International Journal of Advanced Technology and Innovative Research

ISSN 2348–2370
Vol.07,Issue.11,
August-2015,
Pages:1952-1959

www.ijatir.org

Efficient Integer DCT Architectures for HEVC

N. MANJULA[1], D. SUBBA RAO[2], N. MALATHI[3]

[1]PG Scholar, Dept of VLSI & Embedded Systems, Siddhartha Institute of Engineering and Technology, Ibrahimpatnam, Hyderabad, Telangana, India, E-mail: manjulanalavothula@gmail.com.

[2]HOD, Dept of ECE, Siddhartha Institute of Engineering and Technology, Ibrahimpatnam, Hyderabad, Telangana, India, E-mail: subbu.dasari@gmail.com.

[3]Associate Professor Dept of ECE, Siddhartha Institute of Engineering and Technology, Ibrahimpatnam, Hyderabad, Telangana, India, E-mail: n.malathiraj@gmail.com.

Abstract: In this paper, we present High Efficiency Video Coding (HEVC) inverse transform for residual coding uses 2-D 4x4 to 32x32 transforms with higher precision as compared to H.264/AVC's 4x4 and 8x8 transforms resulting in an increased hardware complexity. In this paper, an energy and area efficient VLSI architecture of an HEVC-compliant inverse transform and de-quantization engine is presented. We implement a pipelining scheme to process all transform sizes at a minimum throughput of 2 pixel/cycle with zero-column skipping for improved throughput. We use data gating in the 1-D Inverse Discrete Cosine Transform engine to improve energy-efficiency for smaller transform sizes. A high-density SRAM-based transpose memory is used for an area-efficient design. This design supports decoding of 4K Ultra-HD (3840x2160) video at 30 frame/sec. The inverse transform engine takes 98.1 k gate logic, 16.4 Kbit SRAM and 10.82 pJ/pixel while the de-quantization engine takes 27.7 k gate logic, 8.2 Kbit SRAM and 1.10 pJ/pixel in 40 nm CMOS technology. Although larger transforms require more computation per coefficient, they typically contain a smaller proportion of non-zero coefficients. Due to this trade-off, larger transforms can be more energy-efficient. The proposed architecture is found to support ultrahigh definition 7680×4320 at 60 frames/s video, which is one of the applications of HEVC.

Keywords: Discrete Cosine Transforms (DCT), H.265, High Efficiency Video Coding (HEVC), Integer Discrete Cosine Transform (DCT), Video Coding.

I. INTRODUCTION

The High Efficiency Video Coding (HEVC) standard is the most recent joint video project of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) standardization organizat-ions, working together in a partnership known as the Joint Collaborative Team on Video Coding (JCTVC). The first edition of the HEVC standard is expected to be finalized in January 2013, resulting in an aligned text that will be published by both ITU-T and ISO/IEC. Additional work is planned to extend the standard to support several additional application scenarios, including extended-range uses with enhanced precision and color format support, scalable video coding, and 3-D/stereo/multi-view video coding. In ISO/IEC, the HEVC standard will become MPEG-H Part 2 (ISO/IEC 23008-2) and in ITU-T it is likely to become ITU-T Recommendation H.265. Video coding standards have evolved primarily through the development of the well-known ITUT and ISO/IEC standards. The ITU-T produced H.261 and H.263, ISO/IEC produced MPEG-1 and MPEG-4 Visual, and the two organizations jointly produced the H.262/MPEG-2 Video and H.264/MPEG-4 Advanced Video Coding (AVC) standards. The two standards that were jointly produced have had a particularly strong impact and have found their way into a wide variety of products that are increasingly prevalent in our daily lives. Throughout this evolution, continued efforts have been made to maximize compression capability and improve other characteristics such as data loss robustness, while considering the computational resources that were practical for use in products at the time of anticipated deployment of each standard.

The Discrete cosine transform (DCT) plays a vital role in video compression due to its near optimal de correlation efficiency. Several variations of integer DCT have been suggested in the last two decades to reduce the computational complexity. The new H.265/High Efficiency Video Coding (HEVC) standard has been recently finalized and poised to replace H.264/AVC. Some hardware architectures for the integer DCT for HEVC have also been proposed for its real time implementation decomposed the DCT matrices into sparse sub matrices where the multiplications are avoided by using the lifting scheme used the multiplier less multiple constant multiplication (MCM) approach for four-point and eight-point DCT, and have used the normal multipliers with sharing techniques for 16 and 32-point DCTs have used Chen's factorization of DCT where the butterfly operation has been implemented by the processing element with only shifters, adders, and multiplexors proposed a unified structure to be used for forward as well as inverse transform after the matrix decomposition. One key feature of HEVC is that it supports DCT of different sizes such as 4, 8, 16, and 32. Therefore, the hardware architecture should be flexible

enough for the computation of DCT of any of these lengths. The existing designs for conventional DCT based on constant matrix multiplication (CMM) and MCM can provide optimal solutions for the computation of any of these lengths, but they are not reusable for any length to support the same throughput processing of DCT of different transform lengths.

Considering this issue, we have analyzed the possible implementations of integer DCT for HEVC in the context of resource requirement and reusability, and based on that, we have derived the proposed algorithm for hardware implementation. We have designed scalable and reusable architectures for 1-D and 2-D integer DCTs for HEVC that could be reused for any of the prescribed lengths with the same throughput of processing irrespective of transform size. In this project, algorithms are used for hardware implementation of the HEVC integer DCTs of different lengths 4, 8, 16, and 32. H illustrate the design of the proposed architecture for the implementation of four-point and eight-point integer DCT along with a generalized design of integer DCT of length N, which could be used for the DCT of length N = 16 and 32. Moreover, it demonstrate the reusability of the proposed system. Here, power-efficient designs of transposition buffers for full-parallel and folded implementations of 2-D Integer DCT are used. Here a bit-pruning scheme for the implementation of integer DCT and present the impact of pruning on forward and inverse transforms are used. Finally comparing the synthesis result of the proposed architecture with those of existing architectures for HEVC.

II. BLOCK DIAGRAM

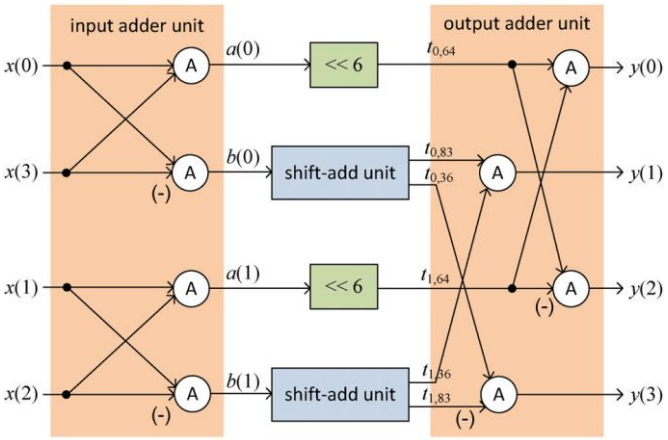

Fig.1. Proposed architecture of four-point integer DCT.

The proposed architecture for four-point integer DCT is shown in Fig.1. It consists of an input adder unit (IAU), a shift-add unit (SAU), and an output adder unit (OAU). The IAU computes a(0), a(1), b(0), and b(1) according to STAGE- 1 of the algorithm as described in Table I. The computations of ti,36 and ti,83 are performed by two SAUs according to STAGE-2 of the algorithm. The computation of t0,64 and t1,64 does not consume any logic since the shift operations could be rewired in hardware. The structure of SAU is Shown in Fig.2. Outputs of the SAU are finally added by the OAU according to STAGE-3 of the algorithm.

A. Structure of IAU

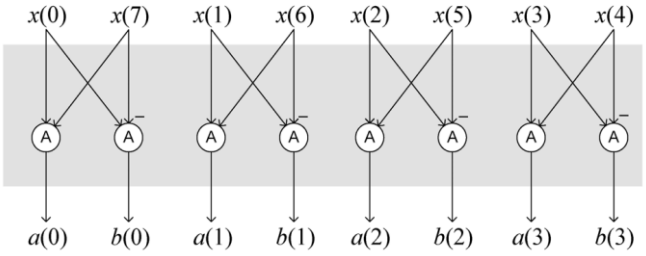

Fig.2. structure of IAU.

B. Proposed Generalized Architecture For Integer DCT of Lengths N = 8

The generalized architecture for N-point integer DCT based on the proposed algorithm is shown in Fig.3. It consists of four units, namely the IAU, (N/2)-point integer DCT unit, SAU, and OAU. The IAU computes a(i) and b(i) for i = 0, 1, ...,N/2 − 1 according to STAGE-1 of the algorithm of Section II-B. The SAU provides the result of multiplication of input sample with DCT coefficient by STAGE-2 of the algorithm. Finally, the OAU generates the output of DCT from a binary adder tree of $\log_2 N - 1$ stages, respectively, illustrates the structures of IAU, SAU, and OAU in the case of eight-point integer DCT. Four SAUs are required to compute ti,89, ti,75, ti,50, and ti,18 for i = 0, 1, 2, and 3 according to STAGE-2 of the algorithm. The outputs of SAUs are finally added by two-stage adder tree according to STAGE- 3 of the algorithm. Structures for 16- and 32-point integer DCT can also be obtained similarly.

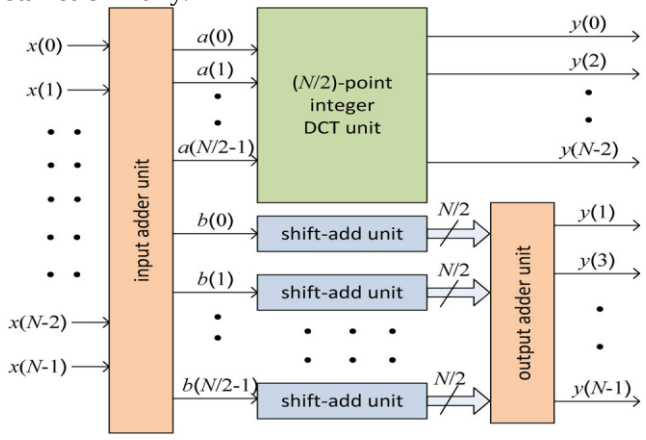

Fig.3. Proposed generalized architecture for integer DCT of lengths N = 8.

C. Proposed Reusable Architecture for N = 8

The proposed reusable architecture for the implementation of DCT of any of the prescribed lengths is shown in Fig. 4. There are two (N/2)-point DCT units in the structure. The input to one (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/2 − 1)], depending on whether it is used for N-point DCT computation or for the DCT of a lower size. The other (N/2)-point DCT unit takes the input [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 an array of (N/2) AND gates to disable this (N/2)-point DCT unit. The output of this (N/2)-point DCT unit is multiplexed with that of the OAU, which is preceded by the SAUs and IAU of the structure. The N AND gates before IAU 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, mN is used to decide the size of DCT computation. Specifically, for N = 32, m32 is a 2-bits 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 combinational logics for control units are shown in Fig. 4for N = 16 and 32, respectively. For N = 8, m8 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. The proposed structure can compute one 32-point DCT, two 16-point DCTs, four eight point DCTs, and eight four-point DCTs, while the throughput remains the same as 32 DCT coefficients per cycle irrespective of the desired transform size.

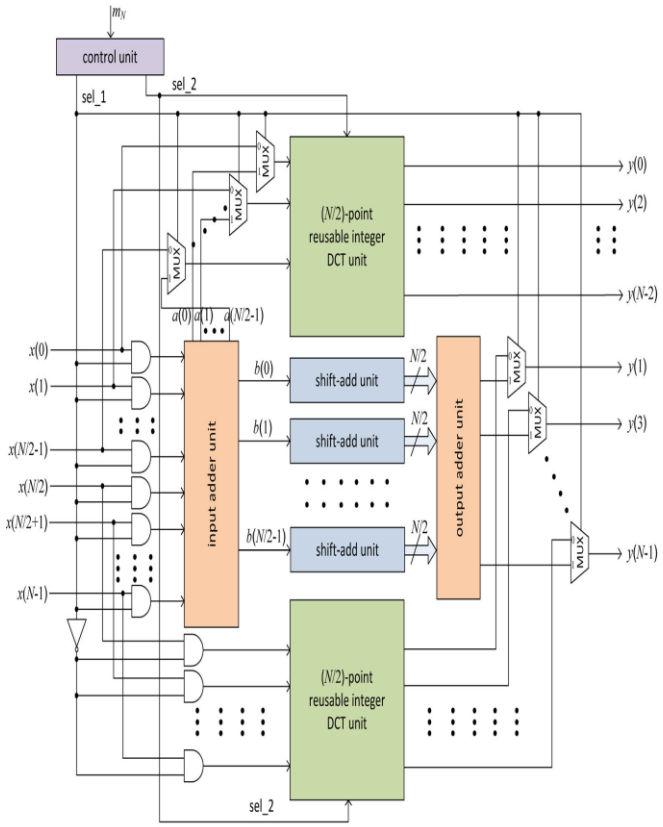

Fig.4. The proposed reusable architecture for the implementation of DCT.

D. Folded Structure for 2-D Integer DCT

The folded structure for the computation of (N × N)-point 2-D integer DCT is shown in Fig.5. It consists of one N-point 1-D DCT module and a transposition buffer. The structure of the proposed 4 × 4 transposition buffer is shown in Fig.5. It consists of 16 registers arranged in four rows and four columns. (N × N) transposition buffer can store N values in any one column of registers by enabling them by one of the enable signals ENi for i = 0, 1, ・ ・ ・,N −1. One can select the content of one of the rows of registers through the MUXes. During the first N successive cycles, the DCT module receives the successive columns of (N×N) block of input for the computation of STAGE-1, and stores the intermediate results in the registers of successive columns in the transposition buffer. In the next N cycles, contents of successive rows of the transposition buffer are selected by the MUXes and fed as input to the 1-D DCT module. N MUXes are used at the input of the 1-D DCT module to select either the columns from the input buffer (during the first N cycles) or the rows from the transposition buffer (during the next N cycles).

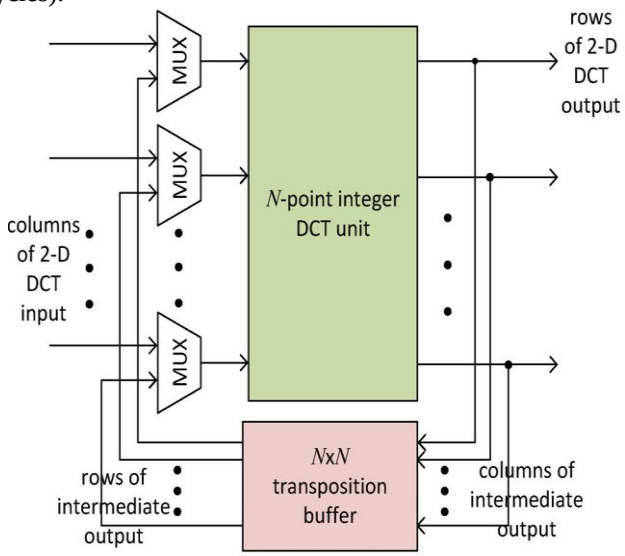

Fig.5. Folded Structure for 2-D Integer DCT.

E. Full-Parallel Structure for 2-D Integer DCT

The full-parallel structure for (N × N)-point 2-D integer DCT is shown in Fig.6. It consists of two N-point 1-D DCT modules and a transposition buffer. The structure of the 4 × 4 transposition buffer for full-parallel structure is shown in Fig. 6. It consists of 16 register cells (RC) [shown in Fig. 6] arranged in four rows and four columns. N × N transposition buffer can store N values in a cycle either row wise or column-wise by selecting the inputs by the MUXes at the input of RCs. The output from RCs can also be collected either row-wise or column-wise. To read the output from the buffer, N number of (2N −1):1 MUXes [shown in Fig. 6] are used, where outputs of the i^{th} row and the i^{th} column of RCs are fed as input to the i^{th} MUX. For the first N successive cycles, the i^{th} MUX provides output of N successive RCs on the i^{th} row. In the next N successive cycles, the i^{th} MUX provides output of N successive RCs on the i^{th} column. By this arrangement, in the first N cycles, we can read the output of N successive columns of RCs and in the next N cycles, we can read the output of N successive rows of RCs. The transposition buffer in this case allows both read and write

operations concurrently. If for the N cycles, results are read and stored column-wise now, then in the next N successive cycles, results are read and stored in the transposition buffer row-wise. The first 1-D DCT module receives the inputs column-wise from the input buffer.

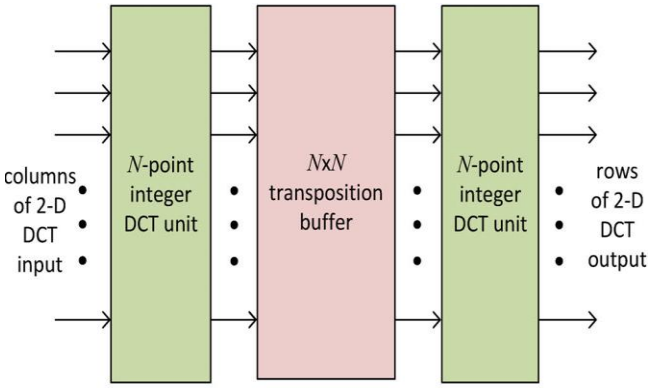

Fig.6. The full-parallel structure for (N × N)-point 2-D integer DCT.

III. APPLICATIONS

A. Proposed System Application

Discrete Cosine Transforms (DCT): A discrete cosine transform (DCT) expresses a sequence of finitely many data points in terms of a sum of cosine functions oscillating at different frequencies. DCTs are important to numerous applications in science and engineering, from lossy compression of audio and images, to spectral methods for the numerical solution of partial differential equations. The use of cosine rather than sine functions is critical in these applications: for compression, it turns out that cosine functions are much more efficient, whereas for differential equations the cosines express a particular choice of boundary conditions. In particular, a DCT is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers. DCTs are equivalent to DFTs of roughly twice the length, operating on real data with even symmetry, where in some variants the input and/or output data are shifted by half a sample. There are eight standard DCT variants, of which four are common.

Software Requirement

- **Verification Tool :** ModelSim 6.4c
- **Synthesis Tool :** Xilinx ISE 13.2

B. Modelsim

ModelSim SE - High Performance Simulation and Debug: ModelSim SE is our UNIX, Linux, and Windows-based simulation and debug environment, combining high performance with the most powerful and intuitive GUI in the industry.

ModelSim SE Benefits:

- High performance HDL simulation solution for FPGA & ASIC design teams.
- The best mixed-language environment and performance in the industry.
- Intuitive GUI for efficient interactive or post-simulation debug of RTL and gate-level designs.
- Merging, ranking and reporting of code coverage for tracking verification progress.
- Sign-off support for popular ASIC libraries.
- All ModelSim products are 100% standards based. This means your investment is protected, risk is lowered, reuse is enabled, and productivity is enhanced.
- Award-winning technical support.

C. High-Performance, Scalable Simulation Environment

ModelSim provides seamless, scalable performance and capabilities. Through the use of a single compiler and library system for all ModelSim configurations, employing the right ModelSim configuration for project needs is as simple as pointing your environment to the appropriate installation directory. ModelSim also supports very fast time-tenet-simulation turnarounds while maintaining high performance with its new black box use model, known as bbox. With bbox, non-changing elements can be compiled and optimized once and reused when running a modified version of the test bench. box delivers dramatic throughput improvements of up to 3X when running a large suite of test cases.

Easy-to-Use Simulation Environment: An intelligently engineered graphical user interface (GUI) efficiently displays design data for analysis and debug. The default configuration of windows and information is designed to meet the needs of most users. However, the flexibility of the ModelSim SE GUI allows users to easily customize it to their preferences. The result is a feature-rich GUI that is easy to use and quickly mastered. A message viewer enables simulation messages to be logged to the ModelSim results file in addition to the standard transcript file. The GUI's organizational and filtering capabilities allow design and simulation information to be quickly reduced to focus on areas of interest, such as possible causes of design bugs.

ModelSim SE allows many debug and analysis capabilities to be employed post-simulation on saved results, as well as during live simulation runs. For example, the coverage viewer analyzes and annotates source code with code coverage results, including FSM state and transition, statement, expression, branch, and toggle coverage. Signal values can be annotated in the source window and viewed in the waveform viewer. Race conditions, delta, and event activity can be analyzed in the list and wave windows. User-defined enumeration values can be easily defined for quicker understanding of simulation results. For improved debug productivity, ModelSim also has graphical and textual dataflow capabilities. The memory window identifies memories in the design and accommodates flexible viewing and modification of the memory contents. Powerful search, fill, load, and save functionalities are supported. The memory window allows memories to be pre-loaded with specific or randomly generated values, saving the time-consuming step of initializing sections of the simulation merely to load memories. All functions are available via the command line, so they can be used in scripting.

Advanced Code Coverage: The ModelSim advanced code coverage capabilities deliver high performance with ease of use. Most simulation optimizations remain enabled with code coverage. Code coverage metrics can be reported by-instance or by-design unit, providing flexibility in managing coverage data. All coverage information is now stored in the Unified Coverage DataBase (UCDB), which is used to collect and manage all coverage information in one highly efficient database. Coverage utilities that analyze code coverage data, such as merging and test ranking, are available.The coverage types supported include:

- **Statement coverage:** number of statements executed during a run.
- **Branch coverage:** expressions and case statements that affect the control flow of the HDL execution.
- **Condition coverage:** breaks down the condition on a branch into elements that make the result true or false.
- **Expression coverage:** the same as condition coverage, but covers concurrent signal assignments instead of branch decisions.
- **Focused expression coverage:** presents expression coverage data in a manner that accounts for each independent input to the expression in determining coverage results.
- **Enhanced toggle coverage:** in default mode, counts low-to-high and high-to-low transitions; in extended mode, counts transitions to and from X.
- **Finite State Machine coverage:** state and state transition coverage.

D. XILINX ISE Design Tools

Xilinx ISE is the design tool provided by Xilinx. Xilinx would be virtually identical for our purposes. There are four fundamental steps in all digital logic design. These consist of:

- Design – The schematic or code that describes the circuit.
- Synthesis – The intermediate conversion of human readable circuit description to FPGA code (EDIF) format. It involves syntax checking and combining of all the separate design files into a single file.
- Place & Route – Where the layout of the circuit is finalized. This is the translation of the EDIF into logic gates on the FPGA.
- Program – The FPGA is updated to reflect the design through the use of programming (.bit) files.

Test bench simulation is in the second step. As its name implies, it is used for testing the design by simulating the result of driving the inputs and observing the outputs to verify your design. ISE has the capability to do a variety of different design methodologies including: Schematic Capture, Finite State Machine and Hardware Descriptive Language (VHDL or Verilog).

IV. VARIOUS SNAPSHOTS

Results of this paper is as shown in bellow Snapshots. 1 to 10.

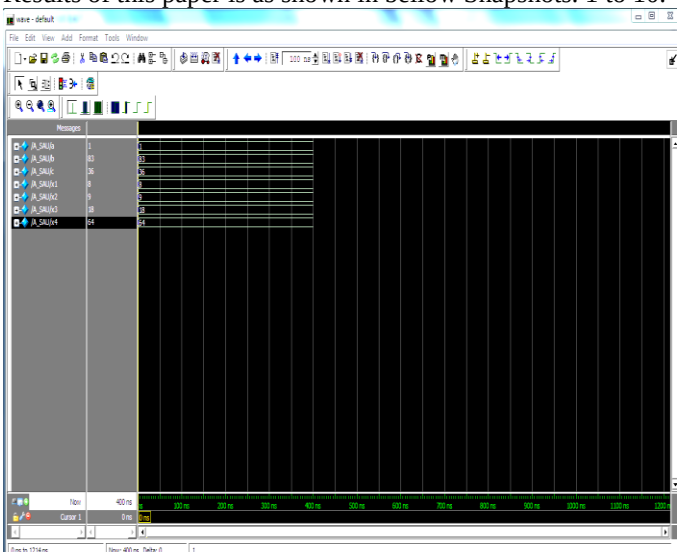

Fig1. Structure of SAU.

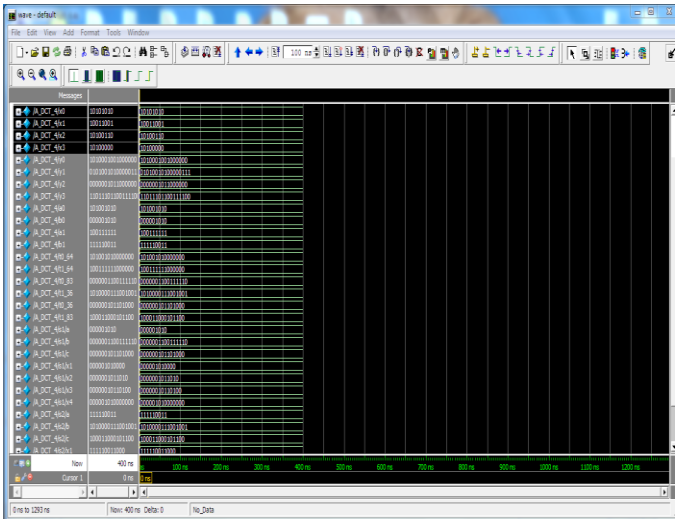

Fig2. Proposed architecture of four-point integer DCT.

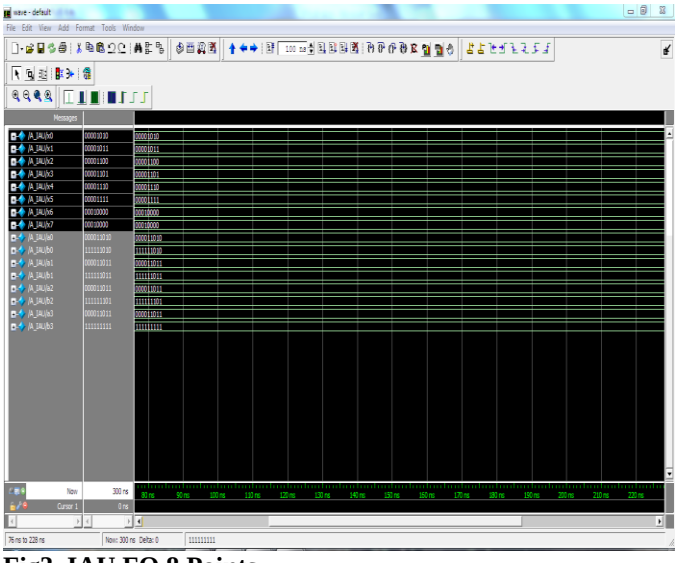

Fig3. IAU FO 8 Points.

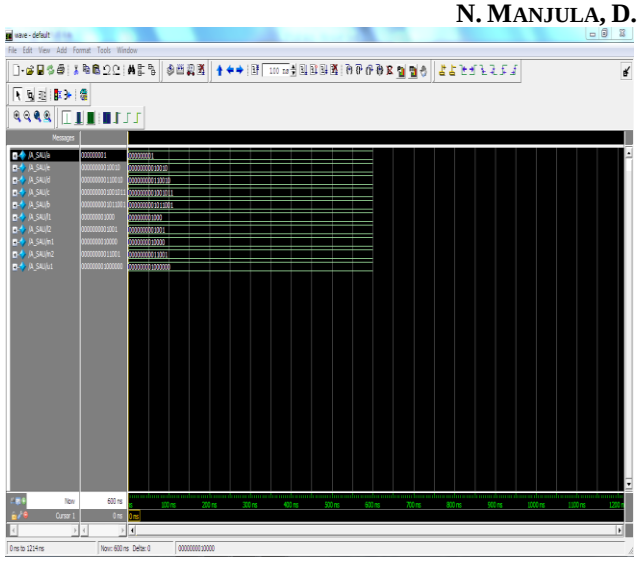

Fig4. SAU for 8 point.

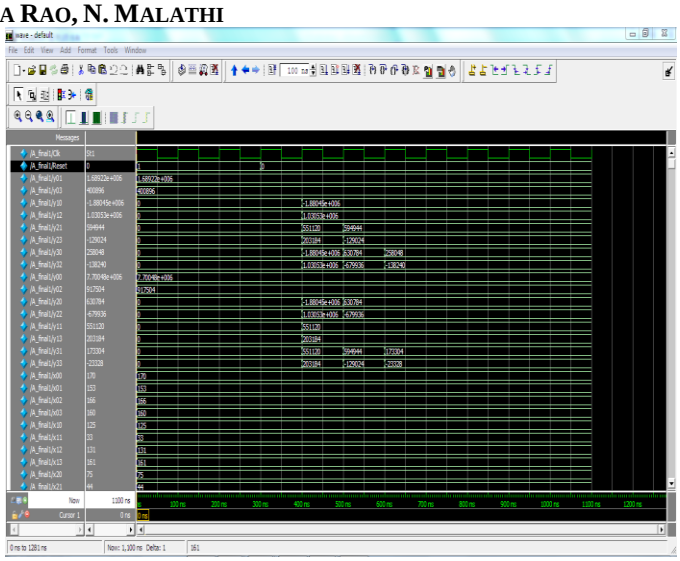

Fig7. Folded Structure

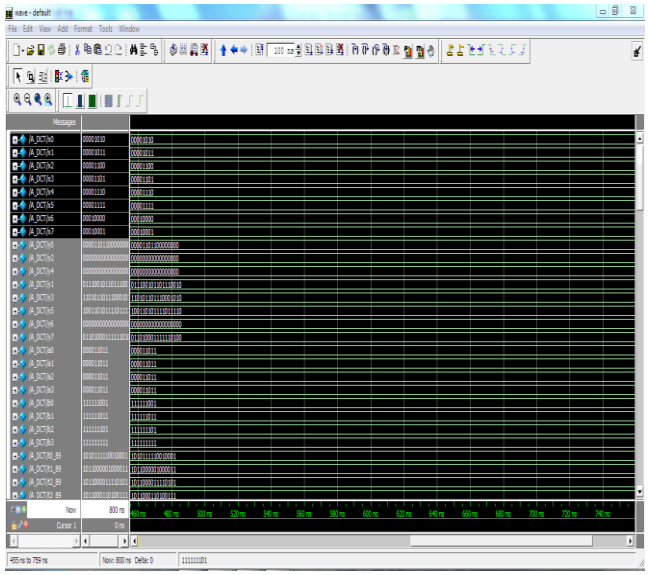

Fig5. Eight point DCT.

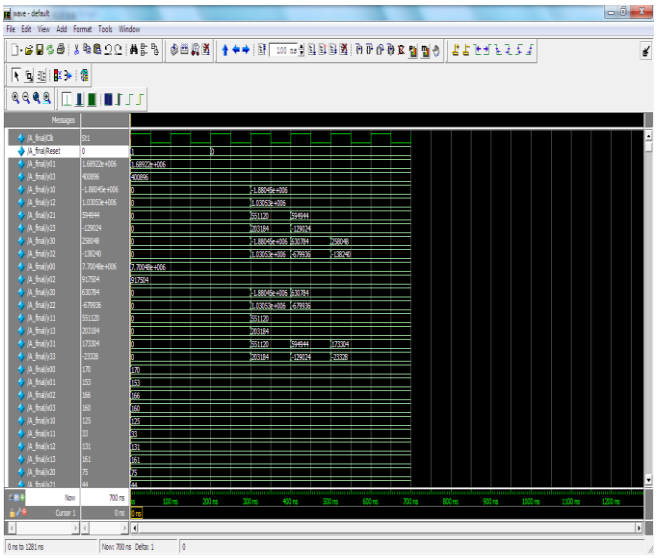

Fig8. Full Parallel.

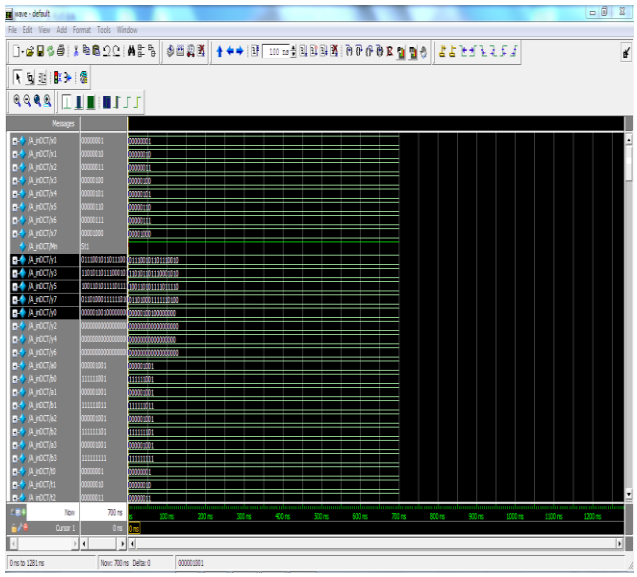

Fig6. Proposed reusable architecture for N = 8.

A. Device Utilization Summary

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	9	7,168	1%	
Number of 4 input LUTs	562	7,168	7%	
Logic Distribution				
Number of occupied Slices	380	3,584	10%	
Number of Slices containing only related logic	380	380	100%	
Number of Slices containing unrelated logic	0	380	0%	
Total Number of 4 input LUTs	609	7,168	8%	
Number used as logic	562			
Number used as a route-thru	47			
Number of bonded IOBs	370	141	262%	OVERMAPPED
Number of GCLKs	1	8	12%	
Total equivalent gate count for design	7,464			
Additional JTAG gate count for IOBs	17,760			

Fig9. Folded Structure.

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	9	7,168	1%	
Number of 4 input LUTs	510	7,168	7%	
Logic Distribution				
Number of occupied Slices	364	3,584	10%	
Number of Slices containing only related logic	364	364	100%	
Number of Slices containing unrelated logic	0	364	0%	
Total Number of 4 input LUTs	556	7,168	7%	
Number used as logic	510			
Number used as a route-thru	46			
Number of bonded IOBs	370	141	262%	OVERMAPPED
Number of GCLKs	1	8	12%	
Total equivalent gate count for design	6,999			
Additional JTAG gate count for IOBs	17,760			

Fig10. Full Parallel

Advantages:

- Reused for any of the prescribed lengths.
- The proposed structure could be reusable for DCT of lengths 4, 8, 16, and 32 with a throughput of 32 DCT coefficients per cycle irrespective of the transform size.
- Less-area delay due to Parallel implementation.
- The proposed architecture could be pruned to reduce the complexity of implementation substantially with only a marginal affect on the coding performance.

Applications:

- It is used in Mobile Multimedia Devices.
- The proposed architecture is found to support ultrahigh definition 7680 × 4320 at 60 frames/s video, which is one of the applications of HEVC.
- Signal & Image Processing.
- Digital Cameras.
- HDTV

Future Scope:

- The proposed system can modified by reducing the Area and delay of the design in future.
- The fast algorithm for the 8-point DCT 2D architecture designed by applying the 1D DCT structure in the folded and full parallel 2D DCT architecture.

V. CONCLUSION

Area- and Power-efficient architectures are used for the implementation of integer DCT of different lengths to be used in HEVC. The computation of N-point 1-D DCT involves an (N/2)-point 1-D DCT and a vector-matrix multiplication with a constant matrix of size (N/2) × (N/2). In this project MCM-based architecture is highly regular and involves significantly less area-delay complexity and less energy consumption than the direct implementation of matrix multiplication for odd DCT coefficients. Here for the proposed architecture to derive a reusable architecture for DCT that can compute the DCT of lengths 4, 8, 16, and 32 with throughput of 32 output coefficients per cycle.

VI. REFERENCES

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

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

[3]Y. Chen, S. Oraintara, and T. Nguyen, "Video compression using integer DCT," in Proc. IEEE Int. Conf. Image Process., Sep. 2000, pp. 844–845.

[4]J. Wu and Y. Li, "A new type of integer DCT transform radix and its rapid algorithm," in Proc. Int. Conf. Electric Inform. Control Eng., Apr. 2011, pp. 1063–1066.

[5]A. M. Ahmed and D. D. Day, "Comparison between the cosine and Hartley based naturalness preserving transforms for image watermarking and data hiding," in Proc. First Canad. Conf. Comput. Robot Vision, May 2004, pp. 247–251.

[6]M. N. Haggag, M. El-Sharkawy, and G. Fahmy, "Efficient fast multiplication-free integer transformation for the 2-D DCT H.265 standard," in Proc. Int. Conf. Image Process., Sep. 2010, pp. 3769–3772.

Author's Profile:

N.MANJULA has completed her B.Tech. in ECE, from Nalla Malla Reddy Engineering College, Narapally Ghatkesar, R.R dist, Telangana, JNTU Hyderabad India, Hyderabad. Presently she is pursuing her Masters in VLSI & EMBEDDED SYSTEMS, from Siddhartha Institute of Engineering and Technology, Ibrahimpatnam, Hyderabad, Telangana.
Email id:manjulanalavothula@gmail.com.

Mrs. N.MALATHI , a highly enlightened person honored as Master of Technology in VLSI Systems Design from TKR college of Engineering and Technology , Hyderabad, Btech in Sree Nidhi Institute of Engineerring and Technology ,presently working as Associate Professor, Department of ECE, SIET, Ibrahimpatnam. she has Life time membership of Indian Society for Technical Education (ISTE). Email id:n.malathiraj@gmail.com

Dr. D Subba Rao, is a proficient Ph.D person in the research area of Image Processing from Vel-Tech University, Chennai along with initial degrees of Bachelor of Technology in Electronics and Communication Engineering (ECE) from Dr. S G I E T, Markapur and Master of Technology in Embedded Systems from SRM University, Chennai. He has 13 years of teaching experience and has published 12 Papers in International Journals, 2 Papers in National Journals and has been noted under 4 International Conferences. He has a fellowship of The Institution of Electronics and Telecommunication Engineers (IETE) along

with a Life time membership of Indian Society for Technical Education (ISTE). He is currently bounded as an Associate Professor and is being chaired as Head of the Department for Electronics and Communication Engineering discipline at Siddhartha Institute of Engineering and Technology, Ibrahimpatnam, Hyderabad.
Email –Id: subbu.dasari@gmail.com.