RTL RUMBLE

problem statement

BYTE
DEPARTMENT OF EEE

DCT (Discrete Cosine Transform) is a popular tool used in image (JPEG), video (H.264) and audio (MP3) compression pipelines like JPEG, H.264, etc. The complexity and nature of calculations in the algorithm has lead to various works in finding efficient and optimized Hardware implementation of the algorithm.

In this challenge, you have been tasked with designing an optimized hardware implementation of 16-Point 1-D Fast Discrete Cosine Transform in 48 hours.

MATHEMATICAL BACKGROUND

Given 16 real input samples:

$$x[0], x[1], \ldots, x[15]$$

The 16-point DCT produces 16 output coefficients:

$$X[0], X[1], \ldots, X[15]$$

Each output coefficient represents the correlation of the input sequence with a cosine basis function of a specific frequency.
The mathematical definition of the DCT-II (used in this challenge) is:

$$X[k] = \sum_{n=0}^{15} x[n] \cdot \cos\left(\frac{\pi}{16}(n + 0.5)k\right)$$

$$:k = 0, 1, 2 \ldots 15$$

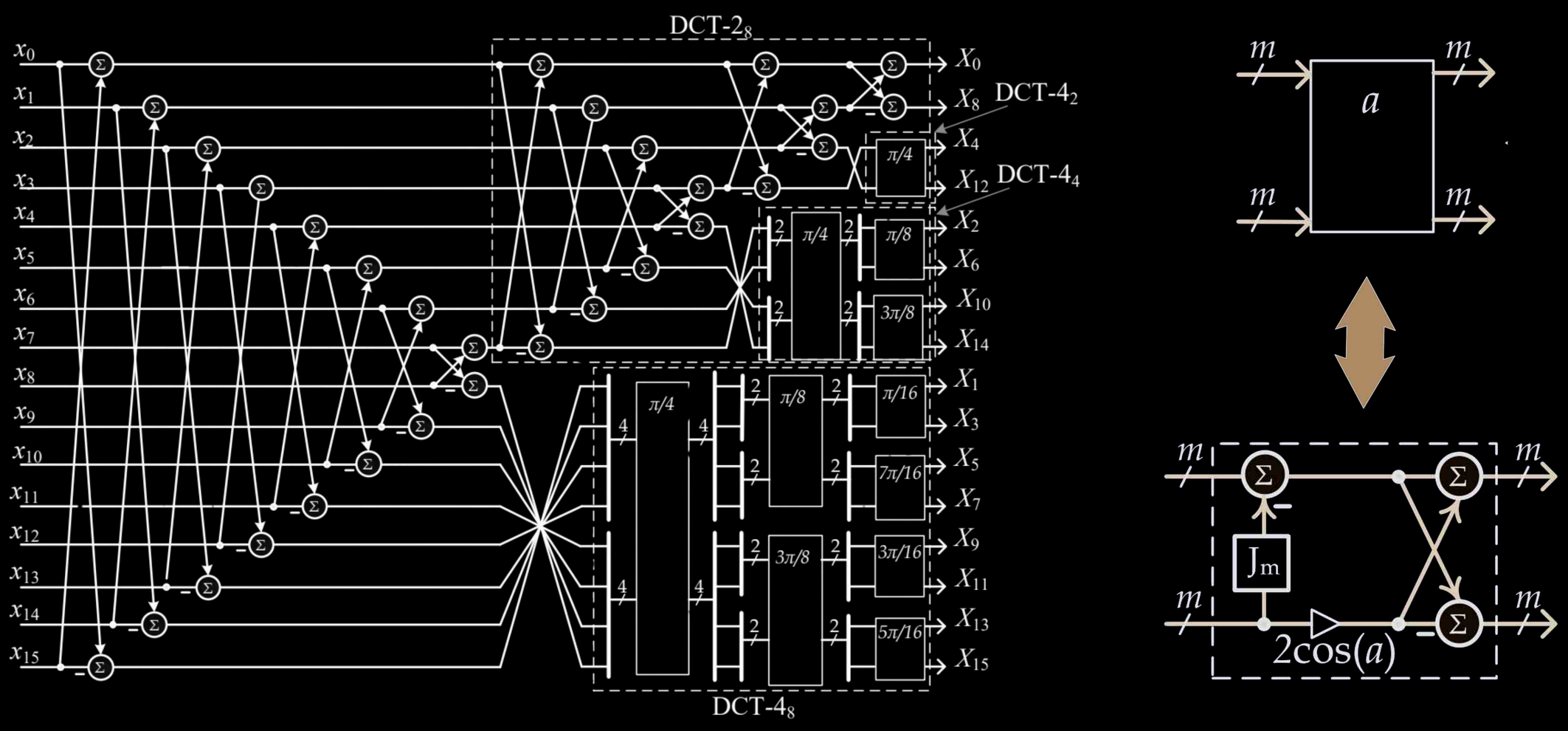
This calculation can be cumbersome and Hardware inefficient due to this hardware implementations often use decimation schemes to calculate DCT coefficients efficiently.
One such implementation is described in the research paper provided with this document.

RESOURCES PROVIDED

- Research Paper Describing The Hardware Implementation Of A 16-Point DCT
- Sample Python Code For Output Verification

HARDWARE IMPLEMENTATION

Fig. 1 & 2 of the research paper describe the hardware implementation of a 16 point fast-DCT scheme. Fig 1. describes the overall architecture and Fig 2. describes the structure of the cosine building blocks used in the decimation scheme. Your task is to replicate this architecture to compute DCT coefficients.



Note: The output of the architecture shown in Fig. 1 outputs a 'scaled' DCT transform. Computing coefficients from $X[k] = \sum_{n=0}^{15} x[n] \cdot \cos\left(\frac{\pi}{16}(n + 0.5)k\right)$ may not match the system's output. Use the python script provided to compare outputs.

RTL RUMBLE

problem statement

BYTE
DEPARTMENT OF EEE

HARDWARE SPECIFICATION AND CONSTRAINTS

The module shall have a **reset** button.

THE INPUT CYCLE

The module shall have a 1-bit **start** input.

when the start input is signaled HIGH, The module shall start loading 2 unsigned (assume positive) 8 bit inputs. (First input comes with the posedge of the start signal)

The inputs will be given in the following order:

clock 1 (start HIGH received):	**input 1: x[0]**	**input 2: x[15]**
clock 2:	**input 1: x[1]**	**input 2: x[14]**
clock 3:	**input 1: x[2]**	**input 2: x[13]**

And so on, for 8 clock cycles

THE OUTPUT CYCLE

The module shall have a 1-bit **output_en** signal which remains HIGH when the module outputs the final coefficients.

The Outputs shall be in Q.8 form ie the outputs shall be in fixed-point form with a decimal accuracy of 8 bits.

The module shall output 2 signed coefficients in each clock cycle, **OUTPUT_A & OUTPUT_B** whilst outputting final coefficients.

SUGGESTION: Determine the bit-size of the OUTPUT_A & OUTPUT_B before starting implementation

The order ie whether X[0], X[8] are outputted first or X[1], X[3] are outputted first can be changed according to participant's architecture.

The module shall have 2 4-bit outputs: **INDEX_A & INDEX_B** representing the index of the coefficient which has been outputted ie

if **OUTPUT_A = X[0], OUTPUT_A = X[8]** are outputted, **INDEX_A = 0, INDEX_B = 8**

MODULE HEADER FOR THE TOP MODULE

```
module DCT(
    input clk,
    input reset,
    input start,
    input [7:0] INPUT_A,
    input [7:0] INPUT_B,
    output signed [?:0] OUTPUT_A,
    output signed [?:0] OUTPUT_B,
    output [3:0] INDEX_A,
    output [3:0] INDEX_B,
    output output_en
);
```

DELIVERABLES

- The RTL code for the design.
- A testbench with the input array, [1, 3, 5, 7, 9, 17, 19, 21, 22, 18, 18, 16, 8, 6, 4, 2]
- A Document containing the following information about the designed architecture
  - A brief summary of the design, how many adders, multipliers used, the size of the output determined, pipelined stages/FSMs used, etc
  - The Utilization Report, both synthesis and implementation
  - The timing summary with clock set with a period of 10ns - implementation timing summary
  - Screenshot of the elaborated design of the top module, and the individual modules utilized
  - Screenshot of the testbench run
  - Report the number of clock cycles taken to generate final output (include input and output clock cycles)

**We recommend using Xilinx-Vivado, but in case the tool is unavailable, We will check utilization and timing of the designed architecture on our systems.

**The synthesis and all the reports have to be generated with ZedBoard Zynq-7000 evaluation board selected as the Board

RTL RUMBLE

problem statement

BYTE
DEPARTMENT OF EEE

EVALUATION CRITERIA

- The Architecture of the Building Block and it's functional correctness.
- Overall Architecture's functional correctness, Outputs shall match with the provided Python Script.
- Implementation of the Hardware specification and constraints specified earlier, especially meeting the accuracy requirement. (8 decimal bits)
- Maximizing clock frequency, Having pipeline stages.
- Achieving balance between clock cycles taken and hardware utilized.

SOME CLARIFICATIONS

- This is a 1D DCT only (not 2D).
- No knowledge of image compression is required.
- The focus is on hardware realization, not theoretical proof.

SUBMISSION GUIDELINES

- Submission links will be sent to team leads via emails and Whatsapp groups.
- Submission Deadline: Saturday, 21st February 11:59PM
- All the documents, codes, etc shall be zipped in a folder with your team name as the title
- We encourage any type of submission, even if incomplete
- If your synthesis reports are widely inconsistent with the synthesis of your design done on our systems, your team may be disqualified
- If your team is unable to provide proper explanation of the RTL Design created by you, team will face a penalty