High Performance Integer DCT Architectures For HEVC

V.Sruthi[1], V.Rekha[2], M. Subitha[3], M.Sugitha[4], S. Jeya Anusuya M.E.[5], V. Satheesh kumar M.E.[6]
[1, 2, 3, 4,]Dept Of Electronics and Communication Engineering
[5, 6]Associate professor, Dept Of Electronics and Communication Engineering
[1, 2, 3, 4, 5, 6]T.J.S Engineering College, Peruvoyal

Abstract*- Our proposed system proceeds VLSI architecture for integer Discrete Cosine Transform (integer DCT), which is used in real time High Efficiency Video Coding (HEVC) applications. It has N-point 1D-Integer DCT architecture, which includes signed configurable carry save adder tree based multiplier unit. So, the depth of the architecture falls within the bounds of O (log2 N). The proposed 1D architecture is used to perform one N-point or Integer DCTs in parallel. The proposed 1D architecture is used to design 2D folded and parallel designs. The performance results show that the proposed architecture gives better performance compared with existing architectures using 45 nm CMOS TSMC libraries. The proposed 32*32-point parallel Integer DCT achieves 59.1% of improvement in worst path delay compared with odd-even decomposition based architecture.*

Keywords- Integer DCT, HEVC, 1D DCT Architecture, 2D DCT Architecture

I. INTRODUCTION

Digital signal processors (DSPs) are very important for the real-time processing of real-world digitized data to do high-speed numeric calculations used for lot of applications from basic consumer electronics to sophisticated industrial instrumentation. The discrete transform is used to change the representation of a signal from one domain to another for reducing the complexity of a particular digital signal processing application.

Discrete cosine transform (DCT) is very powerful transformation used in image compression. The circuit complexity of DCT is greater than integer DCT because DCT is floating point and the integer DCT is fixed point. So, the delay of the multiplier adders used in the adder. The output can be stored at one particular 1*32 Buffer. The outputs of Ith, 1* 32-Buffer are b32i, b16i, b8i, b4i, and b2i, which are the resultants of 32, 16, 8, 4, and 2-point Integer DCTs respectively. Each 1*32-Buffer is made up of 32 numbers of registers and 2-to-1 multiplexers with common select line.
Multimedia communication typically involves the transfer of large amount of data Therefore, compression of video, audio, and image data is essential for a cost-efficient use of existing communication channels and storage media. The DCT helps separate the image into parts of differing importance with respect to the image's visual quality. The DCT chip presented here will form a part of one such image compression system. The system is based on a 2-D block cosine transform coding scheme, where the image is of size 256x256 and each block is of size 8x8. There are two main computational task involved. The first consists of computing the 2-D DCT on blocks of size 8x8, while the second task consists of quantizing the transform coefficients using scalar quantizes. We present an implementation of a chip that computes the DCT of an 8x8 element block.

The DCT application can have many purposes Such as filtering, teleconferencing, high-definition television (HDTV), speech coding, image coding, data compression, and more. All of these use DCT algorithm for compression and/or filtering purposes. The DCT has the energy packing capabilities and also approaches the statistically optimal transform in decor relating a signal. It was implemented with discrete components at the board level. This was followed by its implementation using general purposes (DSP) chips. Also, image compression boards and multiprocessor workstations based on DCT have been developed by industry. For our project, it is using the algorithm for image compression purpose. With high speed and low power design, it is best for handheld device use. Such device consumes power from its battery. It is an impact to have low power consumption for the device, because battery carry limited power. Therefore, the design must have low power consumption components to compose the chip. Otherwise, the device will be force to offline due to insufficient power supply. Furthermore, high-speed algorithm is necessary for urge of current software and operating system. The performance of the chip is optimized and specified for image compression purposes.

II. LITERATURE SURVEY

High performance Multiplier less DCT Architecture for HEVC, Wenjun Zhao, Takao Onoye, and Tian Song(2015)

There are numerous video compression format for storage or transmission of digital video content. High Efficiency Video Coding (HEVC) is a video compression standard, a successor to H.264/MPEG-4 Advanced Video Coding (AVC). In this paper, we propose an efficient architecture for the computation of 4, 8, 16 and 32 point DCT used in HEVC standard. The architecture uses the Canonical Signed Digit (CSD) representation and Common Sub-expression Elimination (CSE) technique to perform the multiplication with shift-add operation.

A Reconfigurable Multi-Transform VLSI Architecture Supporting Video Codec Design Kanwen Wang, Jialin Chen, Wei Cao, Ying Wang, Lingli Wang.

The proposed system for the real-time processing of 1080P HD video, which can support both forward and inverse transforms of MPEG using multi transform VLSI architecture. The MCM (RMCM) algorithm is the multiple constant multiplication algorithm with 2 fusing strategies, which is provided to generate constant multipliers in the matrix calculation blocks.

Multi-mode parallel and folded VLSI architectures for 1D-fast Fourier transform Mohamed Asan Basiri M and Noor Mahammad Sk.

This paper proposes efficient FFT VLSI architectures using folded/parallel implementation. The folded FFT architecture has number of cycles required to complete the operation is less than single/multi-path delay commutator (MDC) architectures. *N*-point FFT is implemented by using one *N*/2-point FFT without much extra hardware. Both the proposed architectures are implemented for radix-2, 2^2.

III. EXISTING SYSTEM

In all the existing architectures, thread-shift network based multiplier is used. So, the delay of the multiplier is based on the number of adders used in the add-shift network. The existing technique is add-shift network. It uses configurable carry save addition

Disadvantages of Existing System:

- Multiplication requirement is more.
- More delay
- High Power

IV. PROPOSED SYSTEM

In the proposed architecture, configurable carry save adder (CSA) tree based multiplier is used. It shows the series of multiplexers used for configurable carry save addition based multiplication in the proposed architecture. The maximum number of values to be added in the configurable carry save addition based 32-point Integer DCT is log2N = log232 = 5.IV.

The mathematical representations of the 2-D Forward DCT and the 2-D IDCT are represented in the following:

Formulae

Forward DCT

$$F(u,v)=C(u)C(v)[\sum_{x=0}^{(N-1)}\sum_{y=0}^{(N-1)} f(x,y)\cos\frac{(2x+1)u\pi}{2N}\cos\frac{(2y-1)v\pi}{2N}]$$

Inverse DCT

$$f(x,y)=[\sum_{u=0}^{(N-1)}\sum_{v=0}^{(N-1)} C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{2N}\cos\frac{(2y-1)v\pi}{2N}]$$

Where: C(u) = $\frac{1}{\sqrt{N}}$, C(v) = $\frac{1}{\sqrt{N}}$
for u,v = 0

C(u) = $\sqrt{\frac{2}{N}}$, C(v) = $\sqrt{\frac{2}{N}}$ for u,v = 1 through N-1;
N = 4, 8, or 16

In the design, N = 8. F(u,v) is called the (u,v)th transform coefficient. The above formula shows that the 2-D DCT can be computed by applying the 1-D DCT to each of the columns of the matrix separately and then applying the 1-D DCT to each of the rows separately. This is the reparability property of the 2-D DCT. All the 2-D DCT processors developed so far have made use of this property of the 2-D DCT. In this report, we present the design of the 2-D DCT function under VLSI architecture for image processing. The design layout will be at cells block level, which it does not show in great detail for the entire chip design.

V. WORKING PRINCIPLE

DCT Algorithm

2-D DCT Architecture: The two dimensional (2-D) Discrete Cosine Transform (DCT) forms the cornerstone of many image processing standards such as JPEG and MPEG. Many proposed solutions are based on “row column decomposition” implementation which allows the 2-D DCT to be implemented by two one dimensional (1-D) DCTs separated by a transposition memory.

1-D DCT Architecture: The derivation of the l-D DCT architecture can be more easily explained by examining the l-D DCT in matrix form, given as below:

$$[Y]=[C]*[X]$$

$$\begin{bmatrix} Y(0)\\ Y(1)\\ Y(2)\\ Y(3)\\ Y(4)\\ Y(5)\\ Y(6)\\ Y(7) \end{bmatrix} = \sqrt{\frac{1}{4}} \begin{bmatrix} C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 \\ C_1 & C_3 & C_5 & C_7 & -C_7 & -C_5 & -C_3 & -C_1 \\ C_2 & C_6 & -C_6 & -C_2 & -C_2 & -C_6 & C_6 & C_2 \\ C_3 & -C_7 & -C_1 & -C_5 & -C_5 & C_1 & C_7 & -C_3 \\ C_4 & -C_4 & -C_4 & C_4 & C_4 & -C_4 & -C_4 & C_4 \\ C_5 & -C_1 & C_7 & C_3 & -C_3 & -C_7 & C_1 & -C_5 \\ C_6 & -C_2 & C_2 & -C_6 & -C_6 & C_2 & -C_2 & C_6 \\ C_7 & -C_5 & C_3 & -C_1 & C_1 & -C_3 & -C_5 & -C_7 \end{bmatrix} * \begin{bmatrix} X(0)\\ X(1)\\ X(2)\\ X(3)\\ X(4)\\ X(5)\\ X(6)\\ X(7) \end{bmatrix}$$

Where C(k) = Cos(2∏K/32). As multipliers are *m* times more complex than adders, the aim is to reduce the number of multiplications at the expense of additions. The sparse matrix approach achieves this by manipulating the terms in the input matrix as shown in equation.

$$\begin{bmatrix} Y(0)\\ Y(2)\\ Y(4)\\ Y(6)\\ Y(1)\\ Y(5)\\ Y(6)\\ Y(7) \end{bmatrix} = \sqrt{\frac{1}{4}} \begin{bmatrix} C_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -C_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_2 & C_6 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_6 & -C_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & C_1 & C_3 & C_5 & C_7 \\ 0 & 0 & 0 & 0 & C_3 & -C_7 & -C_1 & -C_5 \\ 0 & 0 & 0 & 0 & C_5 & -C_1 & C_7 & C_3 \\ 0 & 0 & 0 & 0 & C_7 & -C_5 & C_3 & -C_1 \end{bmatrix} * \begin{bmatrix} X(0)+X(1)+X(2)+X(3)+X(4)+X(5)+X(6)+X(7) \\ X(0)+X(7)+X(3)+X(4)-X(1)-X(2)-X(5)-X(6) \\ X(0)+X(7)-X(3)-X(4) \\ X(1)+X(6)-X(2)-X(5) \\ X(0)-X(7) \\ X(1)-X(6) \\ X(2)-X(5) \\ X(3)-X(4) \end{bmatrix}$$

The circuit consists of 38 multipliers and 8 adders and 8 subtracters connected in a regular matrix of cells. Bit serial logical adders and subtracter cells have been used and the array multipliers have been implemented. The bit serial multiplier will be pipelined every two cells. By using K-Map, The serial logical adder and subtracter equation is implemented. The following table showing the equations:

Sum = A ⊕ B ⊕ Cin

Difference = A ⊕ B ⊕ bin

Carry = AB + ACin +BCin　　　　Borrow
= A’B + A’bin + Bbin

Consider the two unsigned binary numbers X and Y that are M and N bits wide respectively. X and Y in a binary representation are as below:

$$x=\sum_{i=0}^{M-1} x_i 2^{i} \quad , \quad y=\sum_{j=0}^{N-1} y_j 2^{j}$$

With Xi, Yj € {0,1}. The multiplication operation is then defined as follow:

$$z = x * y = \sum_{k=0}^{M+N-1} z_k 2^{x}$$

$$= \left(\sum_{i=0}^{M-1} x_i 2^{i}\right)\left(\sum_{j=0}^{N-1} y_j 2^{j}\right)$$

$$= \sum_{i=0}^{M-1}\left(\sum_{j=0}^{N-1} x_i y_j 2^{i+j}\right)$$

The multiplicand is consecutively multiplied with every bit of the multiplier, resulting in a number of partial products. These intermediate results are adder after the proper shifting has been applied. Use the algorithms of two binary number multiplications to implement the array multiplier. The array multiplier consists numerous of AND and full adder. This type of multiplier requires M-bit (Multiplicand) x N-bit (Multiplier) number of AND gates and full adders. The transpose component is an array of 8x8, 12-bit shift registers. It receives the output from the 1-D DCT (row), and transposes the row to the column of the second 1-D DCT input. The shift register used to store the bits into registers. Then, connect the metal plate to other shifter’s input, and shift the each 12 bits arrange as column format.

VI. FLOWCHART

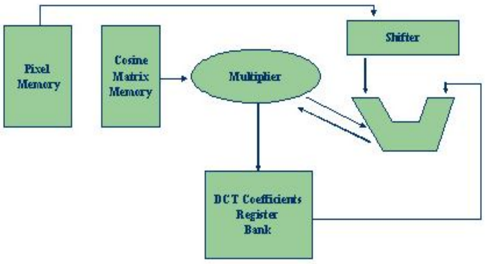


Fig.1 Flow chart of 1D DCT Architecture

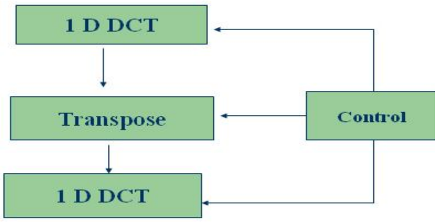
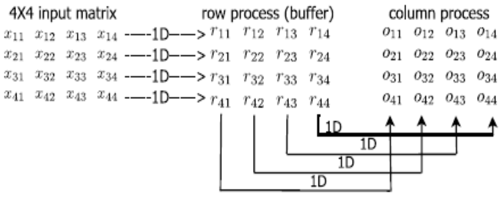


Fig.2 Flow chart of 2D-DCT Architecture

VII. PROPOSED DCT ARCHITECTURE

Proposed block architecture used for 32-point 1D-Integer DCT. In 32-point 1D-Integer DCT, the co-efficient matrix is in the size of 32_32. The input signal sample values should be multiplied with the co-efficient, which forms the matrix-vector multiplier.


4X4 input matrix
row process (buffer)
column process
x_{11} x_{12} x_{13} x_{14} ----1D---> r_{11} r_{12} r_{13} r_{14} o_{11} o_{12} o_{13} o_{14}
x_{21} x_{22} x_{23} x_{24} ----1D---> r_{21} r_{22} r_{23} r_{24} o_{21} o_{22} o_{23} o_{24}
x_{31} x_{32} x_{33} x_{34} ----1D---> r_{31} r_{32} r_{33} r_{34} o_{31} o_{32} o_{33} o_{34}
x_{41} x_{42} x_{43} x_{44} ----1D---> r_{41} r_{42} r_{43} r_{44} o_{41} o_{42} o_{43} o_{44}
1D
1D
1D
1D

Example for row and column process of 4 × 4-point 2D-Integer DCT

In all the existing architectures, the add-shift network based multiplier is used. So, the delay of the multiplier is based on the number of adders used in the add-shift network. In the proposed architecture, configurable carry save adder (CSA) tree based multiplier is used. Fig. 3(a) shows the series of multiplexers used for configurable carry save addition based multiplication in the proposed architecture.

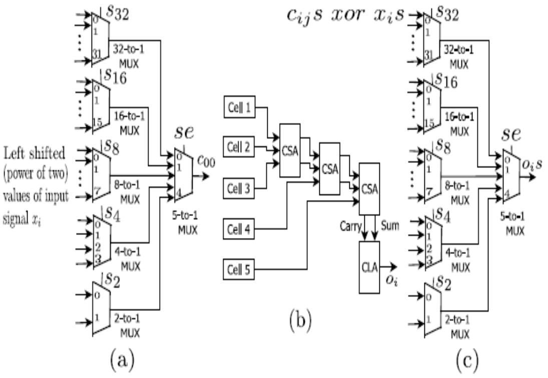


Fig.3 Proposed Architecture

The maximum number of values to be added in the configurable carry save addition based 32-point Integer DCT is log2N = log232 = 5. For example, the multiplication of the co-efficient 87 with the input signal sample value xi is equal to 87xi = 64xi +16xi +4xi +2xi +xi. The minimum number of values to be added in the configurable carry save addition based 32-point Integer DCT is 1. For example, the multiplication of the co-efficient 4 with the input signal sample value xi is equal to 4xi = 4xi + 0xi + 0xi + 0xi + 0xi. So, the corresponding left-shifted (power of two) input signal values are sent as the input of the series of multiplexers used in Fig. 3(a), which is named as Cell. The maximum possible cells used to obtain one multiplication result is 5. Therefore, five Cells are used in Fig. 3(b). So, the maximum possible levels of the configurable carry save adder (CSA) tree is log25 = 3. The Sum and Carry from the final carry save adder are added. The proposed block architecture (Block) used for 32-point 1D-Integer DCT with (a) Series of multiplexers used for configurable carry save addition based multiplication (Cell) (b) configurable carry save adder tree based multiplication unit (c) Series of multiplexers used to find the resultant sign bits for the multiplication.

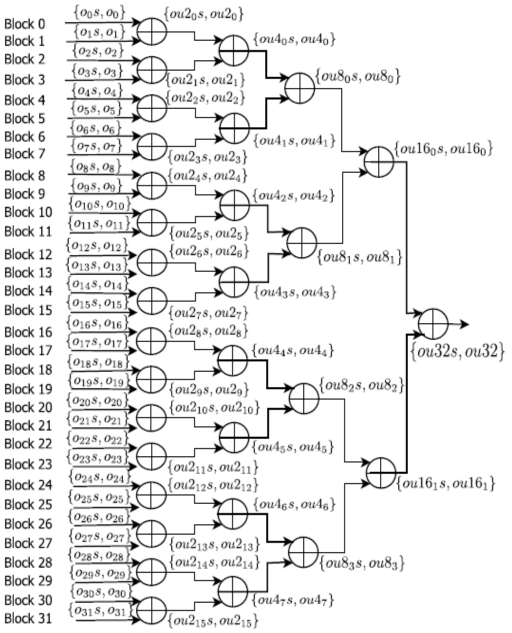


Fig.4 VLSI architectures for proposed 32-point 1D-Integer DCT

The overall architecture of proposed 32-point 1D-Integer DCT, where the inputs are from 32 numbers of Blocks as shown in Fig.4. Therefore, log232 = 5 levels of signed fixed point adders are used. Therefore, the critical path depth of the signed adder tree (Tadd; pro delay) used in the N-point proposed Integer DCT architecture is (log2N)T(add). Here, T (add) represents the critical path depth of the signed adder. The proposed 32-point 1D architecture is used to perform one 32-point or two 16-point or four 8-point or eight 4-point or sixteen 2-point Integer DCTs in parallel. The 32-point Integer DCT output is fou32s; ou32g. Fig.5 shows the 32 X 32-Buffer architecture, where 32 numbers of 1*32-Buffers are used. The 1*32-Buffer inputs are the outputs from the column of 5-to-1 multiplexers, with select line se. Here, se = 0; 1; 2; 3, and 4 for 32; 16; 8; 4; and 2-point Integer DCTs respectively. Each 1*32-Buffer is made up of 32 numbers of registers and 2-to-1 multiplexers with common select line. The select lines used in the 1*32-Buffers 0, 1, ... 30, and 31 are en0, en1,...en30, and en31 respectively. The output from Fig. 4 can be stored at one particular 1*32-Buffer with corresponding select line as 1. The 1*32-Buffer architecture is shown in Fig. 6. The outputs of ith 1*32-Buffer are b32i, b16i, b8i, b4i, and b2i, which are the resultants of 32, 16, 8, 4, and 2-point Integer DCTs respectively. Here, eni = 0 to maintain the values (32 values) stored in the buffer and eni = 1 if the the new value.

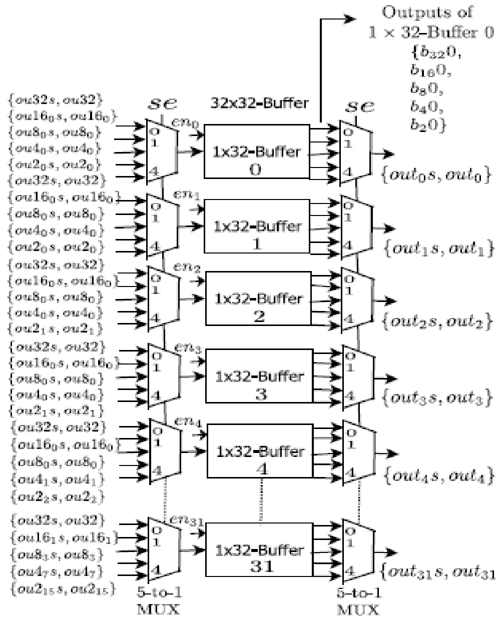


Fig.5 VLSI architectures for 32 X 32-Buffer

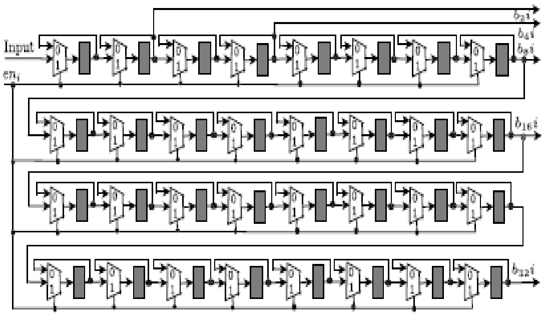


*Fig.6 Architecture of 1*32-Buffer*

1* 32-Buffer architecture: The output from Fig. 4 can be stored at one particular 1*32-Buffer with corresponding select line as 1. The 1*32-Buffer architecture is shown in Fig. 5. The outputs of ith 1 * 32-Buffer are b32i, b16i, b8i, b4i, and b2i, which are the resultants of 32, 16, 8, 4, and 2-point Integer DCTs respectively. Here, eni = 0 to maintain the values (32 values) stored in the buffer and eni = 1 if the the new value is obtained.

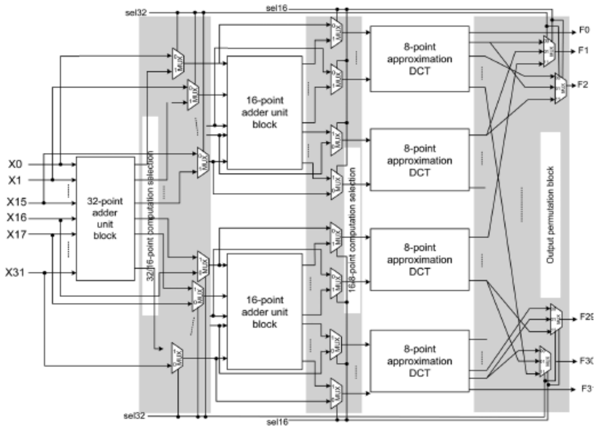
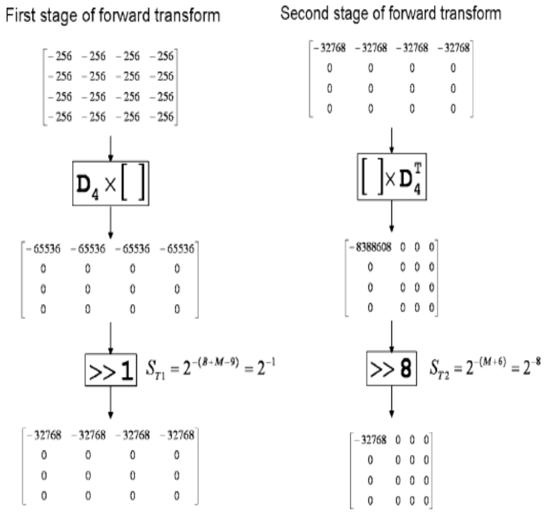

sel32
sel16
X0
X1
X15
X16
X17
X31
32-point adder unit block
16-point adder unit block
16-point adder unit block
8-point approximation DCT
8-point approximation DCT
8-point approximation DCT
8-point approximation DCT
32/16-point computation selection
16/8-point computation selection
Output permutation block
F0
F1
F2
F29
F30
F31

Fig.7 Proposed 32 DCT

FORWARD TRANSFORM (DCT):

First Stage of Forward Transform: The first stage of the forward transform consists of multiplication of the result of the D4. The input into the second stage of the forward transform is the output matrix from the first stage of forward transform which is a matrix with only the DC element. The output of multiplication with DT4 will be a matrix with first column elements. Consequently, the scaling required after the first stage of the forward transform for the output to fit within 16 bits is $S_{T1}{}^{=}\ 2^{-(B-M+9)}$.



Second Stage of Forward Transform: The second stage of the forward transform consists of multiplication of the result of the first transform stage with D4. The input into the second stage of the forward transform is the output from the first stage which is a matrix with all elements in the first row. All other elements will be zero. The output of multiplication with will be a matrix with only a DC value. This implies that the scaling required after the second stage of transform is in $S_{T2}{}^{=}\ 2^{-(21-B)}$ order for the output to fit within 16 bits.

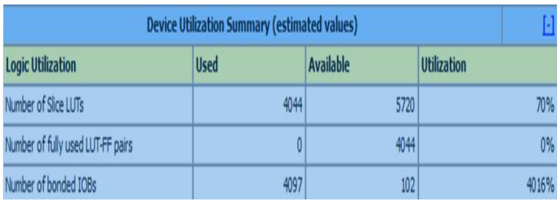
VIII. TECHNOLOGY USED

The multiplier unit used in the latest N-point Integer DCT architectures is in the form of add-shift network, whereas in the proposed architecture, signed configurable carry save adder tree is used. Therefore, the depth of the architecture falls within the bounds of O(log2 N). The proposed 1D architecture is used to perform one N-point or multiple N 2 , N 4 , ...2-point Integer DCTs in parallel. The performance results show that the proposed architecture gives better performance compared with existing architectures using 45 nm CMOS TSMC libraries.

Model Sim software is used to check this model. For IC development they have three processes in Xilinx software.

a. Check syntax
b. Pin assignment
c. Implementation

Check syntax is used to check our design having any error. After finishing this process we allocate the input and output pins by using pin assignment. In the final process is implementation. Here we implement the design into our assigning pins. Then we convert our code into bit file then after we dump this bit file in to FPGA spartan3 (XC3S 400 PQ208) and verified it.



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	4044	5720	70%
Number of fully used LUT-FF pairs	0	4044	0%
Number of bonded IOBs	4097	102	4016%

Device Utilization Summary 32-Point DCT

IX. ADVANTAGES

- Better Compression Performance
- Computation Performance is good

X. APPLICATIONS

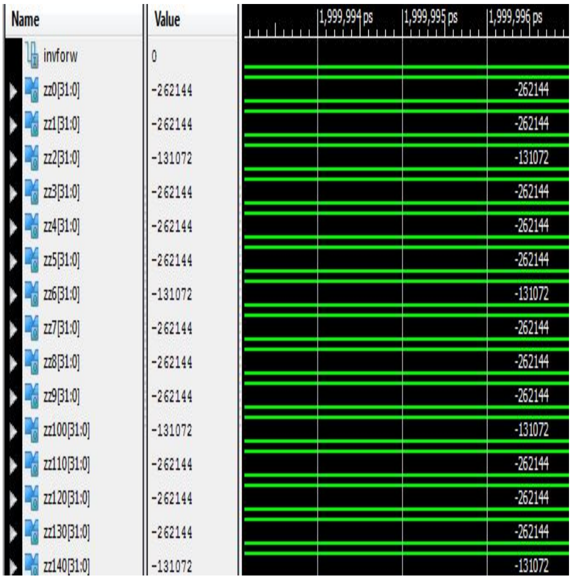
- Used in health department.
- Human welfare
- It is used to monitor industrial radiation levels.

XI. ACKNOWLEDGEMENT

We would like to thank all those who provide us the possibility to propose this project. A special gratitude to our Project Guide Mr. V. Satheesh kumar, Assistant professor and Project coordinator Mrs. S. Jeya Anusuya, Associate professor, whose contribution in giving suggestions and encouragement helped us for this project to complete.

XII. RESULTS AND CONCLUSION

The multipliers are, at less, twice faster than the conventional design, and consume half of the power. This can be done by ignore the zeros' in the multiply constant and the insignificant parts of the answer. The circuit is further reduced. Therefore, it consumes less power. The reduction will more stages, because the scares of resources. This can be countered by doing multi-stage in one period. The multiplier operation will take one period and one or two adders operations will perform in one period. Then, there will be less power consumption without comprising the speed. The 1 D DCT and Transpose is finished and the simulation is shown above. The area of the 1D DCT chip is .8mm x .6mm. The total delay 258.16ns. The Transpose is 292ns. The area is .2mmx.5mmThe results is the same as the calculation. We don't have time to construct the 2 D DCT. But, it is simple. It just need to connect two 1D DCT to the transpose. The performance results show that the proposed architecture gives good improvement as compared with existing architectures. The Snapshot below gives the clear elaboration of application.



Name	Value
invforw	0
zz0[31:0]	-262144
zz1[31:0]	-262144
zz2[31:0]	-131072
zz3[31:0]	-262144
zz4[31:0]	-262144
zz5[31:0]	-262144
zz6[31:0]	-131072
zz7[31:0]	-262144
zz8[31:0]	-262144
zz9[31:0]	-262144
zz100[31:0]	-131072
zz110[31:0]	-262144
zz120[31:0]	-262144
zz130[31:0]	-262144
zz140[31:0]	-131072

Fig.8 Simulation Snapshot of 32-POINT DCT

REFERENCES

[1] Mohamed Asan Basiri M and Noor Mahammad Sk, "Multimode Parallel and Folded VLSI Architectures for 1D-Fast Fourier Transform", Integration, the VLSI Journal, Elsevier, vol. 55, pp. 43-56, Sept. 2016.

[2] Fei Liang, Xiulian Peng, and Jizheng Xu2, "A light-weight HEVC encoder for image coding", IEEE

International Conference on Visual Communications and Image Processing (VCIP), pp. 1-5, Nov. 2013.

[3] Pramod Kumar Meher, Sang Yoon Park, Basant Kumar Mohanty, Khoon Seong Lim, and Chuohao Yeo,, “Efficient Integer DCT Architectures for HEVC”, IEEE Transactions on Circuits and Systems for Video Technology, vol. 24, no. 1, pp. 168- 178, Jan. 2014.

[4] Pai-Tse Chiang and Tian Sheuan Chang, “A Reconfigurable Inverse Transform Architecture Design for HEVC Decoder”, IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1006-1009, May 2013. 125.

[5] Honggang Qi, Qingming Huang, and Wen Gao, “A Low-Cost Very Large Scale Integration Architecture for Multi Standard Inverse Transform”, IEEE Transactions on Circuits and Systems - II, Express Briefs, vol. 57, no. 7, pp. 551-555, July 2010.

[6] Khan Wahid, Muhammad Martuza, Mousumi Das, and Carl McCrosky, “Resource Shared Architecture of Multiple Transforms for Multiple Video Codecs”, IEEE International Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 947-950, May 2011.

[7] Kanwen Wang, Jialin Chen, Wei Cao, Ying Wang, Lingli Wang, and Jiarong Tong, “A Reconfigurable Multi-Transform VLSI Architecture Supporting Video Codec Design”, IEEE Transactions on Circuits and Systems - II, Express Briefs, vol. 58, no. 7, pp. 432-436, July 2011.

[8] Yao Ziyou, He Weifeng, Hong Liang, He Guanghui, and Mao Zhigang, “Area and Throughput Efficient IDCT/IDST Architecture for HEVC Standard”, IEEE International Symposium on Circuits and Systems(ISCAS), pp. 2511-2514, June 2014.

[9] Hong Liang, He Weifeng, Zhu Hu, and Mao Zhigang, “A Cost Effective 2-D Adaptive Block Size IDCT Architecture for HEVC Standard”, IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1290-1293, Aug. 2013.

[10] Wenjun Zhao, Takao Onoye, and Tian Song, “High-Performance Multiplierless Transform Architecture for HEVC”, IEEE International Symposium on Circuits and Systems, pp. 1668-1671, May 2013.

[11] Mohamed Asan Basiri M and Noor Mahammad Sk, “An Efficient VLSI Architecture for Discrete Hadamard Transform”, IEEE International VLSI Design Conference, pp. 140-145, Jan. 2016.

[12] Ricardo Gonzalez, Benjamin M. Gordon, and Mark A. Horowitz, “Supply and Threshold Voltage Scaling for Low Power CMOS”, IEEE Journal of Solid State Circuits, vol. 32, no. 8, pp. 1210-1216, Aug. 1997. 126.