Lesson 110: Discrete Cosine Transform

University of Victoria
British Columbia • Canada

Discrete Cosine Transform

- 8-point 1D DCT

$$X(u) = \frac{C(u)}{2} \sum_{i=0}^{7} x(i) \cos \frac{(2i+1)u\pi}{16}$$

where $x(i)$ are the inputs, $X(u)$ are the outputs, and $C(u) = 1/\sqrt{2}$ for $u = 0$, otherwise is 1

- 8×8-point 2D DCT:

$$X(u,v) = \frac{C(u)}{2} \frac{C(v)}{2} \sum_{i=0}^{7} \sum_{j=0}^{7} x(i,j) \cos \frac{(2i+1)u\pi}{16} \cos \frac{(2j+1)v\pi}{16}$$

Discrete Cosine Transform (cont'd)

- Using matrix notation, the 8×8-point 2D DCT can be expressed as a matrix product

$$[X] = [C_{8\times8}] * [x]$$

where $[x]$ and $[X]$ are the two-dimensional input and output sequences, respectively

- A standard strategy to compute the 2D DCT is the row-column separation

$$[X] = [C] * [x] * [C]^t$$

where $[C]$ is the matrix for 1D DCT

Discrete Cosine Transform (cont'd)

- The expanded matrix representation for the 8-point 1D DCT

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \\ X(7) \end{bmatrix} = \begin{bmatrix} C4 & C4 & C4 & C4 & C4 & C4 & C4 & C4 \\ C1 & C3 & C5 & C7 & -C7 & -C5 & -C3 & -C1 \\ C2 & C6 & -C6 & -C2 & -C2 & -C6 & C6 & C2 \\ C3 & -C7 & -C1 & -C5 & C5 & C1 & C7 & -C3 \\ C4 & -C4 & -C4 & C4 & C4 & -C4 & -C4 & C4 \\ C5 & -C1 & C7 & C3 & -C3 & -C7 & C1 & -C5 \\ C6 & -C2 & C2 & -C6 & -C6 & C2 & -C2 & C6 \\ C7 & -C5 & C3 & -C1 & C1 & -C3 & C5 & -C7 \end{bmatrix} * \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ x(5) \\ x(6) \\ x(7) \end{bmatrix}$$

where $Cn = \cos n\pi/16$

- A direct implementation of would require 64 multiplications and 56 additions

University of Victoria
British Columbia • Canada

Discrete Cosine Transform (cont'd)

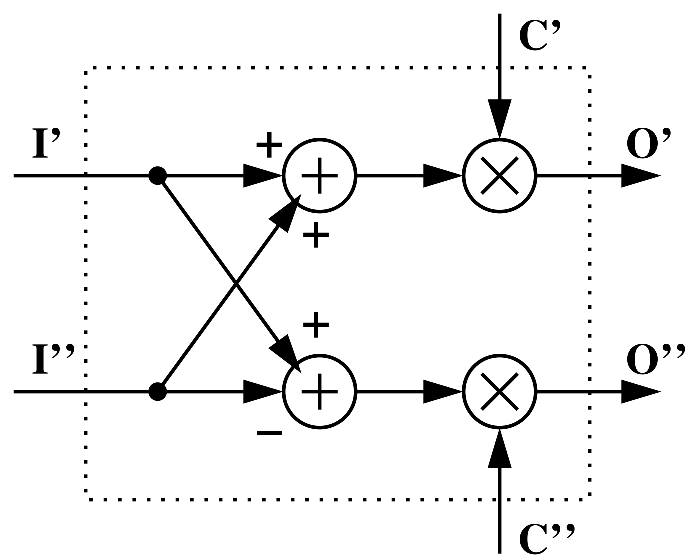
- By exploiting the symmetry in the DCT coefficient matrix the number of multiplications is reduced to 32, at the expense of additional 8 additions/subtractions

$$\begin{bmatrix} X(0) \\ X(2) \\ X(4) \\ X(6) \\ X(1) \\ X(3) \\ X(5) \\ X(7) \end{bmatrix} = \begin{bmatrix} C4 & C4 & C4 & C4 & 0 & 0 & 0 & 0 \\ C2 & C6 & -C6 & -C2 & 0 & 0 & 0 & 0 \\ C4 & -C4 & -C4 & C4 & 0 & 0 & 0 & 0 \\ C6 & -C2 & C2 & -C6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & C1 & C3 & C5 & C7 \\ 0 & 0 & 0 & 0 & C3 & -C7 & -C1 & -C5 \\ 0 & 0 & 0 & 0 & C5 & -C1 & C7 & C3 \\ 0 & 0 & 0 & 0 & C7 & -C5 & C3 & -C1 \end{bmatrix} * \begin{bmatrix} x(0)+x(7) \\ x(1)+x(6) \\ x(2)+x(5) \\ x(3)+x(4) \\ x(0)-x(7) \\ x(1)-x(6) \\ x(2)-x(5) \\ x(3)-x(4) \end{bmatrix}$$

Discrete Cosine Transform (cont'd)

- **Butterfly** computation is widely encountered:

$$\begin{aligned} O' &= (I' + I'') \cdot C' \\ O'' &= (I' - I'') \cdot C'' \end{aligned} \tag{1}$$



University of Victoria
British Columbia • Canada

Discrete Cosine Transform (cont'd)

- Butterfly – vector-valued function
  - Any restrictions due to ARM architecture?
- Additions and subtractions – that's easy
- Multiplication by constant (seven constants)
  - Software: use MUL operation
  - Hardware: one/two full multipliers or seven multipliers-by-constant?
- How many new instructions?
  - What syntax?
    DCT Rs1, Rs2, Rt
    where Rs1 = [I', C'], Rs2 = [I'',C''], Rt = [O1, O'']
  - What latency?
  - How to rewrite the code using these new instructions?

DCT – project requirements

- Build the testbench: the input is a 8-by-8 matrix of integers
- Write a DCT in software and highlight the Butterfly routine
- Implement Butterfly in:
  - software (write C routines)
  - horizontal firmware with two issue slots
  - custom hardware (write VHDL/Verilog)
- Define a new instruction (or new instructions) that does Butterfly
  - You must comply with the ARM architecture (you can have at most two arguments and one result per instruction call)

DCT – project requirements

- Rewrite the high-level code and instantiate the new instruction
  - Use assembly inlining
- Estimate:
  - the performance improvement of hardware-based solution versus software-based solution
  - the performance improvement of a 2-issue slot firmware-based solution versus software-based solution
- Estimate the penalty in terms of number of gates for the hardware solution

Questions, feedbacks

?