\begin{cases} X(0)=[P_0+P_1]\times c_2 \\ X(1)=M_0\times c_1+M_1\times c_3 \\ X(2)=[P_0-P_1]\times c_2 \\ X(3)=M_0\times c_3-M_1\times c_1 \end{cases}, \text{ where } \begin{cases} P_0=x(0)+x(3) \\ M_0=x(0)-x(3) \\ P_1=x(1)+x(2) \\ M_1=x(1)-x(2) \end{cases}$$

Butterfly: First DCT Stage



$$P_0 = x(0) + x(3)$$

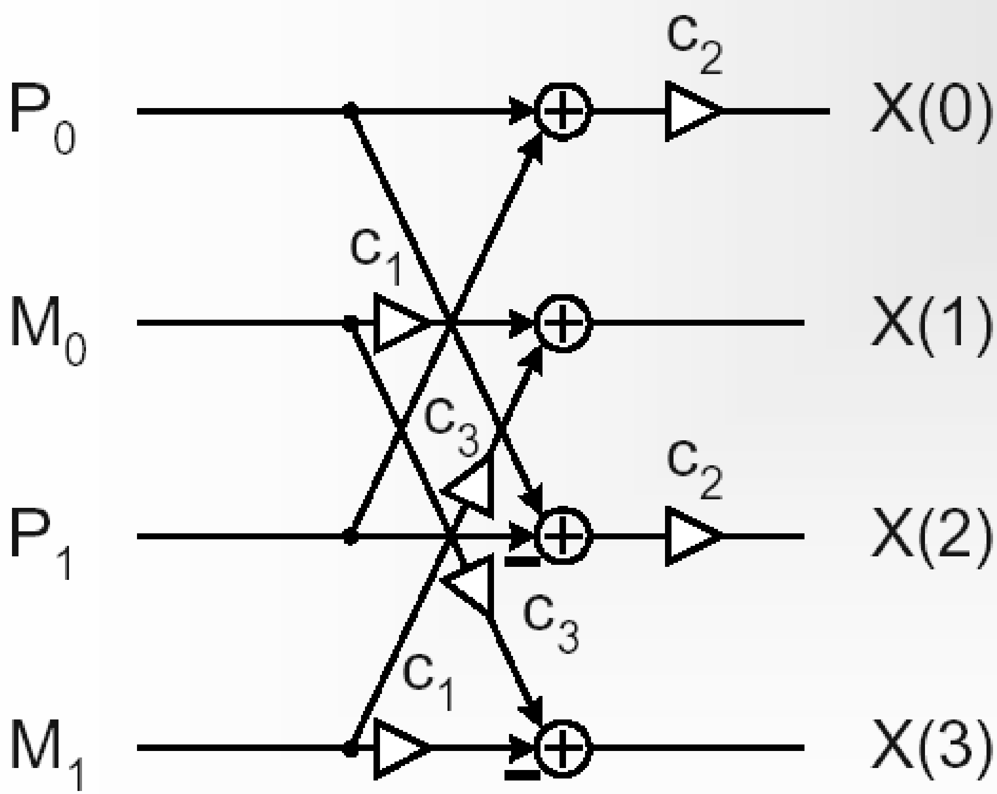
$$M_0 = x(0) - x(3)$$

$$P_1 = x(1) + x(2)$$

$$M_1 = x(1) - x(2)$$

Reversed input order

Butterfly: Second stage



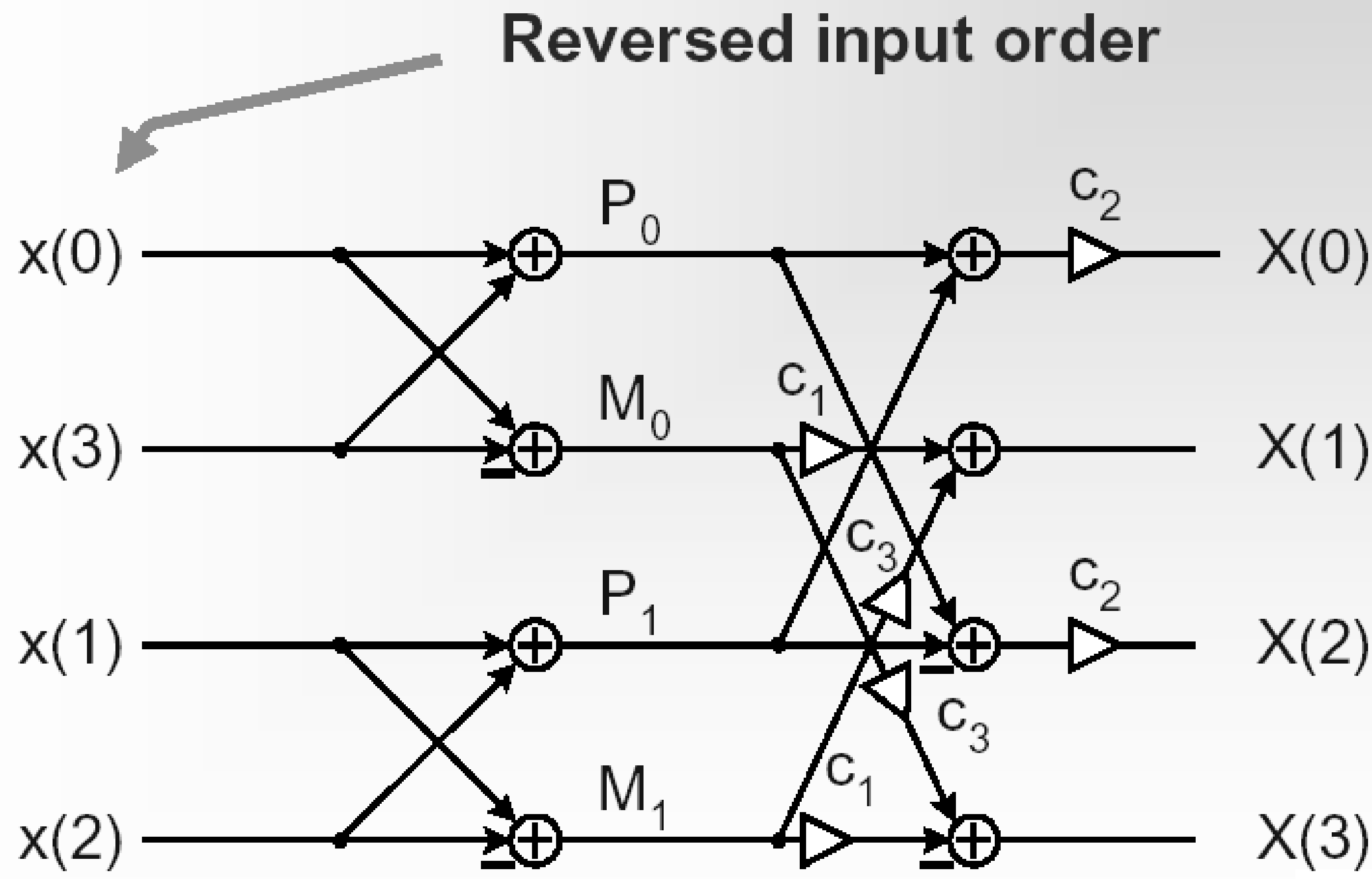
$$X(0)=[P_0+P_1]\times c_2$$

$$X(1)=M_0\times c_1+M_1\times c_3$$

$$X(2)=[P_0-P_1]\times c_2$$

$$X(3)=M_0\times c_3-M_1\times c_1$$

4-point DCT


Reversed input order
x(0)
x(3)
x(1)
x(2)
P0
M0
P1
M1
c1
c3
c3
c1
c2
c2
X(0)
X(1)
X(2)
X(3)

8-point DCT

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \\ X(7) \end{bmatrix} = \begin{bmatrix} c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 \\ c_1 & c_3 & c_5 & c_7 & c_9 & c_{11} & c_{13} & c_{15} \\ c_2 & c_6 & c_{10} & c_{14} & c_{18} & c_{22} & c_{26} & c_{30} \\ c_3 & c_9 & c_{15} & c_{21} & c_{27} & c_1 & c_7 & c_{13} \\ c_4 & c_{12} & c_{20} & c_{28} & c_4 & c_{12} & c_{20} & c_{28} \\ c_5 & c_{15} & c_{25} & c_3 & c_{13} & c_{23} & c_1 & c_{11} \\ c_6 & c_{18} & c_{30} & c_{10} & c_{22} & c_2 & c_{14} & c_{26} \\ c_7 & c_{21} & c_3 & c_{17} & c_{31} & c_{13} & c_{27} & c_9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ x(5) \\ x(6) \\ x(7) \end{bmatrix}$$

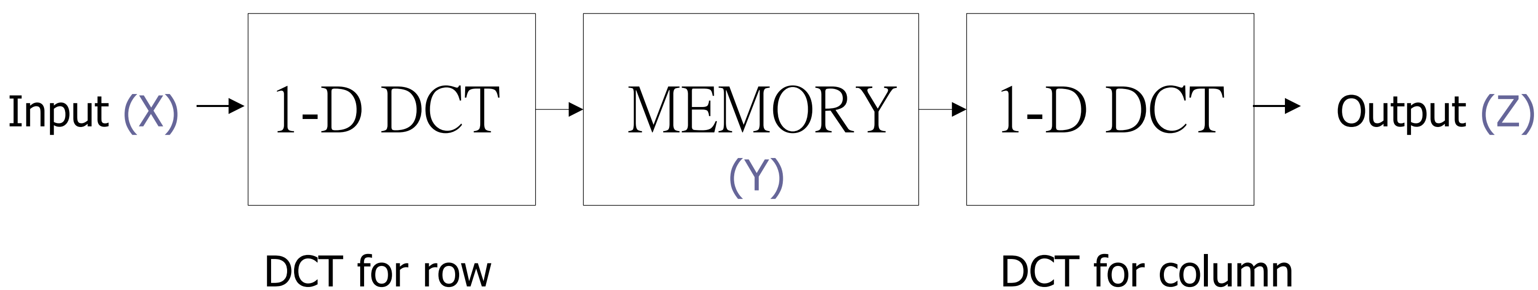
$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \\ X(7) \end{bmatrix} = \begin{bmatrix} c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 \\ c_1 & c_3 & c_5 & c_7 & -c_7 & -c_5 & -c_3 & -c_1 \\ c_2 & c_6 & -c_6 & -c_2 & -c_2 & -c_6 & c_6 & c_2 \\ c_3 & -c_7 & -c_1 & -c_5 & c_5 & c_1 & c_7 & -c_3 \\ c_4 & -c_4 & -c_4 & c_4 & c_4 & -c_4 & -c_4 & c_4 \\ c_5 & -c_1 & c_7 & c_3 & -c_3 & -c_7 & c_1 & -c_5 \\ c_6 & -c_2 & c_2 & -c_6 & -c_6 & c_2 & -c_2 & c_6 \\ c_7 & -c_5 & c_3 & -c_1 & c_1 & -c_3 & c_5 & -c_7 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ x(5) \\ x(6) \\ x(7) \end{bmatrix}$$

1-D DCT Row-Column Method

Row-Column Method(1/2)

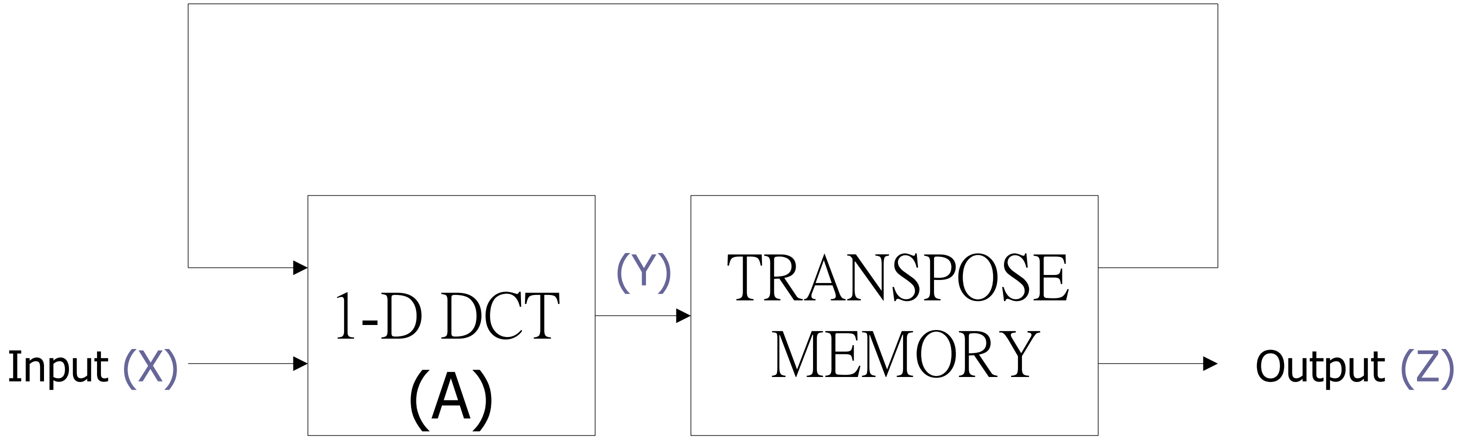
- Basic concept：

2-D DCT = 1-D DCT(row) → 1-D DCT(column)



Row-Column Method(2/2)

- Use transpose memory



$$Z = AXA^T$$

Row-Column Method Example

A. Madisetti and A. N. Willson Jr., “A 100 MHz 2-D 8x8 DCT/IDCT processor for HDTV applications,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 2, April 1995

Matrix Decomposition

- For 8-bit 1D-DCT unit array A:

$$Y = AX$$

$$A = \begin{bmatrix} a & a & a & a & a & a & a & a \\ b & d & e & g & -g & -e & -d & -b \\ c & f & -f & -c & -c & -f & f & c \\ d & -g & -b & -e & e & b & g & -d \\ a & -a & -a & a & a & -a & -a & a \\ e & -b & g & d & -d & -g & b & -e \\ f & -c & c & -f & -f & c & -c & f \\ g & -e & d & -b & b & -d & e & -g \end{bmatrix}$$

$$\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \end{bmatrix} = \sqrt{\frac{2}{N}} \begin{bmatrix} \cos\frac{\pi}{4} \\ \cos\frac{\pi}{16} \\ \cos\frac{\pi}{8} \\ \cos\frac{3\pi}{16} \\ \cos\frac{5\pi}{16} \\ \cos\frac{3\pi}{8} \\ \cos\frac{7\pi}{16} \end{bmatrix}$$

Matrix Decomposition

- Use symmetrical property of DCT coefficients:

DCT：

$$\begin{bmatrix} Y(0) \\ Y(2) \\ Y(4) \\ Y(6) \end{bmatrix} = \begin{bmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{bmatrix} \begin{bmatrix} X(0)+X(7) \\ X(1)+X(6) \\ X(2)+X(5) \\ X(3)+X(4) \end{bmatrix}$$

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \\ Y(7) \end{bmatrix} = \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(0)-X(7) \\ X(1)-X(6) \\ X(2)-X(5) \\ X(3)-X(4) \end{bmatrix}$$

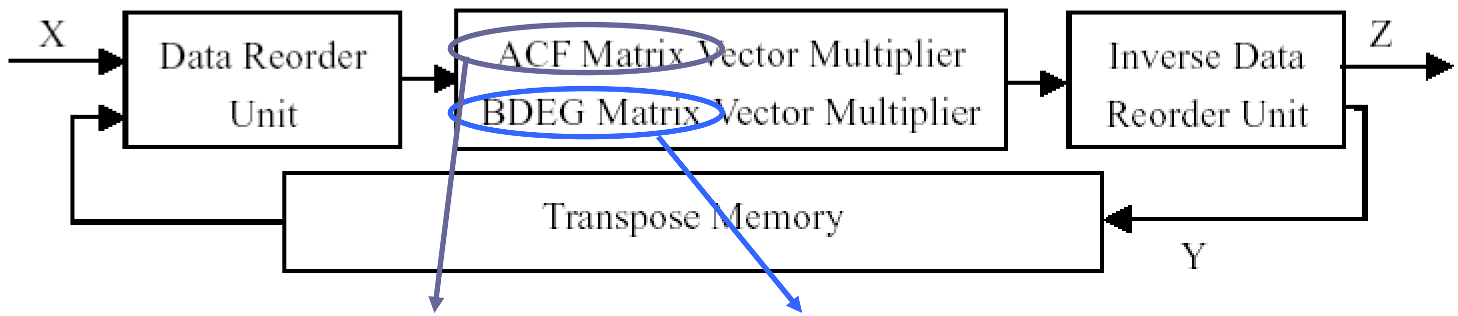
IDCT：

$$\begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ Y(3) \end{bmatrix} = \begin{bmatrix} a & c & a & f \\ a & f & -a & -c \\ a & -f & -a & c \\ a & -c & a & -f \end{bmatrix} \begin{bmatrix} X(0) \\ X(2) \\ X(4) \\ X(6) \end{bmatrix} + \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(1) \\ X(3) \\ X(5) \\ X(7) \end{bmatrix}$$

$$\begin{bmatrix} Y(7) \\ Y(6) \\ Y(5) \\ Y(4) \end{bmatrix} = \begin{bmatrix} a & c & a & f \\ a & f & -a & -c \\ a & -f & -a & c \\ a & -c & a & -f \end{bmatrix} \begin{bmatrix} X(0) \\ X(2) \\ X(4) \\ X(6) \end{bmatrix} - \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(1) \\ X(3) \\ X(5) \\ X(7) \end{bmatrix}$$

Overall Architecture (1/2)

- Architecture:



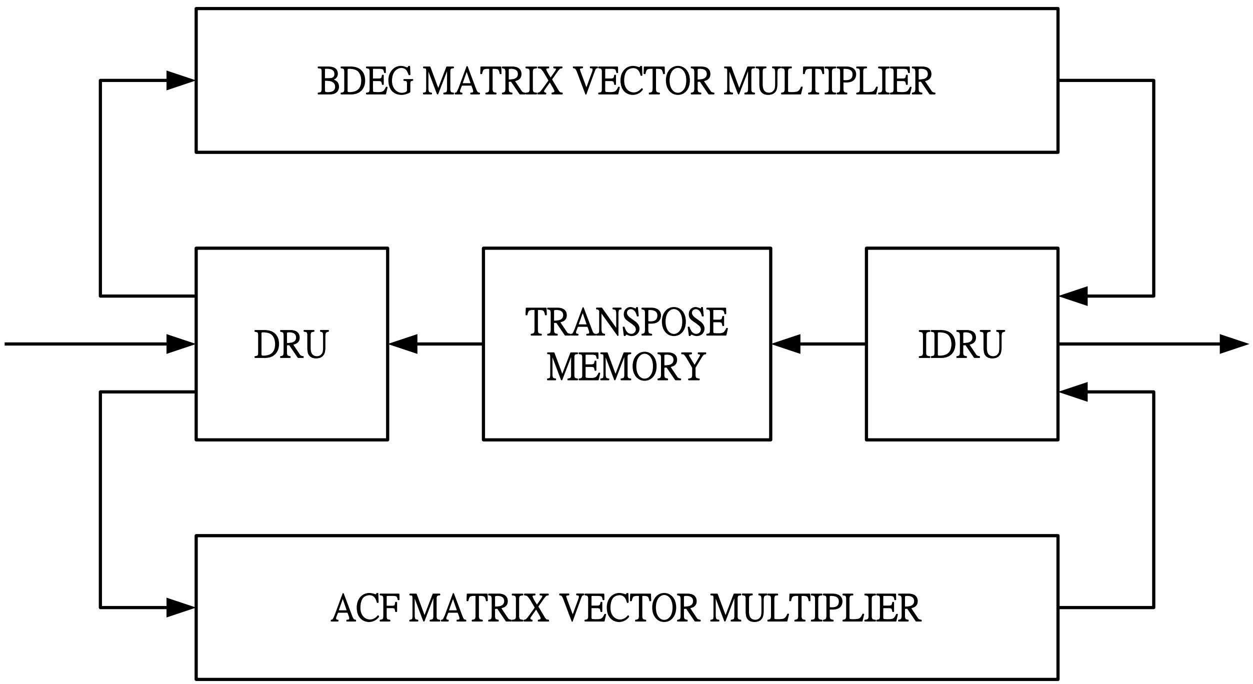
$$\begin{bmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{bmatrix}$$

ACF Matrix

$$\begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix}$$

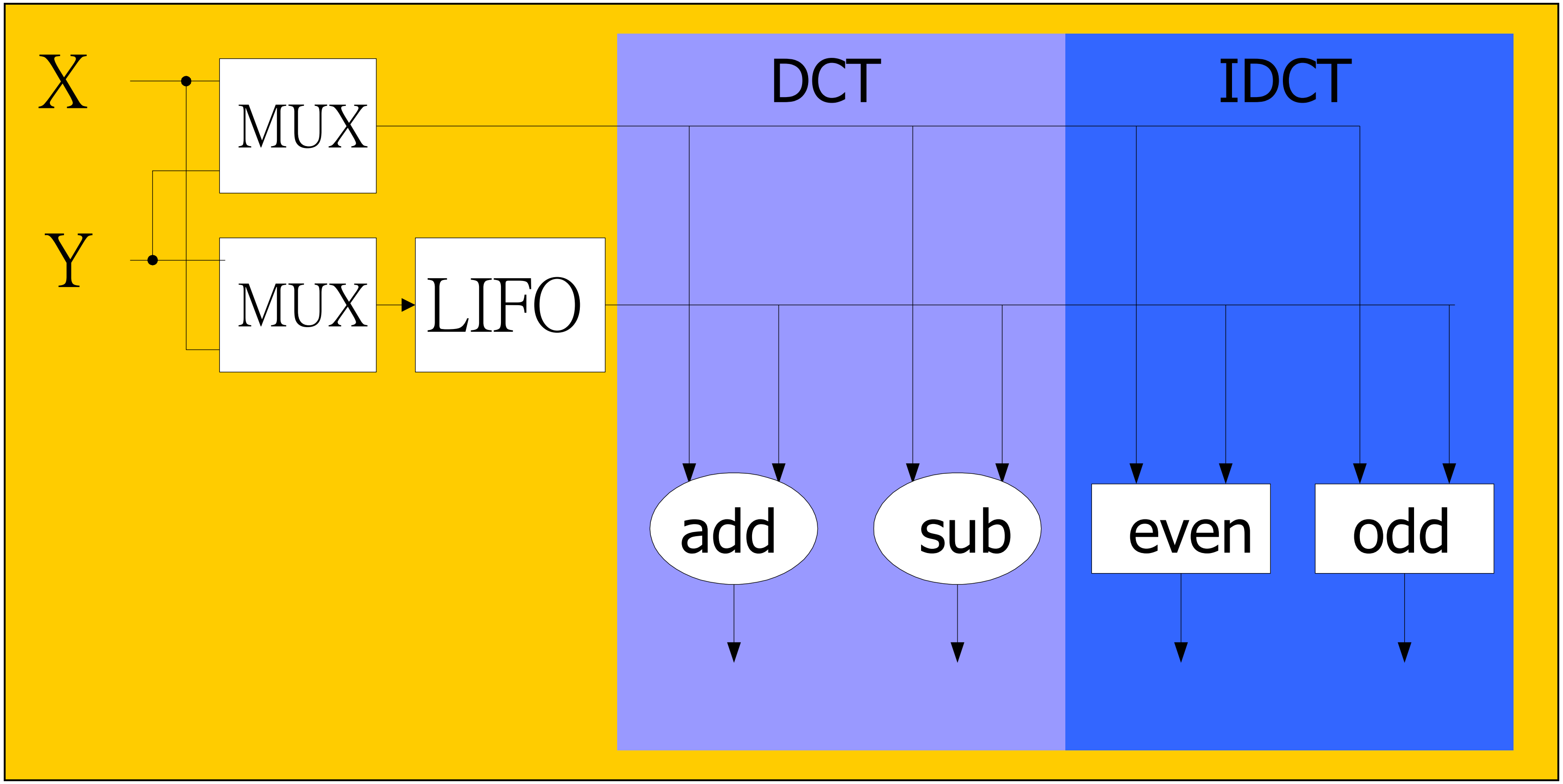
BDEG Matrix

Overall Architecture (2/2)



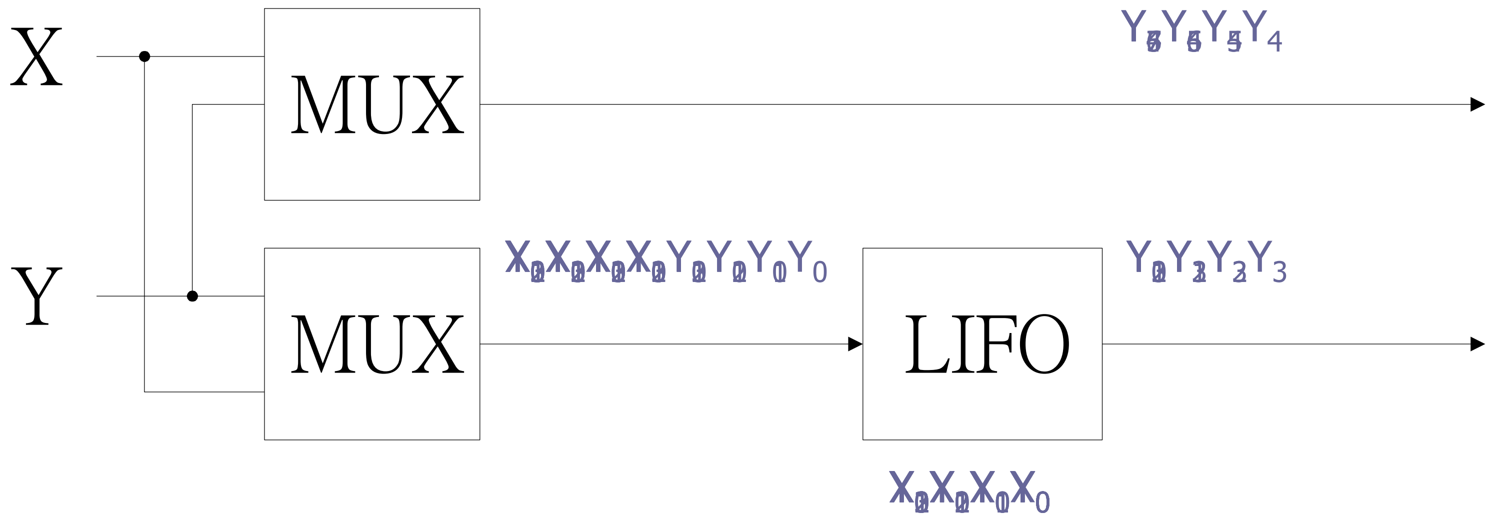
Data Reorder Unit (1/3)

- DRU architecture



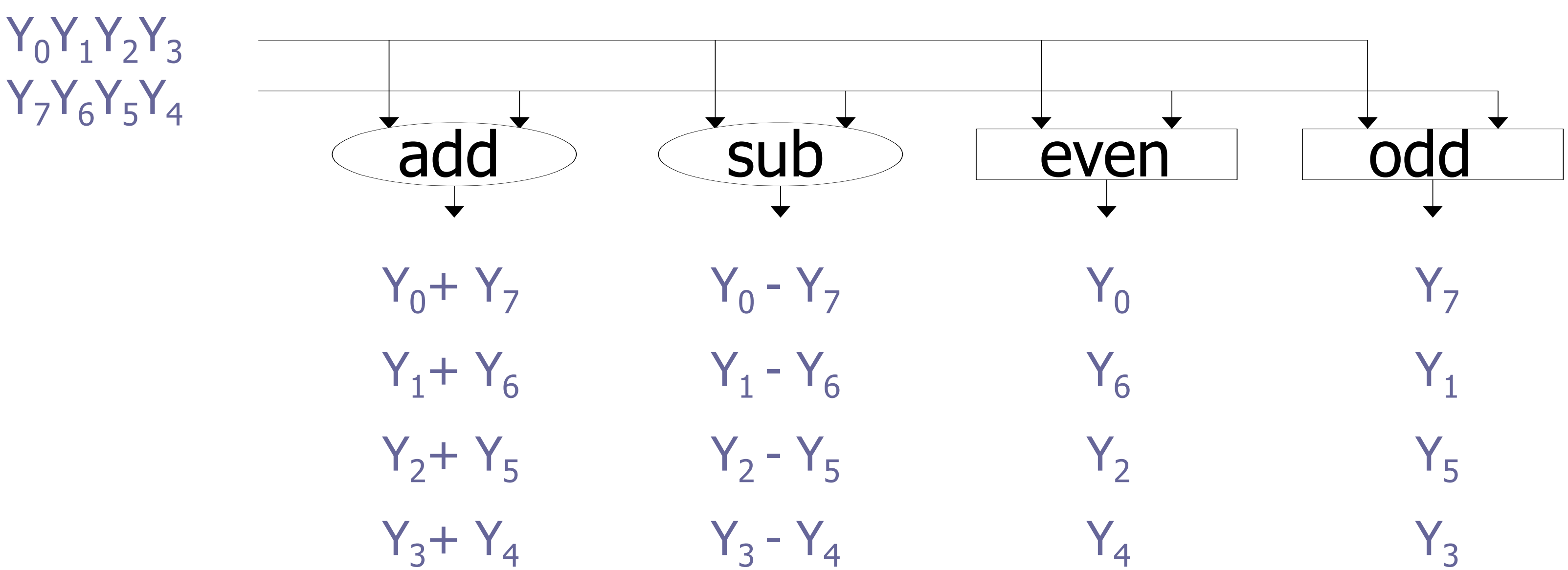
Data Reorder Unit (2/3)

- DRU_1 Data flow:



Data Reorder Unit (3/3)

- DRU_2 Data flow:



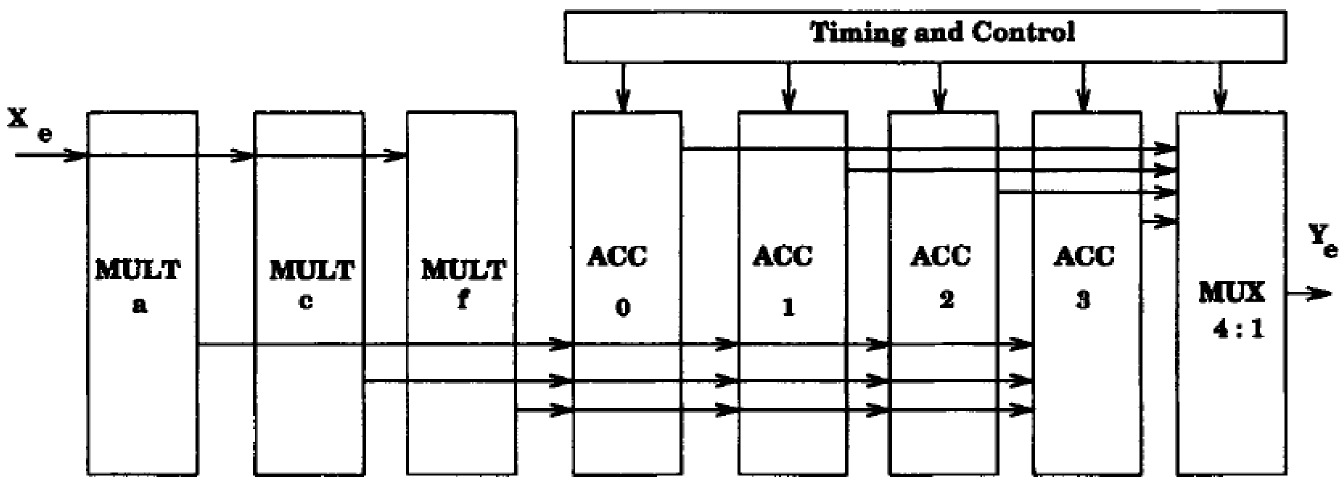
Matrix multiplier (1/4)

- ACF matrix vector multiplier

acc0

$$\begin{bmatrix} Y(0) \\ Y(2) \\ Y(4) \\ Y(6) \end{bmatrix} = \begin{bmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{bmatrix} \begin{bmatrix} X(0)+X(7) \\ X(1)+X(6) \\ X(2)+X(5) \\ X(3)+X(4) \end{bmatrix}$$

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \\ Y(7) \end{bmatrix} = \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(0)-X(7) \\ X(1)-X(6) \\ X(2)-X(5) \\ X(3)-X(4) \end{bmatrix}$$



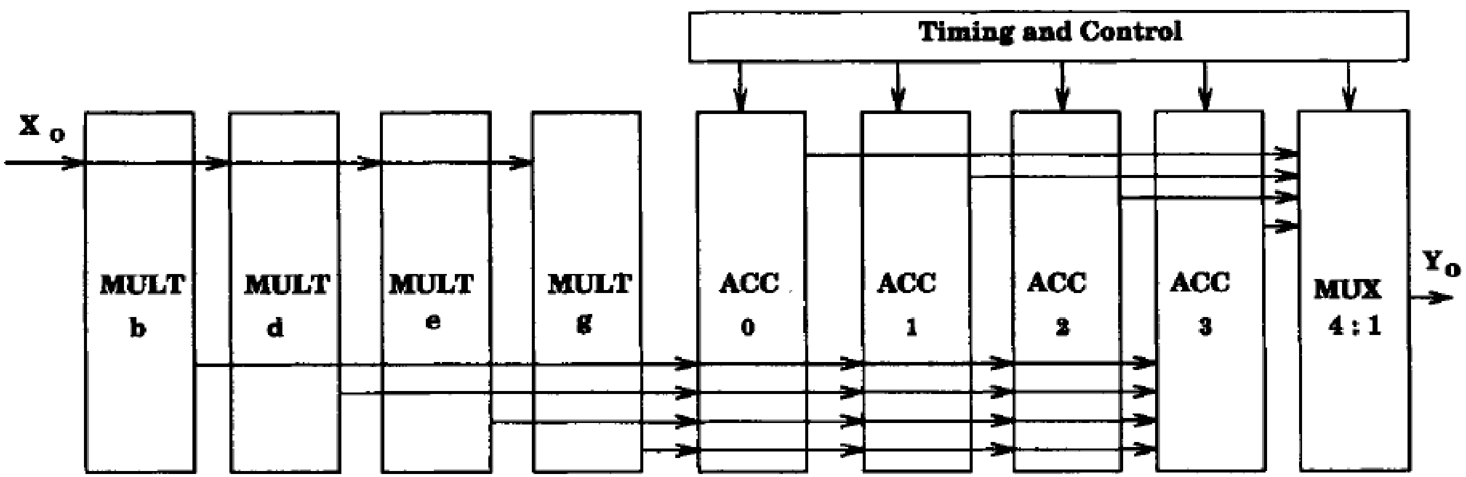
Matrix multiplier (2/4)

- BDEG matrix vector multiplie

$$\begin{bmatrix} Y(0) \\ Y(2) \\ Y(4) \\ Y(6) \end{bmatrix} = \begin{bmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{bmatrix} \begin{bmatrix} X(0)+X(7) \\ X(1)+X(6) \\ X(2)+X(5) \\ X(3)+X(4) \end{bmatrix}$$

acc0

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \\ Y(7) \end{bmatrix} = \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(0)-X(7) \\ X(1)-X(6) \\ X(2)-X(5) \\ X(3)-X(4) \end{bmatrix}$$



Matrix multiplier (3/4)

- Hardwired multipliers
  - Use SDC number system

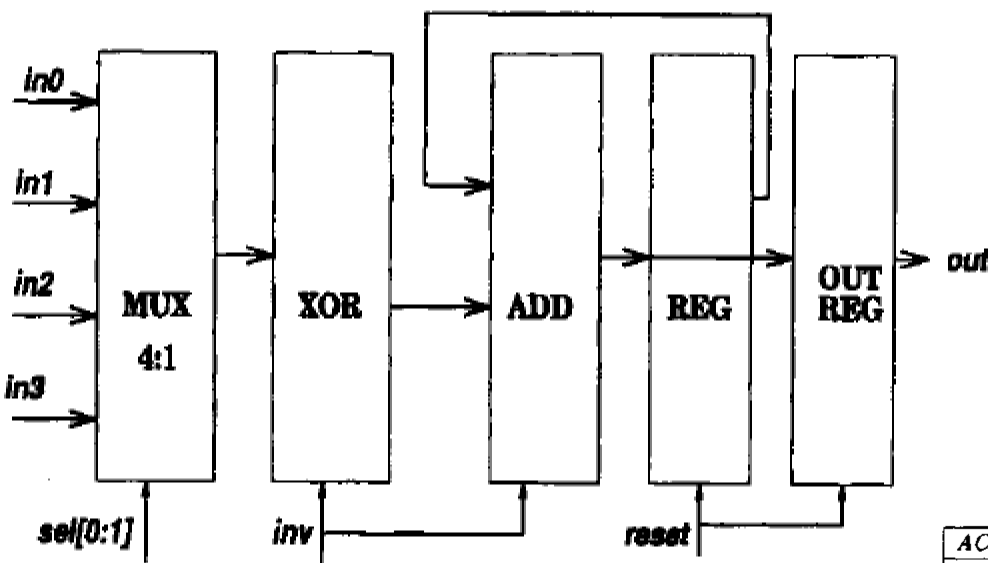
TABLE II

SIGNED DIGIT REPRESENTATION OF THE DCT COEFFICIENTS

coefficient	value	12 bit signed	SD representation	No. bits
a	0.35355	0.35351	$2^{-2}+2^{-4}+2^{-5}+2^{-7}+2^{-9}=0.35352$	5
b	0.49039	0.49023	$2^{-1}-2^{-7}-2^{-9}+2^{-13}=0.49036$	4
c	0.46194	0.46191	$2^{-1}-2^{-5}-2^{-7}+2^{-10}=0.46191$	4
d	0.41573	0.41552	$2^{-2}+2^{-3}+2^{-5}+2^{-7}+2^{-9}-2^{-12}=0.41577$	6
e	0.27779	0.27734	$2^{-2}+2^{-5}-2^{-8}+2^{-11}=0.27783$	4
f	0.19134	0.19091	$2^{-3}+2^{-4}+2^{-8}-2^{-14}=0.19135$	4
g	0.09755	0.09716	$2^{-4}+2^{-5}+2^{-8}-2^{-13}=0.09753$	4

Matrix multiplier (4/4)

- Accumulator



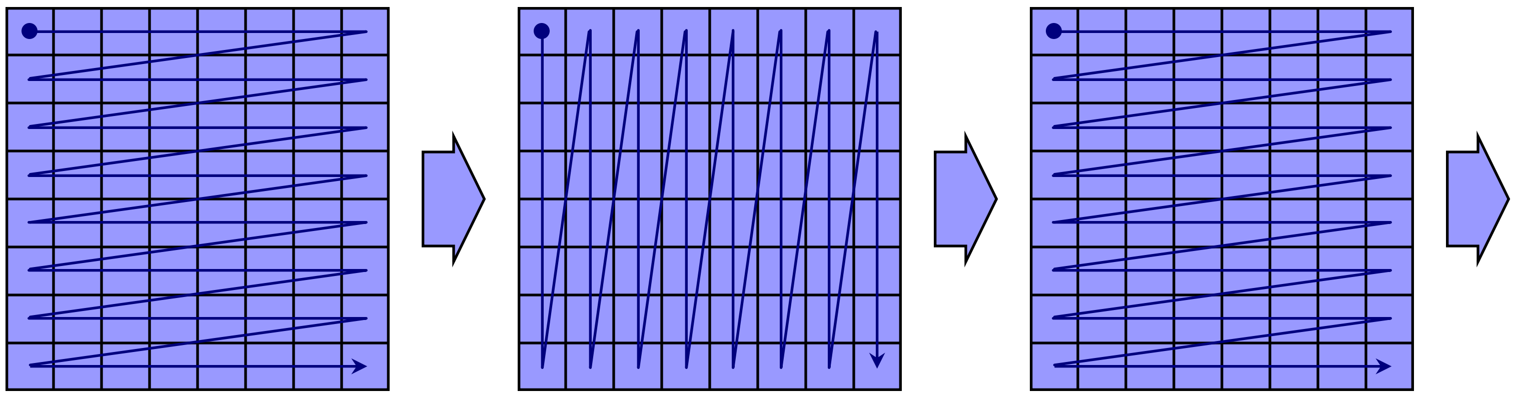
$$\begin{bmatrix} Y(0) \\ Y(2) \\ Y(4) \\ Y(6) \end{bmatrix} = \begin{bmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{bmatrix} \begin{bmatrix} X(0)+X(7) \\ X(1)+X(6) \\ X(2)+X(5) \\ X(3)+X(4) \end{bmatrix}$$

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \\ Y(7) \end{bmatrix} = \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(0)-X(7) \\ X(1)-X(6) \\ X(2)-X(5) \\ X(3)-X(4) \end{bmatrix}$$

TABLE I
INPUT SELECTION FOR ACCUMULATORS FOR THE DCT/IDCT

ACF ACCBANK FOR DCT/IDCT				BDEG ACCBANK FOR DCT/IDCT			
ACC0	ACC1	ACC2	ACC3	ACC0	ACC1	ACC2	ACC3
a/a	c/a	a/a	f/a	g/d	e/g	d/b	b/e
a/c	f/f	a/f	c/c	e/e	b/b	g/g	d/d
a/f	f/c	a/c	c/f	d/b	g/d	b/e	e/g
a/a	c/a	a/a	f/a	b/g	d/e	e/d	g/b

Transpose Memory (1/2)



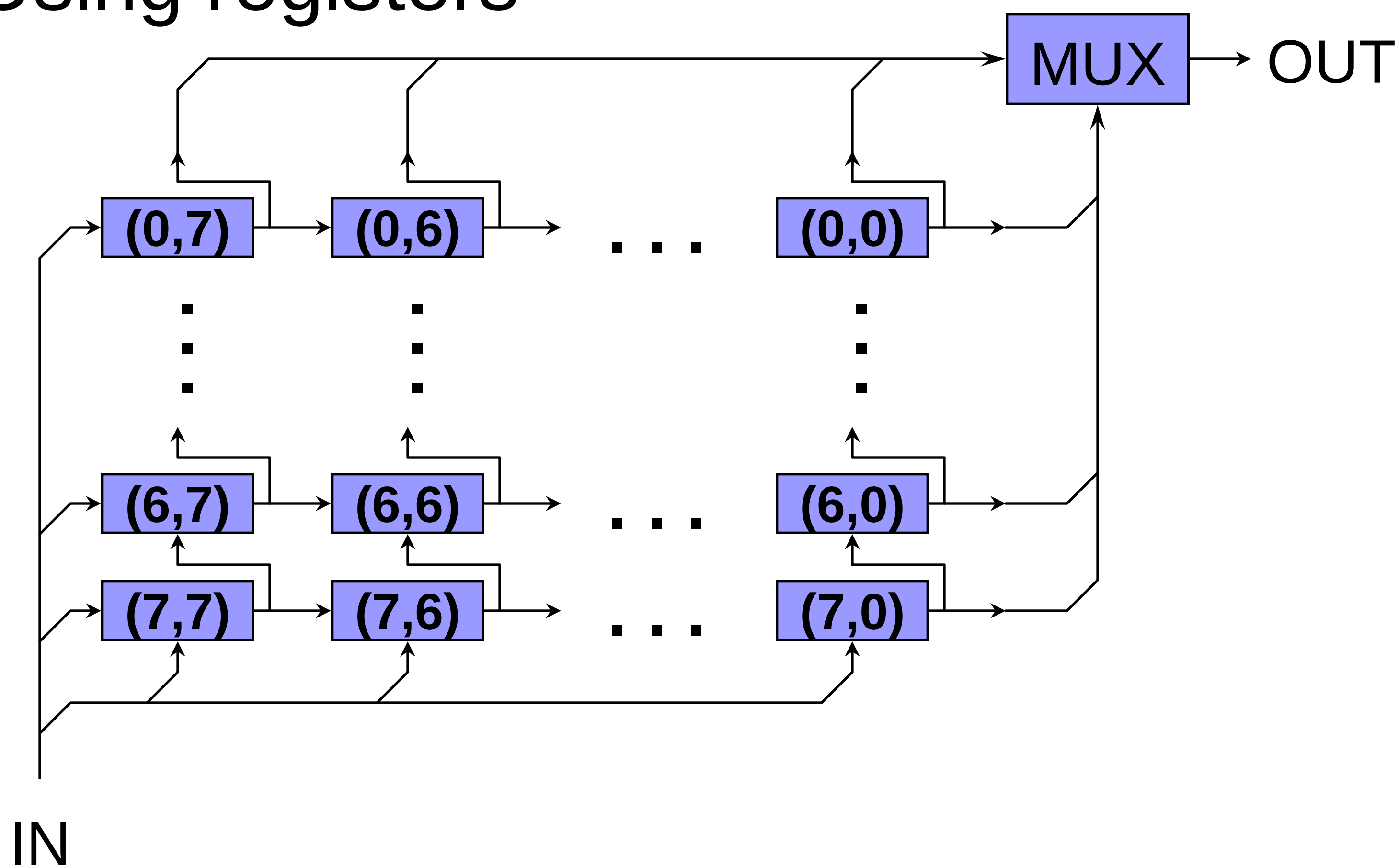
- Pin-pong mode
- Using RAMs

ADDR			
TIME	0 1 2 3... 7	8 9 10 11...15	16 17...62 63
	0 8 16 24...56	1 9 17 25...57	2 10...55 63
	0 1 2 3... 7	8 9 10 11...15	16 17...62 63

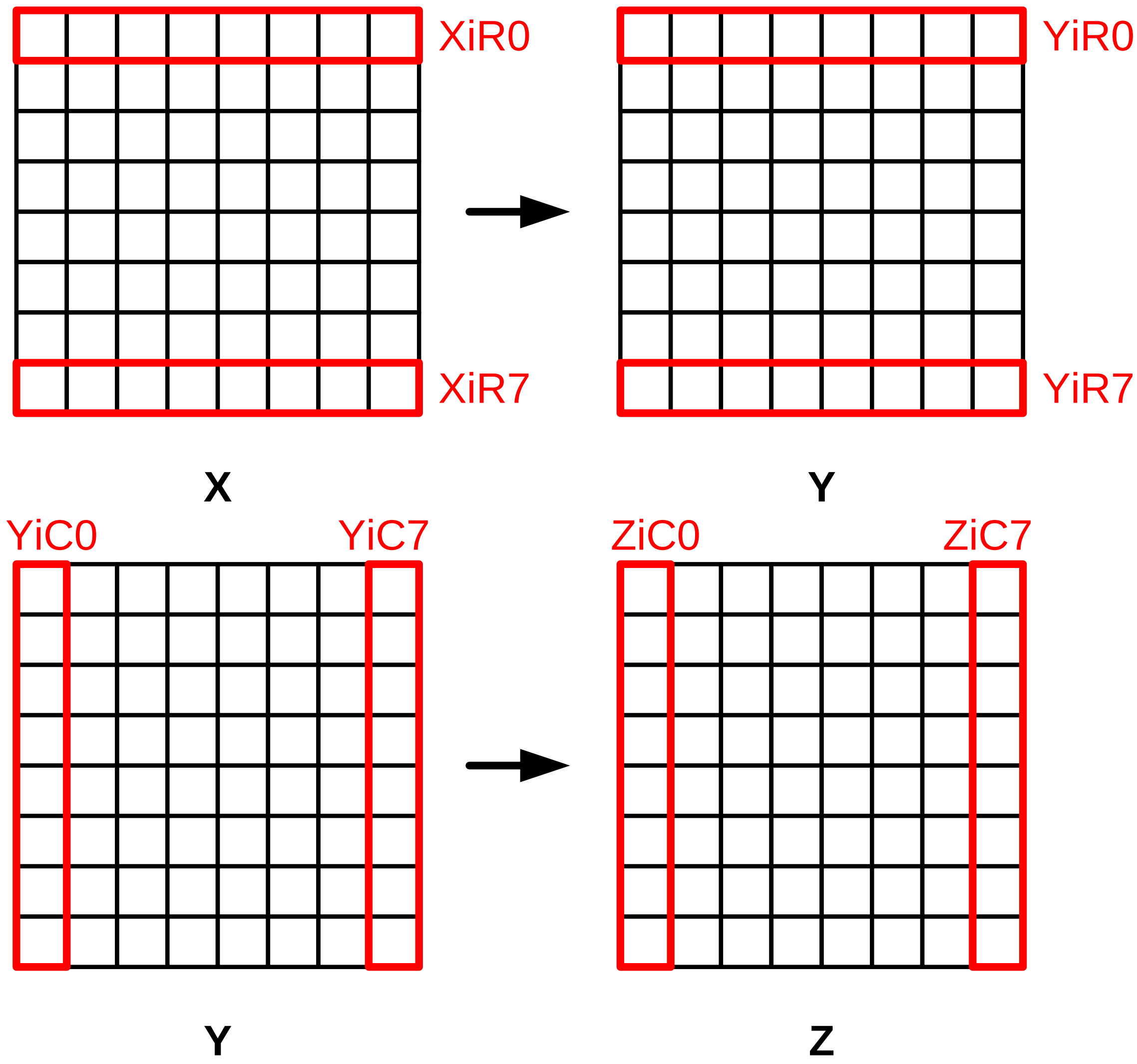
⋮

Transpose Memory (2/2)

- Using registers

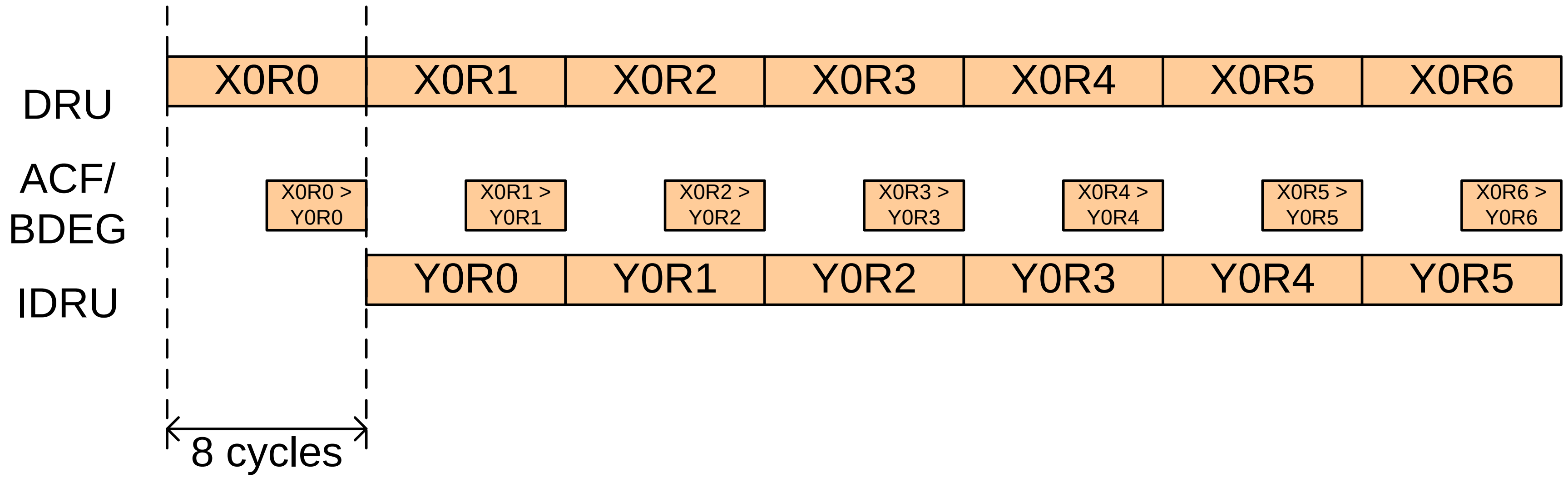


Scheduling

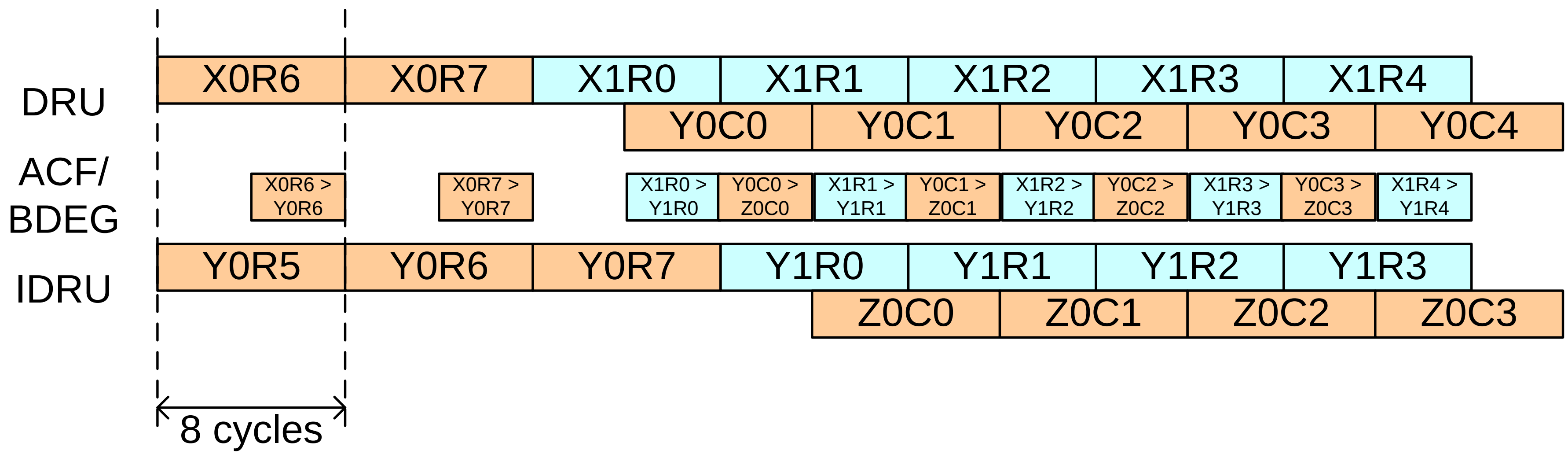


R: row
C: column
i: block index

Scheduling


DRU
ACF/
BDEG
IDRU
X0R0
X0R1
X0R2
X0R3
X0R4
X0R5
X0R6
X0R0 > Y0R0
X0R1 > Y0R1
X0R2 > Y0R2
X0R3 > Y0R3
X0R4 > Y0R4
X0R5 > Y0R5
X0R6 > Y0R6
Y0R0
Y0R1
Y0R2
Y0R3
Y0R4
Y0R5
8 cycles

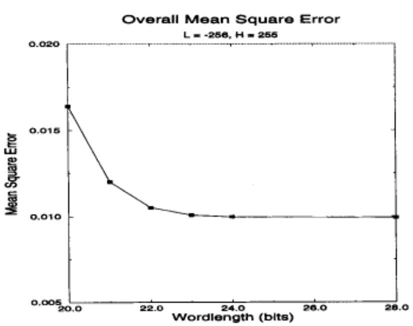
Scheduling

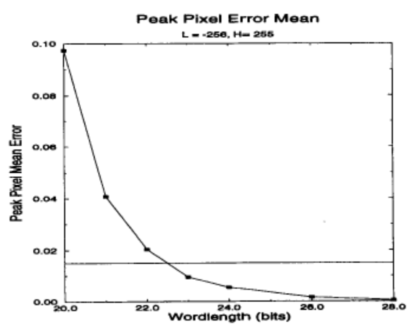


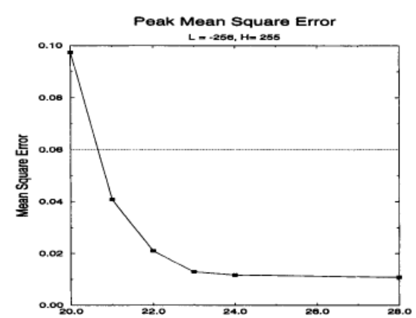
100% hardware utilization!

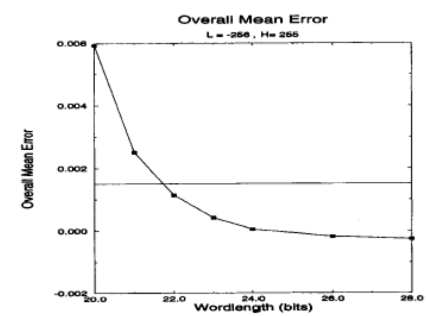
Finite Word Length Analysis

- The word length for the accumulator is 22-bit









	Specification	L,H = -256,255	L = H = 300	L = H = 5
Peak Pixel Error	≤ 1	1	1	1
Peak Pixel Mean Square Error	≤ 0.06	0.0134	0.0153	0.0139
Overall Mean Square Error	≤ 0.02	0.0104	0.0101	0.0028
Peak Pixel Mean Error	≤ 0.015	0.0133	0.0125	0.0139
Overall Mean Error	≤ 0.0015	0.00096	0.0011	0.0011

Implementation Results

Core Characteristic

Inputs	9 bits (DCT), 12 bits (IDCT)
Outputs	12 bits (DCT), 9 bits (IDCT)
Internal Wordlength	22 bits
Technology	0.8-μm CMOS, triple metal
No. of Transistors	67,000
Core Size	10 mm^2
Clock Rate	100 MHz
Mode Selection	DCT or IDCT
Block Size	8×8
Accuracy	CCITT compliant

IO Specification

Function	Number
Input	12
Output	12
Clock	1
Power	8
Ground	8
DCT/IDCT control	1
Start	1

1-D Approach with DA

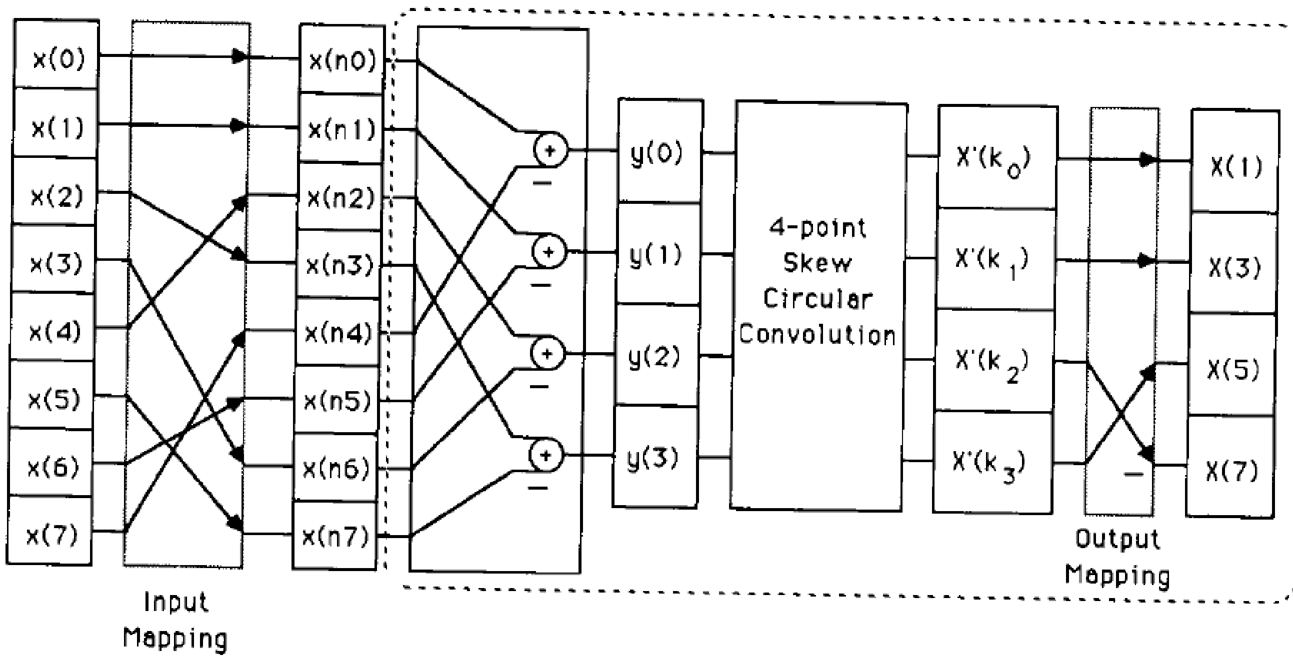
D. Slawecki and W. Li, "DCT/IDCT processor for high data rate image coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 2, June 1992

DCT Algorithm (1/3)

$$\begin{bmatrix} Y(0) \\ Y(2) \\ Y(4) \\ Y(6) \end{bmatrix} = \begin{bmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{bmatrix} \begin{bmatrix} X(0)+X(7) \\ X(1)+X(6) \\ X(2)+X(5) \\ X(3)+X(4) \end{bmatrix}$$

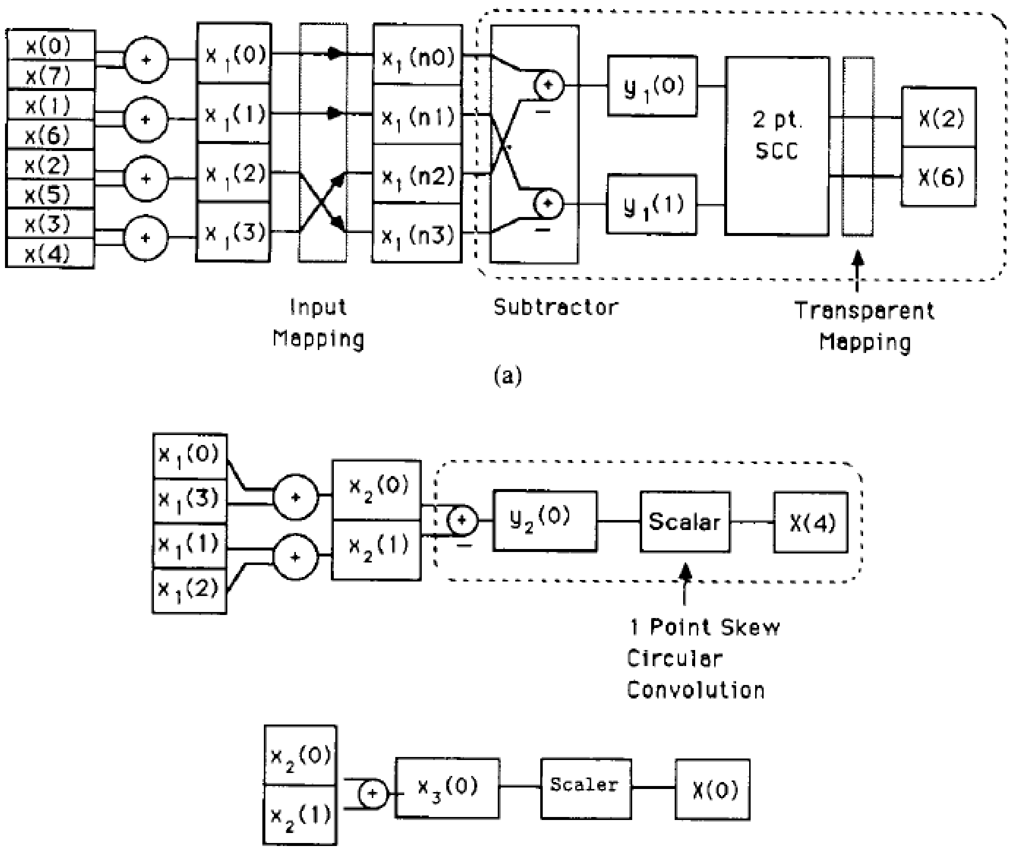
$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \\ Y(7) \end{bmatrix} = \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(0)-X(7) \\ X(1)-X(6) \\ X(2)-X(5) \\ X(3)-X(4) \end{bmatrix}$$

- Odd-indexed DCT component



DCT Algorithm (2/3)

- Even-indexed DCT component



(a)

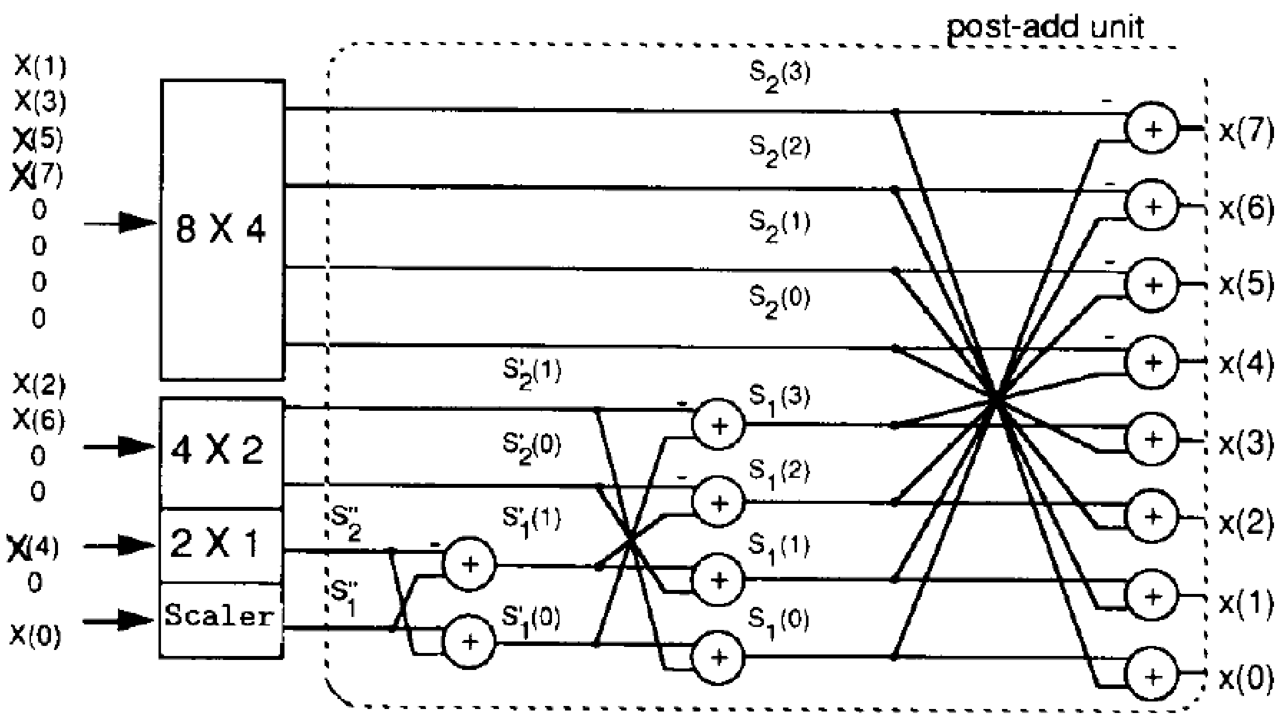
(b)

$$\begin{bmatrix} Y(0) \\ Y(2) \\ Y(4) \\ Y(6) \end{bmatrix} = \begin{bmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{bmatrix} \begin{bmatrix} X(0)+X(7) \\ X(1)+X(6) \\ X(2)+X(5) \\ X(3)+X(4) \end{bmatrix}$$

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \\ Y(7) \end{bmatrix} = \begin{bmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{bmatrix} \begin{bmatrix} X(0)-X(7) \\ X(1)-X(6) \\ X(2)-X(5) \\ X(3)-X(4) \end{bmatrix}$$

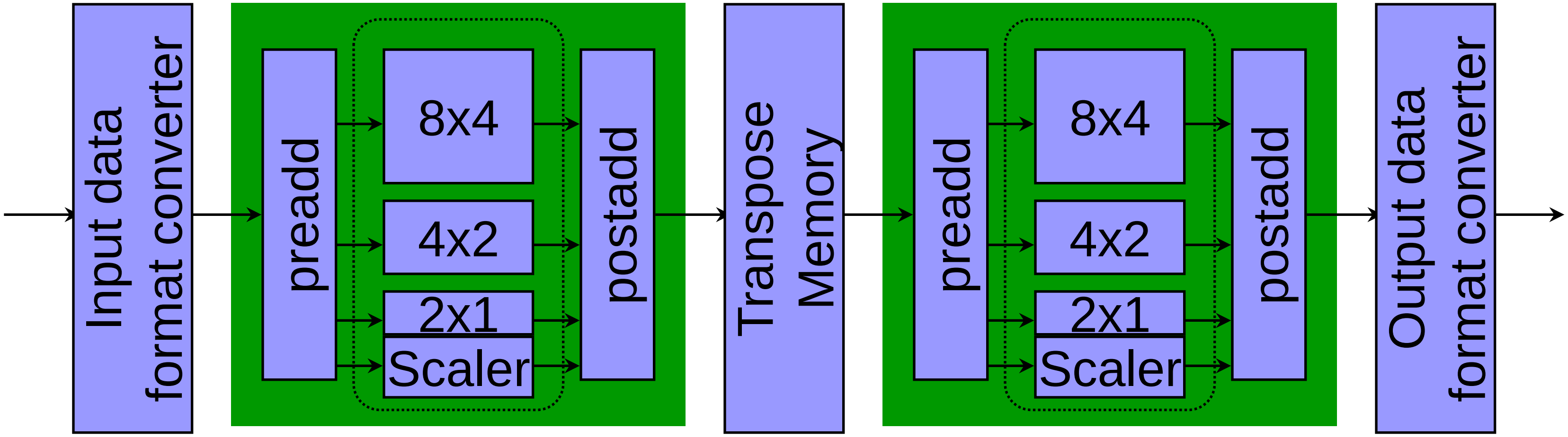
DCT Algorithm (3/3)

- IDCT



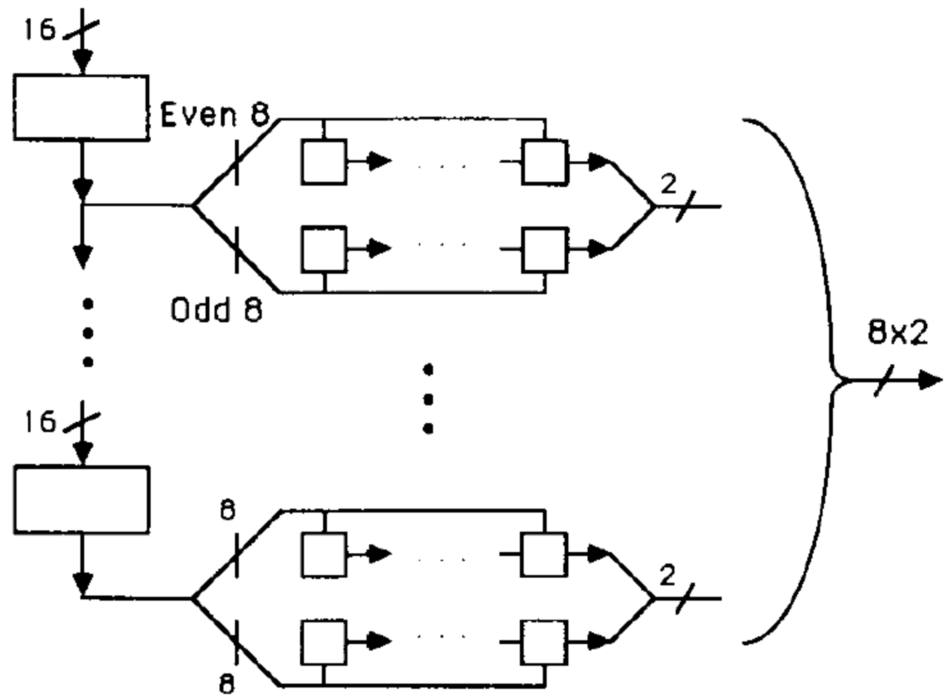
Block Diagram

- Word-serial/bit parallel input/output
- Word-parallel/digit-serial datapath
- Block Diagram

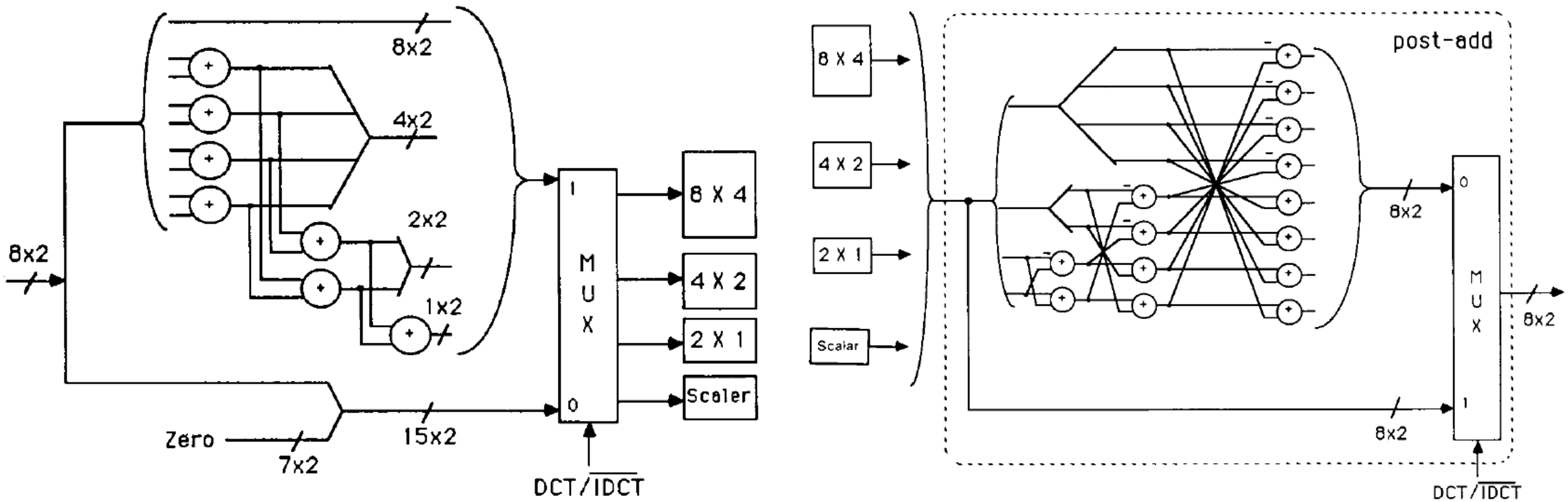


Input Data Format Converter

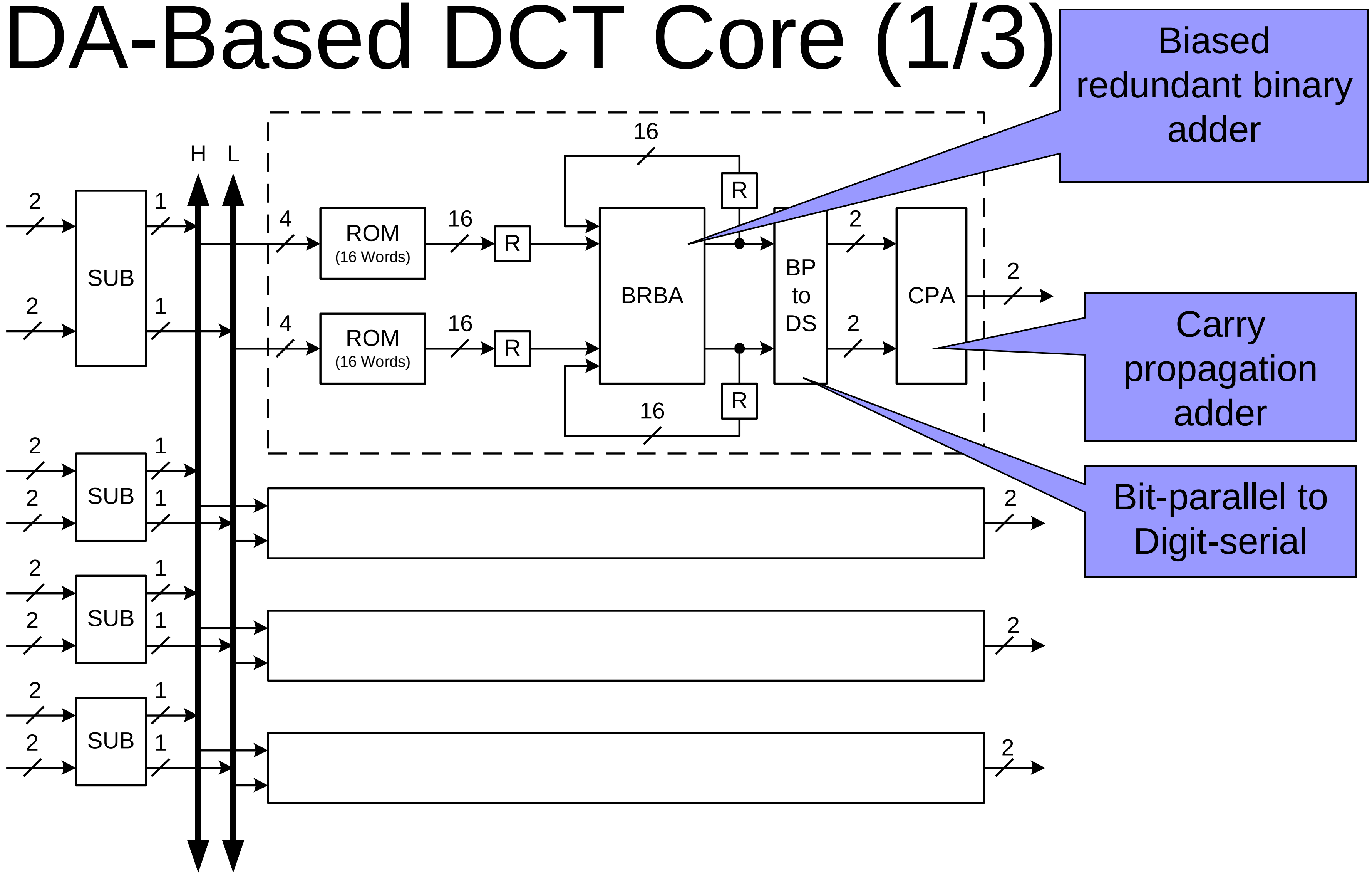
- Input: word-serial/bit-parallel
- Output: word-parallel/2-digit serial



Preadd and Postadd

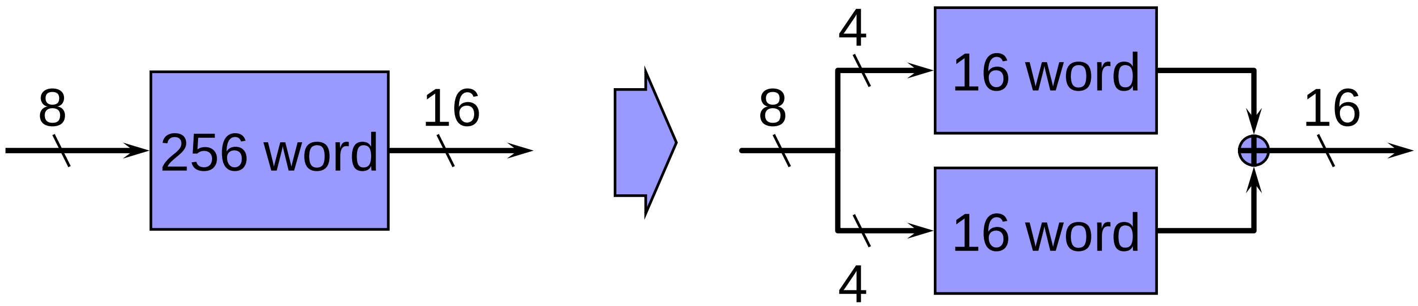


DA-Based DCT Core (1/3)



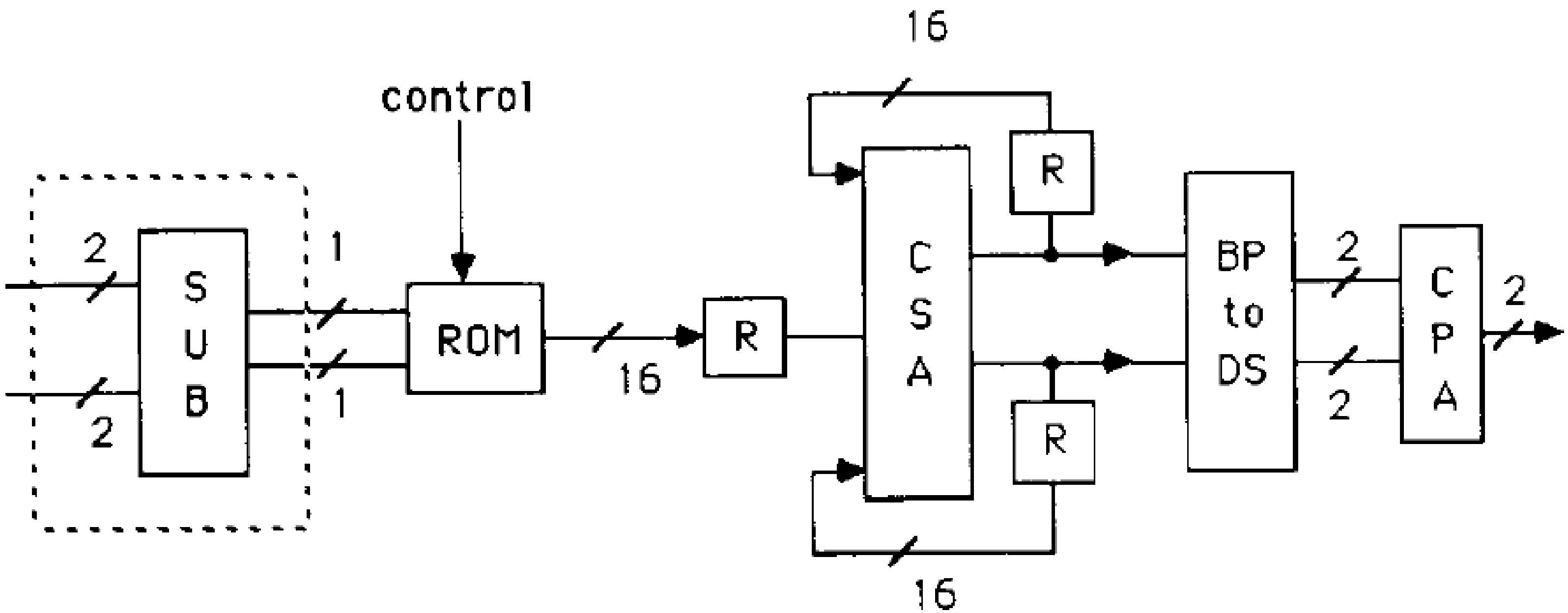
DA-Based DCT Core (2/3)

- ROM size reduction

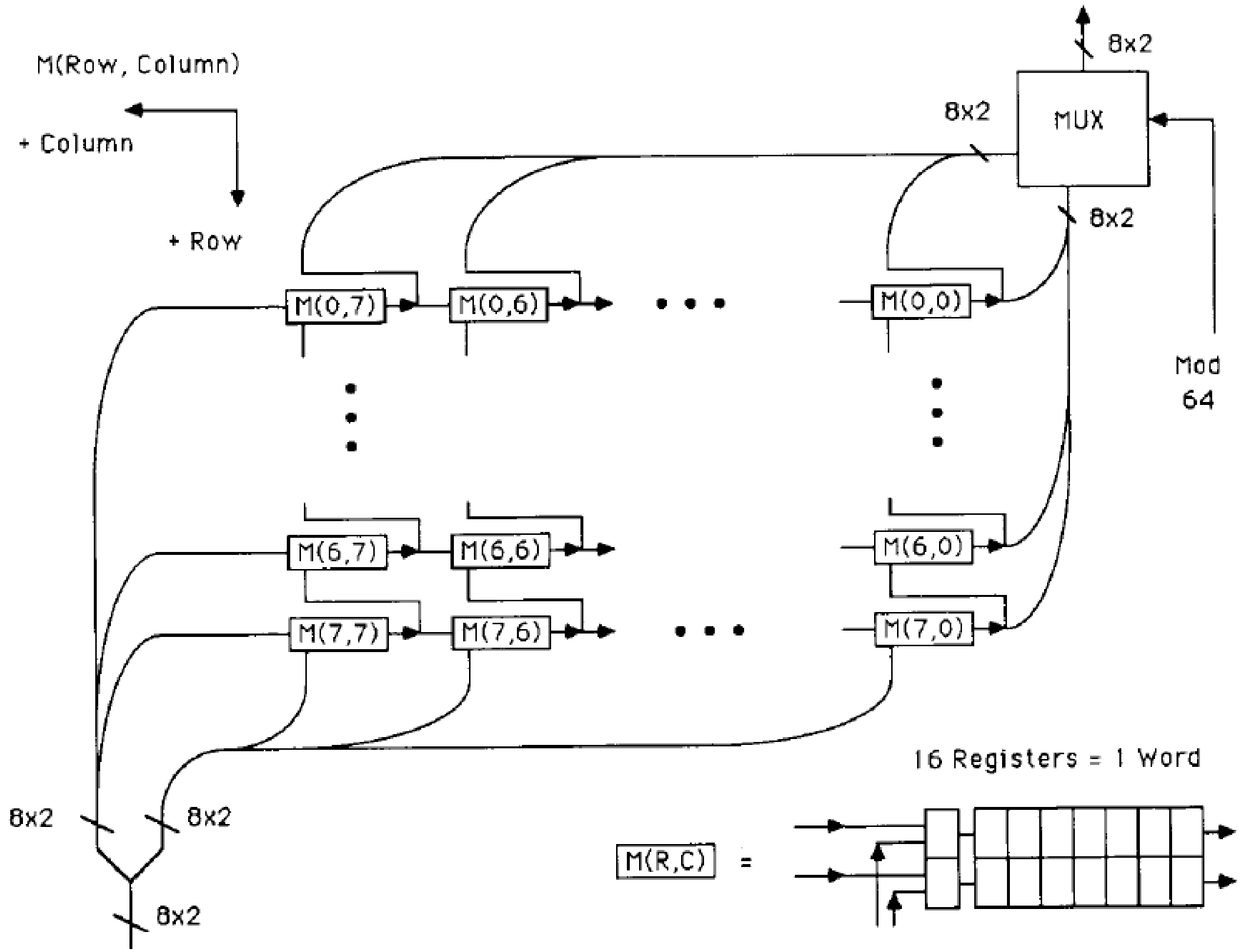


DA-Based DCT Core (3/3)

- 2x1 and scaler units



Transpose Memory



Direct 2-D DCT Architecture

Direct 2-D DCT Method

- Computing the transform directly from the N x N input numbers
  - Derive fast DCT algorithms from the signal flow graph (like FFT)
  - Based on 1-D DCT
  - Larger flow graph
  - Global routing
  - More temporal storage
  - Larger datapath

An Example of 2-D DCT Architecture