International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 17 (2015) pp 37311-37315
© Research India Publications. http://www.ripublication.com

Efficient 2D-DCT Architecture for Video Applications

R.Lalitha
Student, Department of ECE, MVGR College of Engineering, Vizianagaram, A.P.
Lalitha.raparthi235@gmail.com

Mr.P.Suryaprasad
Associate Professor, Department of ECE, MVGR College of Engineering, Vizianagaram, A.P.
suryaprasadp@yahoo.com

Abstract
This paper presents an efficient area and delay architectures for the implementation of one Dimensional and two Dimensional Discrete Cosine Transform (DCT). These are supported to different lengths (4, 8, 16, and 32). DCT blocks are used in the different video coding standards for the image compression. The 2D- DCT calculation is made using the 2D-DCT Separability Property, such that the whole architecture is divided into two 1D-DCT calculations by using a transpose buffer. Based on the existing 1D-DCT architecture two different types of 2D-DCT architectures, folded and parallel types are implemented. Both of these two structures use the same Transpose buffer. Proposed transpose buffer Occupies less area and high speed than existing transpose buffer. Hence the area, low power and delay of both the 2D-DCT architectures are reduced.

Keywords: DCT, HEVC, Transpose buffer, video compression.

INTRODUCTION
Image compressing plays an important role in digital image compression and also important for efficient video Transmission and storage of images. The Discrete Cosine Transform plays a main role in image or Video Compression due to its de-correlation efficiency. There are different video code standards to video compression such as JPEG, MPEG, and AVC. The new H.265/High Efficiency Video Coding (HEVC) standard [2,3] has been recently finalized to replace H.264/AVC [7].The advantage of HEVC is to reduce the half bit rate with same video quality compared than previous video code standard i.e advanced video code standard (H.264). The advantage of HEVC is that it supports DCT of different sizes such as 4, 8, 16, and 32 [9]. The proposed DCT architectures are supported to HEVC. Discrete Cosine Transform divides the image into different parts (or spectral sub bands). DCT is mainly used in the lossey image compression.

EXISTING ARCHITECTURES FOR INTEGER DCT COMPUTATION
In this paper the one dimensional (general, reusable of four point, eight point, sixteen point, thirty-two point), and two dimensional DCT [10] architectures are implemented (4point and 8point) and mainly concentration is given on area and delay efficiency of two dimensional DCT architectures.

A.1D DCT architectures
Discrete Cosine Transform architecture [8] consists of mainly three blocks. They are input adder unit (IAU), shift adder unit (SAU), output adder unit (OAU). IAU consist the butterfly structure [7]. Inputs (x0, x1, x2…..xn) are applied to input adder unit by using below equation:

$$a(i) = x(i) + x(N-i-1)$$
$$b(i) = x(i) - x(N-i-1)$$

Fig.1. shows the proposed architecture of four-point integer DCT.

i) General Architecture for DCT
General architecture for DCT is shown in Fig. 2. The N point DCT Architecture [4] consists of IAU, N/2 point DCT, N/2 SAU blocks and OAU block. Inputs (x0,x1……x(n-1)) are applied to IAU, it generates the (a0,a1…..a(N/2-1) and b0,b1…b(N/2-1) . (a0, a1….a (N/2-1)) applied to N/2 point DCT block. This block produces the even terms of the output. SAU and OAU blocks compute the odd terms of the output. For example in 16 point DCT, 8 point DCT calculates the (y0,y2,y4,y6,y8,y10,y12,y14) and lower SAU and OAU calculates the (y1,y3,y5,y7,y9,y11,y13). Again 8 point DCT contains the 4point DCT and 4 SAU and OAU blocks. These calculate the (y0, y2, y4, y6, y8, y10, y12, y14). But this is not reusable so implemented the reusable architecture of DCT. It supports any lengths with the same throughput of processing irrespective of Transform size.

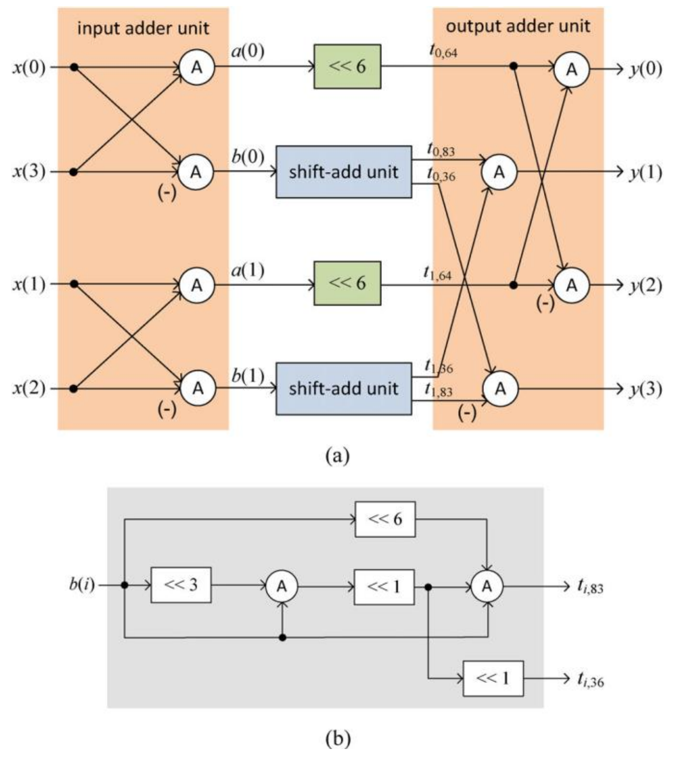


Fig. 1. Architecture of four-point integer DCT. (a) Four-point DCT architecture. (b) Structure of SAU

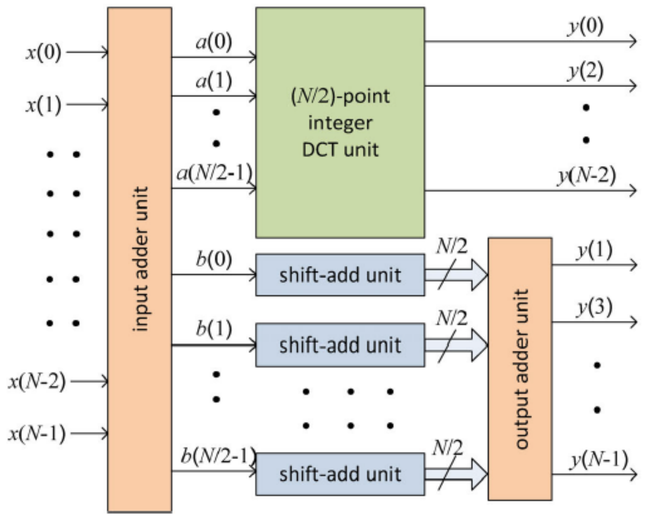


Fig. 2. Generalized architecture for integer DCT of lengths N = 816, and 32,

ii). Reusable Architecture for DCT

The Reusable Architecture of DCT consists of N/2 point upper and lower DCT and control the input to upper (N/2)-point DCT unit is fed through (N/2) 2:1 MUXes that selects either [a (0), ..., a (N/2 − 1)] or [x(0), ..., x(N − 1)], depending on whether it is used for N-point DCT computation or for the DCT of a lower size. The lower (N/2)-point DCT unit takes the inputs [x(N/2), ..., x(N − 1)] when it is used for the computation of DCT of N/2 point or a lower size, otherwise, the input is reset by (N/2) AND gates to reset this (N/2)-point DCT unit. The output of this (N/2)-point DCT unit is multiplexed with that of the OAU. The N AND gates in front of Input adder unit are used to disable the IAU, SAU, and OAU when the architecture is used to compute (N/2)-point DCT computation or a lower size. The input of the control unit, m_N is used to decide the size of DCT computation.

For example, if N = 32, m32 is a 2-bit signal that is set to {00}, {01}, {10}, and {11} to compute four-, eight-, 16-, and 32-point DCT, respectively. The control unit generates sel-1 and sel-2, where sel-1 is used as control signals of N MUXes and input of N AND gates before IAU. Sel-2 is used as the input m(N/2) to two lower size reusable integer DCT units in a recursive manner. The control units for N = 16 and 32, are shown above in Fig. For N = 8, m_8 is a 1-bit signal that is used as sel-1 while sel-2 is not required since four point DCT is the smallest DCT. This structure can compute one 32-point DCT, two 16-point DCTs, four eight point DCTs, and eight four-point DCTs, with same throughput.

B. 2D-DCT Architectures

By using the row-column decomposition technique, an N point 2D-DCT could be computed in two stages [6]. In first stage 1D-DCT is calculated the input matrix of each column and the result is formed as intermediate output matrix. In second stage 1D-DCTis calculated for every row of the intermediate output matrix.

Two different types of architectures are used to compute the 2D-DCT. They are folded architecture and full parallel architecture. These consist of transposition buffer to relate the two 1D-DCT's.

i)Folded architecture

Folded structure is shown in Fig .4. This architecture contains one 1D - DCT unit and a transposition buffer. For the calculation of first stage, the 1D-DCT unit takes the columns of input matrix and result is stored in transposition buffer.

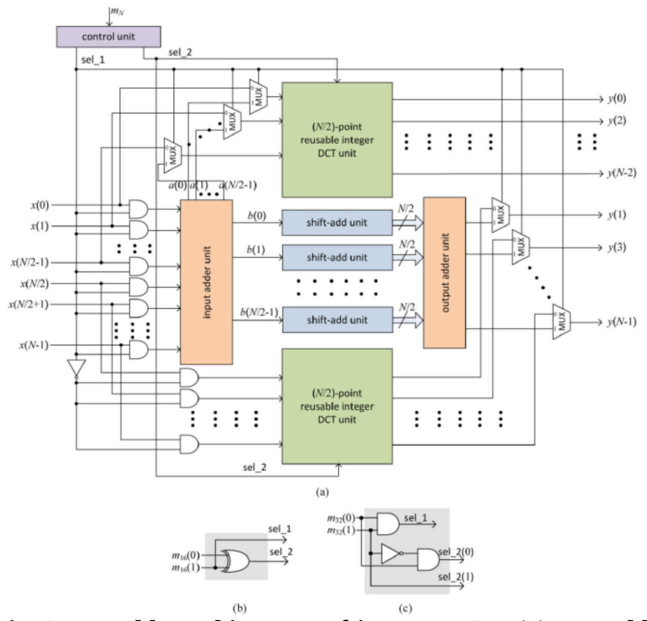


Fig. 3 Reusable architecture of integer DCT. (a) Reusable architecture for N = 8, 16, and 32. (b) Control unit for N = 16. (c) Control unit for N = 32.

second stage, content of the transposition buffer is selected by using the multiplexers and fed as input to the 1D-DCT. These multiplexers select either rows or columns from the buffer.

ii) Full parallel architecture

Full parallel architecture is shown in Fig. 5. This Architecture contains the two 1D-DCT units and transposition buffer. The first 1D-DCT unit takes the input as column wise or row wise. It calculates the intermediate output columns and stored in buffer. Second 1D-DCT unit takes the input from the buffer, calculates rows for 2D-DCT output. After DCT perform the inverse DCT operation in video encoder [5].

C) Transposition Buffer

Transposition buffer structure is shown in Fig. 6. Folded and parallel 2D-DCT Architectures used the same buffer structure and buffer is shown below. This buffer contains the AND gates, registers, Multiplexers. Registers are arranged in a row and column wise. For example, 4 X 4 buffer consists of 16 register cells, 4 Multiplexers, 4 AND gates, 8 X 8 buffer consists of 64 register cells and 8 multiplexers, 8 AND gates, 16 X 16 transpose buffer have 256 register cells and 16 multiplexers. In the same way 32 X 32 buffer have 32 multiplexers and 704 register cells. This buffer stores the N values in register of any column by enabling the column registers using one of the enable signal. AND gates are used to reset the Register cells by using clock and enable signal. Multiplexers are used to select the content of one of the row of registers.

MODIFIED TRANSPOSITION BUFFER

The modified buffer is as shown in Fig. 7. This buffer consists of counter, multiplexers and registers or register cells. In this transposition buffer AND gates are removed hence area and delay are reduced of the buffer compared than traditional transposition buffer. In the existing transpose buffer structure, Enable signal is used to activate all register cells hence delay is high, but in the modified transposition buffer structure, a counter is used. This counter gives the clock pulses to all cells so delay is reduced. In the modify buffer structure, up counter is used, by this the register cells are loaded with all values automatically with no delay. The counter is powered by a clock for every one tick of clock and the registers values are shifted. The Multiplexer selects the input lines from the register cell rows.

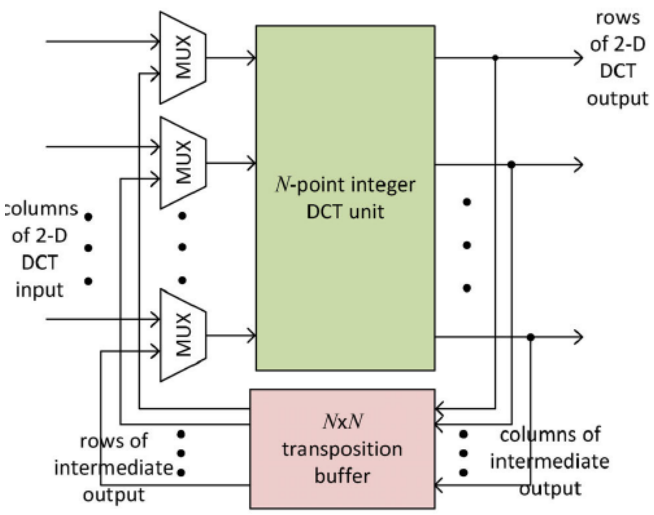


Fig.4. Folded architecture.

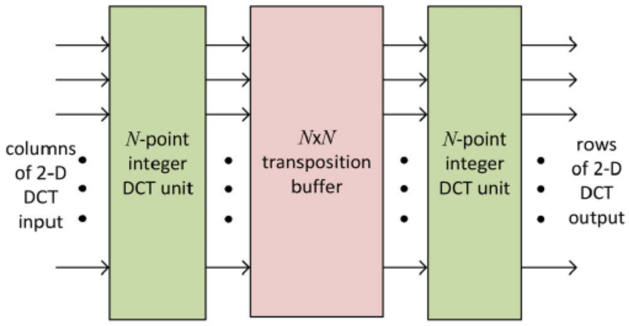


Fig.5 Parallel architecture.

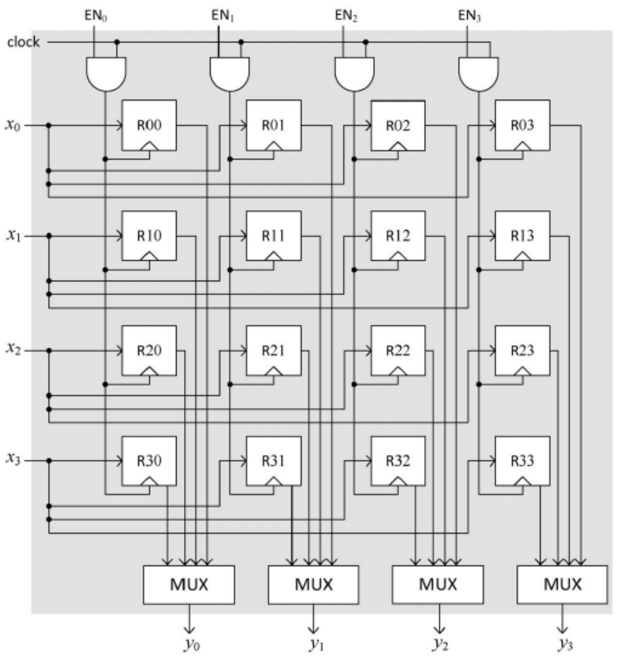


Fig.6. Transpose buffer architecture.

RESULTS

The 1D-DCT and 2D-DCT architectures are coded in Verilog and synthesized by using Xilinx ISE14.2 Synthesis tool and ISIM simulator. The performance of these architectures are compared in terms of occupy slices; look up tables, number of registers and delay and computation time. The Fig. 8 shows the simulation waveforms of 2D DCT architectures. The synthesis report for existing transpose buffer shown in Fig .9 and proposed buffer synthesis report shown in Fig. 10. In the synthesis report, the number of slices, flip-flops. Look up tables, and input-output pins are estimated. Table I shows the result of 1D-DCT Architecture. Table II shows the comparison results of Transpose buffer. Table III shows the results of 4 point 2D-DCT and Table IV shows the results of 8 point 2D-DCT Architecture

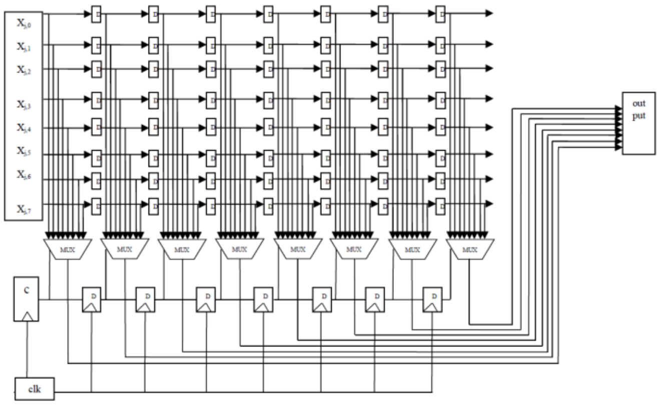
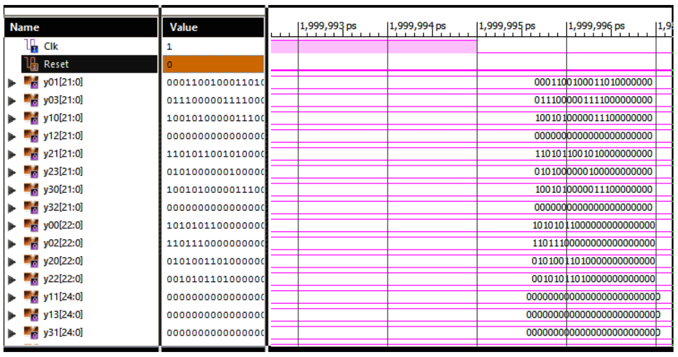
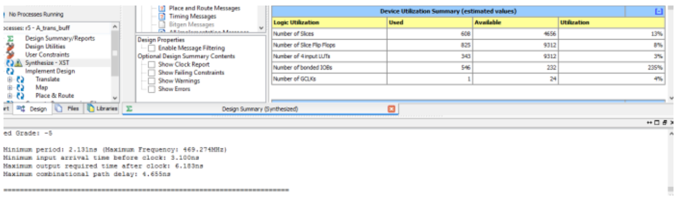


Fig. 7: proposed transposition buffer.



Name	Value	1,999,993 ps … 1,999,996 ps
Clk	1	
Reset	0	
y01[21:0]	0001100100011010	0001100100011010000000
y03[21:0]	0111000001111000	0111000001111000000000
y10[21:0]	1001010000011100	1001010000011100000000
y12[21:0]	0000000000000000	0000000000000000000000
y21[21:0]	1101011001010000	1101011001010000000000
y23[21:0]	0101000000100000	0101000000100000000000
y30[21:0]	1001010000011100	1001010000011100000000
y32[21:0]	0000000000000000	0000000000000000000000
y00[22:0]	1010101100000000	10101011000000000000000
y02[22:0]	1101110000000000	11011100000000000000000
y20[22:0]	0101001101000000	01010011010000000000000
y22[22:0]	0010101101000000	00101011010000000000000
y11[24:0]	0000000000000000	0000000000000000000000000
y13[24:0]	0000000000000000	0000000000000000000000000
y31[24:0]	0000000000000000	0000000000000000000000000

Fig. 8: Simulation waveforms for folded and parallel structure



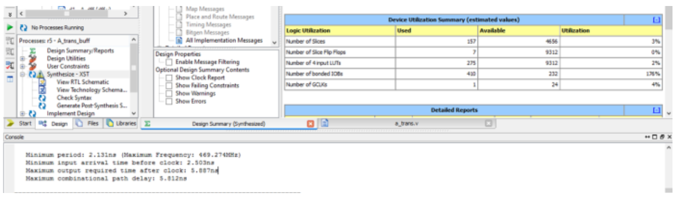
Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	608	4656	13%
Number of Slice Flip Flops	825	9312	8%
Number of 4 input LUTs	343	9312	3%
Number of bonded IOBs	546	232	235%
Number of GCLKs	1	24	4%

```
ed Grade: -5

Minimum period: 2.131ns (Maximum Frequency: 469.274MHz)
Minimum input arrival time before clock: 3.100ns
Maximum output required time after clock: 6.183ns
Maximum combinational path delay: 4.655ns

================================================================
```

Fig. 9: Synthesis report of existing buffer.



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	157	4656	3%
Number of Slice Flip Flops	7	9312	0%
Number of 4 input LUTs	275	9312	2%
Number of bonded IOBs	410	232	176%
Number of GCLKs	1	24	4%

```
Minimum period: 2.131ns (Maximum Frequency: 469.274MHz)
Minimum input arrival time before clock: 2.503ns
Maximum output required time after clock: 5.887ns
Maximum combinational path delay: 5.812ns
```

Fig. 10: Synthesis report of Modified paper buffer.

TABLE.1. Area, Time, and Power Complexities of 1D-DCT architectures of Integer DCT for Various Lengths.

Architecture type	N(point)	AREA		Delay (ns)	Adders / sub tractors	Power Consumption (Watts)	Computation time (ns)
		LUT's	Slices				
Reference Algorithm	4	240	136	16.49	40	0.097	2.84
	8	928	505	25.01	64	0.143	3.25
	16	1854	3505	25.57	256	6.402	3.67
	32	12995	7006	32.98	843	25.53	3.9
Existing Algorithm before pruning	4	185	120	16.00	35	0.081	2.95
	8	307	211	24.82	50	0.122	3.34
	16	1800	3041	22.11	241	4.517	3.92
	32	10177	5416	30.24	837	19.63	4.48
Existing Algorithm before pruning	4	135	103	15.08	30	0.042	2.93
	8	238	222	21.65	50	0.108	3.31
	16	1415	2405	22.00	230	2.858	3.92
	32	5402	2976	28.70	703	17.63	4.45

TABLE.2. Area, delay, computation time of proposed and existing architectures of transpose buffer.

Transpose buffer	Area		Delay (ns)	Clock frequency (MHZ)	Computation time (ns)
	LUT's	Slices			
4 point Existing buffer	608	343	6.18	469.72	3.10
4 point Proposed buffer	157	275	5.88	469	2.50
8point Existing buffer	1200	600	9.10	600.80	6.12
8point Proposed buffer	375	400	6.15	589.75	4.20

TABLE.3. Area, delay, computation time of 2D DCT 4 point proposed and existing architectures.

4-point	Architecture type	AREA		Delay (ns)	Clock frequency (MHZ)	Adders/ sub tractors	Power Consumption (Watts)	Computation time (ns)
		LUT'S	Slices					
Folded architecture	Existing	698	415	21	404	116	4.447	2.97
	proposed	610	390	17	300	76	3.161	2.8
Parallel Architecture	Existing	706	419	17.8	449	115	4.455	2.544
	proposed	550	378	17	414	116	3.211	2.50

TABLE.4. Area, delay, computation time of 2D DCT 8 point proposed and existing architectures.

8-point	Architecture type	AREA		Delay (ns)	Clock frequency (MHZ)	Adders/ sub tractors	Power Consumption (Watts)	Computation time (ns)
		LUT'S	Slices					
Folded architecture	Existing	6212	3618	20.59	389	808	9.562	2.97
	proposed	5633	3018	19.19	372	807	7.013	2.8
Parallel Architecture	Existing	3618	6212	19.09	400	808	9.567	2.544
	proposed	2853	1809	18.44	389	807	7.016	2.50

CONCLUSION

In this paper implement the efficient area, delay and power architectures for 2D DCT of different lengths. Existing Reusable architecture involves less area, delay and power than the general architecture of N point DCT and DCT that can compute the DCT of lengths 4, 8, 16, and 32 with throughput of 32 output coefficients per cycle. Two different architectures are implemented for 2D DCT. Transpose buffer plays important role in folded and parallel structures to match the rows and columns of intermediate output. By replacing the AND gates with up counter in proposed structure, reduce the area and delay, power. From the synthesis result it is found that proposed transposition buffer involves less area and delay than the existing transpose buffer.

REFERENCES

[1]. Pramod kumar, Meher, Sang Yoon Park"Efficient Integer DCT Architecture for HEVC" IEEE transactions on circuits and systems for video technology, vol. 24, no. 1, January 2014.

[2]. B. Bross, W.-J. Han, J.-R. Ohm, G. J. Sullivan, Y.-K. Wang, andT. Wiegand, High Efficiency Video Coding (HEVC) Text Specification Draft 10 (for FDIS and Consent), JCT-VC L1003, Geneva, Switzerland, Jan. 2013.

[3]. G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," IEEE Trans. CircuitsSyst. Video Technol., vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[4]. Ahmed, M. U. Shahid, and A. Rehman, "N Point DCT VLSI Architecture for Emerging HEVC Standard," in Proc. VLSI Design, vol. 2012, Article 752024, pp. 1–13, 2012.

[5]. J.-S. Park, W.-J. Nam, S.-M. Han, and S. Lee, "2-D large inverse Transform (16x16, 32x32) for HEVC (high efficiency video coding)," J. Semicond. Technol. Sci., vol. 12, no. 2, pp. 203–211, Jun. 2012.

[6]. S. Shen, W. Shen, Y. Fan, and X. Zeng, "A unified 4/8/16/32-point integer IDCT architecture for multiple video coding standards," in Proc. IEEE Int. Conf. Multimedia Expo, Jul. 2012, pp. 788–793.

[7]. A. Fuldseth, G. Bjøntegaard, M. Budagavi, and V. Sze. (2011, Nov.). JCTVC-G495, CE10: Core Transform Design for HEVC: Proposal for Current HEVC Transform [Online]. Available: http://phenix.int-evry.fr/ jct/doc end user/documents/7 Geneva/wg11/JCTV%C-G495-v2.zip

[8]. N. Boullis and A. Tisserand, "So optimizations of hardware multiplication by constant matrices," IEEE Trans. Comput., vol. 54, no. 10, pp. 1271–1282, Oct. 2005.

[9]. W. Cham and Y. Chan, "An order-16 integer Cosine Transform," IEEE Trans. Signal Process., vol. 39, no. 5, pp. 1205–1208, May 1991.

[10]. N. Ahmed, T. Natarajan, and K. Rao, "DiscreteCosine Transform," IEEE Trans. Comput., vol. 100, no. 1, pp. 90–93, Jan. 1974.