IJETIE International Journal of Emerging Technology and Innovative Engineering
Volume I, Issue 4, April 2015
ISSN: 2394 - 6598
www.ijetie.org

ONE DIMENSION ARCHITECTURAL DESIGN FOR HIGH EFFICIENT VIDEO ENCODING USING INTEGER DCT

S.Ramya,
M.E. Applied Electronics,
PSN College of Engineering and Technology,
ramya632111@gmail.com

S.Rajeshwari,
AP/ECE,
PSN College of Engineering and Technology

Abstract

HEVC is said to double the data compression ratio compared to H.264/MPEG-4 AVC at the same level of video quality. It can alternatively be used to provide substantially improved video quality at the same bit rate. The design of most video coding standards is primarily aimed at having the highest coding efficiency. Coding efficiency is the ability to encode video at the lowest possible bit rate while maintaining a certain level of video quality. There are two standard ways to measure the coding efficiency of a video coding standard, which are to use an objective metric, such as peak signal-to-noise ratio (PSNR), or to use subjective assessment of video quality. Subjective assessment of video quality is considered to be the most important way to measure a video coding standard since humans perceive video quality subjectively. The Discrete Cosine Transform (DCT) plays a vital role in video compression due to its near-optimal decorrelation efficiency and high energy compaction. HEVC supports DCT of different sizes such as 4,8,16 and 32 and it is a loss compression technique. Integer DCT integrates both loss and lossless coding scheme perfectly. Integer DCT factors butterfly calculations of DCT. Two N/2-point DCT modules are used along with transposition buffers and 2:1 MUX to select either of input depending on whether for N-point DCT computation of for lower size. The proposed algorithm of Integer DCT implementation is flexible to support DCT of any length. It increases the throughput in video encoding. The parallel architecture for 1-D integer DCT of different length is implemented with a benefit of reusability.

Index Terms – HEVC,DCT, IDCT, Peak-Signal-to-Noise Ratio.

1. Introduction

The High Efficiency Video Coding (HEVC) was developed by the ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG), through their Joint Collaborative Team on Video Coding (JCT-VC). HEVC is also known as ISO/IEC 23008-2 MPEG-H Part 2 and ITU-T H.265. HEVC provides superior video quality and up to twice the data compression as the previous standard (H.264/MPEG-4 AVC). HEVC can support 8K Ultra High Definition video, with a picture size up to 8192x4320 pixels.

X265 can encode HEVC bit streams that are entirely lossless (the reconstructed images are bit-exact to the source images) by using the --lossless option. Lossless operation is theoretically simple. Rate control, by definition, is disabled and the encoder disables all quality metrics since they would only waste CPU cycles.

In HEVC, lossless coding means by passing both the DCT transforms and by passing quantization (often referred to as transquant bypass). Normal predictions are still allowed, so the encoder will find optimal inter or intra predictions and then lossless code the residual (with transquant bypass).

Not wanting to mess around with small, incremental improvements, whenever a new compression standard is introduced, it has to be a sizable change. With each jump, the general rule is half the bit rate for the same quality (or greater quality at the same bit rate).

Transform coding constitutes an integral component of contemporary image/video processing applications. Transform coding relies on the premise that pixels in an image exhibit a certain level of correlation with their neighbouring pixels. Similarly in a video transmission system, adjacent pixels in consecutive frames2 show very high correlation. Consequently, these correlations can be exploited to predict the value of a pixel from its respective neighbours. A transformation is, therefore, defined to map this spatial (correlated) data into transformed (uncorrelated) coefficients. Clearly, the transformation should utilize the fact that the information content of an individual pixel is relatively small i.e., to a large extent visual contribution of a pixel can be predicted using its neighbours.

The objective of the source encoder is to exploit the redundancies in image data to provide compression. In other words, the source encoder reduces the entropy, which in our case means decrease in the average number of bits required to represent the image. On the contrary, the channel encoder adds redundancy to the output of the source encoder in order to enhance the reliability of the transmission.

Each sub-block in the source encoder exploits some redundancy in the image data in order to achieve better compression. The transformation sub-block decor relates the image data thereby reducing (and in some cases eliminating) inter pixel redundancy. The transformation is a lossless operation; therefore, the inverse transformation renders a perfect reconstruction of the original image. The quantized sub-block utilizes the fact that the human eye is unable to perceive some visual information in an image. Such information is deemed redundant and can be discarded without introducing noticeable visual artifacts. Such redundancy is referred to as psycho visual redundancy. This idea can be extended to low bit rate receivers which, due to their stringent bandwidth requirements, might sacrifice visual quality in order to achieve bandwidth efficiency. This concept is the basis for rate distortion theory, that is, receivers might tolerate some visual distortion in exchange for bandwidth conservation. Lastly, the entropy encoder employs its knowledge of the transformation and quantization processes to reduce the number of bits required to represent each symbol at the quantize output. DCT has been widely deployed by modern video coding standards, for example, MPEG, JVT etc.

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. Some hardware architectures for the integer DCT for HEVC have also been proposed for its real-time implementation. The multiplierless 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 can also be used. Chen’s factorization of DCT can also be used where the butterfly operation has been implemented by the processing element with only shifters, adders, and multiplexors. unified structure to be used for forward as well as inversetransform after the matrix decomposition can also be used. 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, it has been 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.

The most common DCT definition of a 1-D sequence of length N is

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

The 2-D DCT is a direct extension of the 1-D case and is given by

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

The principle advantage of image transformation is the removal of redundancy between neighbouring pixels. This leads to uncorrelated transform coefficients which can be encoded independently. Clearly, the amplitude of the autocorrelation after the DCT operation is very small at all lags. Hence, it can be inferred that DCT exhibits excellent decorrelation properties.

Efficacy of a transformation scheme can be directly gauged by its ability to pack input data into as few coefficients as possible. This allows the quantizer to discard coefficients with relatively small amplitudes without introducing visual distortion in the reconstructed image. DCT exhibits excellent energy compaction for highly correlated images.

This property, known as separability, has the principle advantage that C(u, v) can be computed in two steps by successive 1-D operations on rows and columns of an image. The arguments presented can be identically applied for the inverse DCT computation.

2. Related Work

A method that uses Karhunen- Lo'eve transform as discrete cosine transform (DCT)[1] is defined and an algorithm to compute it using the fast Fourier transform is developed. It is shown that the discrete cosine transform can be used in the area of digital processing for the purposes of pattern recognition and Wiener filtering. Its performance is compared with that of a class of orthogonal transforms and is found to compare closely to that of the Karhunen-Lo'eve transform, which is known to be optimal. The performances of the Karhunen-Lo'eve and discrete cosine transforms are also found to compare closely with respect to the rate-distortion criterion.

Another method with the implementation of the integer discrete cosine transform (IntDCT) using the Walsh-Hadamard transform and the lifting scheme[2] is in the forms of shifts and adds, and all internal nodes have finite precision. A general-purpose scheme of 8-pt IntDCT with complexity of 45 adds and 18 shifts is proposed which gives comparable performance to the floating-point DCT

(FloatDCT). For this particular scheme with 8-bit input, perfect reconstruction (PR) is preserved even when all the internal nodes are limited to 16-bit words, rendering the Pentium MMX optimization possible. Implementation has been done to incorporate the proposed Int DCT into the H.263+ coder, and the resulting system performs equally well as the original. Further extension to the MPEG coder is straightforward. The proposed IntDCT is reversible, with a low level of power consumption, and is very suitable for source coding, and communication, etc. in a mobile environment.

Without changing the features of the integer DCT transform radix, improve on its matrix filling rules and implemented a new type of integer DCT transform radix[3]. Other than orthogonality, such a variety of integer DCT transform radix also meet the quantity relationships between elements of an integer as a matrix, and the normalized coefficients matrix have similar characteristic with the original DCT matrix. This type of transform radix can be implemented by rapid butterfly algorithm, in such fast algorithm implementation of integer DCT, no multiplication, just additions and shifts operations. This fast algorithm of integer DCT ensure the accuracy of transformation operations, reduce the complexity of the transformation operations.

A novel method is replaced for the real-numbered elements of a discrete cosine transform (DCT) matrix by integers and still maintain the structure. i.e, relative magnitudes and orthogonality, among the matrix elements. The result is an integer cosine transform (ICT)[4]. Thirteen ICT’s have been found and some of them have performance comparable to the DCT.

A 4 or 8-point IDCT are widely used in traditional video coding standards. However larger size (16/32-point) IDCT[5] has been proposed in the next generation video standard such as HEVC. To fulfill this requirement, this work proposes a fast computational algorithm of large size integer IDCT. A unified VLSI architecture for 4/8/16/32-point integer IDCT is also proposed accordingly. It can support the following video standards: MPEG-2/4, H.264, AVS, VC-1 and HEVC. Multiplier less MCM (Multiple Constant Multiplication) is used for 4/8-point IDCT. The transpose memory uses SRAM instead of the traditional register array in order to further reduce the hardware overhead.

A 2-D Large Inverse Transform (16x16, 32x32) for HEVC[6] is a 16x16 and 32x32 inverse transform architecture for HEVC (High Efficiency Video Coding) transforms 16x16 and 32x32 and it suffers from huge computational complexity. To resolve this problem, a new large inverse transform architecture based on hardware reuse is proposed. The processing element is optimized by exploiting fully recursive and regular butterfly structure. To achieve low area, the processing element is implemented by shifters and adders without multiplier. Implementation of the proposed 2-D inverse transform architecture in 0.18 m technology shows about 300 MHz frequency and 287 Kgates area, which can process 4K (3840x2160)@ 30 fps image.

Unified forward+inverse transform architecture for HEVC[7] is the upcoming HEVC video coding standard supports many transform sizes ranging from 4-point to 32-point in square and rectangular form. Multiple transform sizes improve coding efficiency, but also increase the implementation complexity. Furthermore both forward and inverse transforms need to be supported in various consumer devices. It achieves hardware sharing across different transform sizes and also between forward and inverse transforms. It uses 43-45% less area than separate forward and inverse core transform implementations.

3. Methodology

3.1. Architectural Design

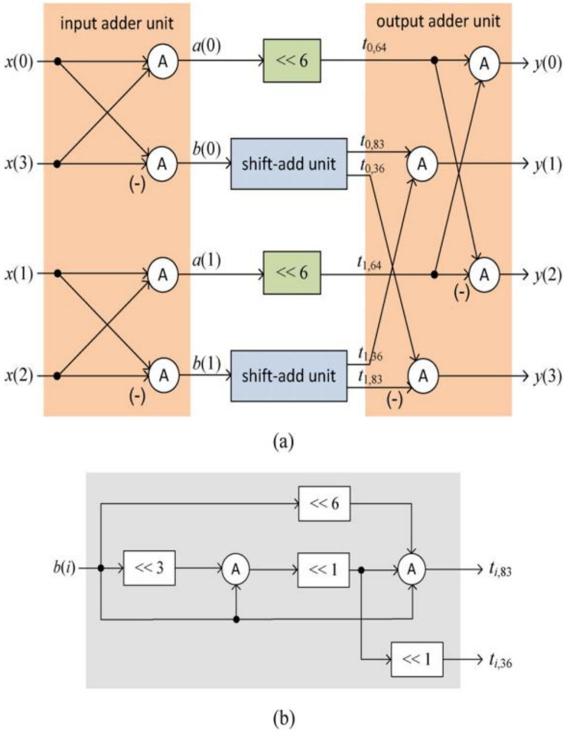


Fig. 1. Proposed architecture of four-point integer DCT. (a) Four-point DCT architecture. (b) Structure of SAU.

The proposed architecture for four-point integer DCT is shown in Fig. 1(a). 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 $t_{i,36}$ and $t_{i,83}$ are performed by two SAUs according to STAGE-2 of the algorithm. The computation of $t_{0,64}$ and $t_{1,64}$ does not consume any logic since the shift operations could be rewired in hardware. The structure of SAU is shown in Fig. 1(b). Outputs of the SAU are finally added by the OAU according to STAGE-3 of the algorithm.

Stage1	$a(i)=x(i) + x(3-i)$, $b(i) = x(i) - x(3-i)$, for i =0 and 1.
Stage2	$m_{i,9}=9b(i)-(b(i)<<3)+b(i)$, $t_{i,64}=64a(i)=a(i)<<6$, $t_{i,83}=83b(i)=b(i)<<6 + m_{i,9} <<1 + b(i)$ $t_{i,36}=36b(i)=m_{i,9} <<2$ for i =0 and 1.
Stage3	$y(0)=t_{0,64} + t_{1,64}$ $y(2)=t_{0,64} - t_{1,64}$ $y(1)=t_{0,83} + t_{1,36}$ $y(3)=t_{0,36} - t_{1,83}$

Table I: The Computation of 4-Point DCT

3.2. Generalized Architectural Design

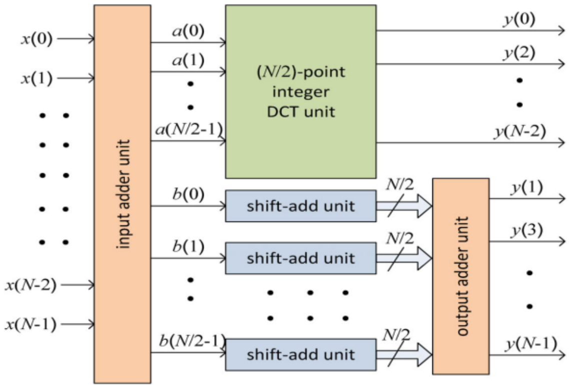


Fig 2: Architecture Design

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. 4 for 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 fourpoint DCT is the smallest DCT. The proposed structure can compute one 32-point DCT, two 16-point DCTs, four eightpoint DCTs, and eight four-point DCTs, while the throughput remains the same as 32 DCT coefficients per cycle irrespective of the desired transform size.

3.3. Discrete Cosine Transformation(DCT)

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. Some hardware architectures for the integer DCT for HEVC have also been proposed for its real-time implementation. The multiplierless 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 can also be used. Chen's factorization of DCT can also be used where the butterfly operation has been implemented by the processing element with only shifters, adders, and multiplexors. unified structure to be used for forward as well as inverse transform after the matrix decomposition can also be used. 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, it has been 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.

The most common DCT definition of a 1-D sequence of length N is

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

The 2-D DCT is a direct extension of the 1-D case and is given by

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

The principle advantage of image transformation is the removal of redundancy between neighbouring pixels. This leads to uncorrelated transform coefficients which can be encoded independently. Clearly, the amplitude of the autocorrelation after the DCT operation is very small at all lags. Hence, it can be inferred that DCT exhibits excellent decorrelation properties.

Efficacy of a transformation scheme can be directly gauged by its ability to pack input data into as few coefficients as possible. This allows the quantizer to discard coefficients with relatively small amplitudes without introducing visual distortion in the reconstructed image. DCT exhibits excellent energy compaction for highly correlated images.

This property, known as separability, has the principle advantage that C(u, v) can be computed in two steps by successive 1-D operations on rows and columns of an image. The arguments presented can be identically applied for the inverse DCT computation.

3.4. Structure of HEVC Encoder/Decoder

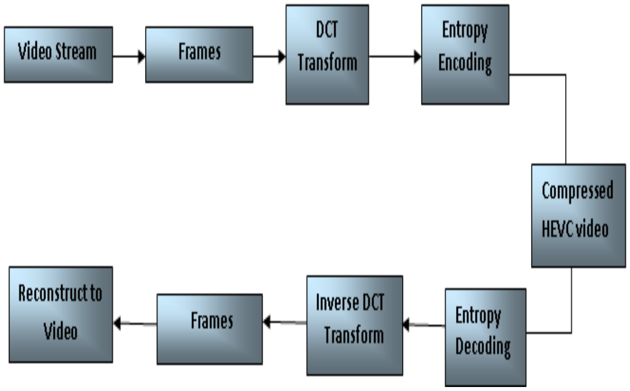


Fig 3: HEVC Encoder/Decoder

A typical hybrid video encoder, e.g., HEVC, include the encoding and decoding chain involves DCT, quantization, dequantization, and IDCT based on the transform design proposed. A data before and after the DCT/IDCT transform is constrained to have maximum 16 bits regardless of internal bit depth and DCT length. Therefore, scaling operation is required to retain the word length of intermediate data. In the main profile that supports only 8-bit samples, if bit truncations are not performed, the word length of the DCT output would be $\log_2 N + 6$ bits more than that of the input to avoid overflow. The output word lengths of the first and second forward transforms are scaled down to 16 bits by truncating least significant $\log_2 N - 1$ and log2 N + 6 bits, respectively. The resulting coefficients from the inverse transforms are also scaled down by the fixed scaling factor of 7 and 12. It should be noted that additional clipping of log2 N − 1 most significant bits is required to maintain 16 bits after the first inverse transform and subsequent scaling.

The scaling operation, however, could be integrated with the computation of the transform without significant impact on the coding result. The SAU includes several left-shift operations whereas the scaling process is equivalent to performing the right shift. Therefore, by manipulating the shift operations in the SAU circuit, we can optimize the complexity of the proposed DCT structure. Each row of dot diagram contains 17 dots, which represents output of the IAU or its shifted form (for 16 bits of the input wordlength). The final sum without truncation should be 25 bits. But, we use only 16 bits in the final sum, and the remaining 9 bits are finally discarded. To reduce the computational complexity, some of the least significant bits (LSB) in the SAU in the gray area can be pruned. It is noted that the worst-case error by the pruning scheme occurs when all the truncated bits are one. In this case, the sum of truncated values amounts to 88, but it is only 17% of the weight of LSB of the final output, 29. Therefore, the impact of the proposed pruning on the final output is not significant. However, as we prune more bits, the truncation error increases rapidly.

The output of the SAU is the product of DCT coefficient with the input values. In the HEVC transform design, 16 bit multipliers are used for all internal multiplications. To have the same precision, the output of SAU is also scaled down to 16 bits by truncating 4 more LSBs. The gray area before the OAU for the output addition corresponding to the computation of y(0). The LSB of the result of the OAU is truncated again to retain 16 bits transform output. By careful pruning, the complexity of SAU and OAU can be significantly reduced without significant impact on the error performance. In the case of the second forward eight-point DCT (in the seventh. the number of input word length is 16 bits. If any truncation is not performed in the processing of IAU, SAU, and OAU, the number of output word length becomes 25 bits by summation of values in columns A, B, and C. By the truncation procedure illustrated in Fig. 8 using the number of truncated bits listed in columns D, E, and F. The word length of DCT output becomes 16 bits.

4. Results and Discussions

The processing rate of the proposed integer DCT unit is 16 pixels per cycle considering 2-D folded structure since 2-D transform of 32 × 32 block can be obtained in 64 cycles. In order to support 8K ultrahigh definition (UHD) (7680 × 4320) at 30 frames/s and 4:2:0 YUV format that is one of the applications of HEVC, the proposed reusable architectures should work at the operating frequency faster than 94 MHz (7680 × 4320 × 30 × 1.5/16). The computation times of 5.56 ns is for reusable architectures-1.

The HEVC codec transformation is designed to have the following properties:

- Closeness to the IDCT
- Almost orthogonal basis vectors
- Almost equal norm of all basis vectors
- Same symmetry properties as the IDCT basis vectors
- Smaller transform matrices are embedded in larger transform matrices
- 8-bit representation of transform matrix elements
- 16-bit transpose buffer
- Multipliers can be represented using 16 bits or less with no cascaded multiplications or intermediate rounding.
- Accumulators can be implemented using less than 32 bits.

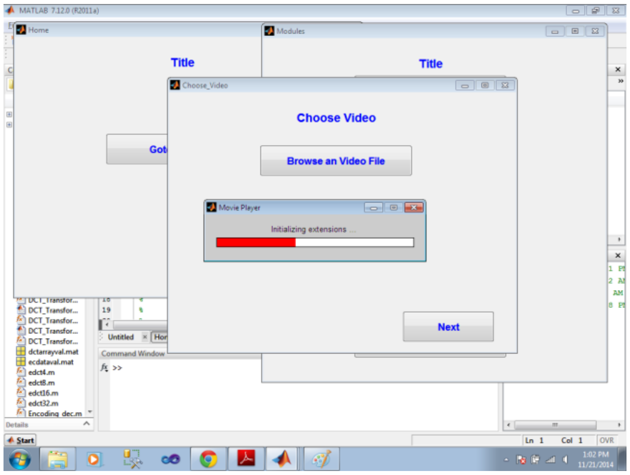

Initializing extensions ...

Fig 4: Input Video

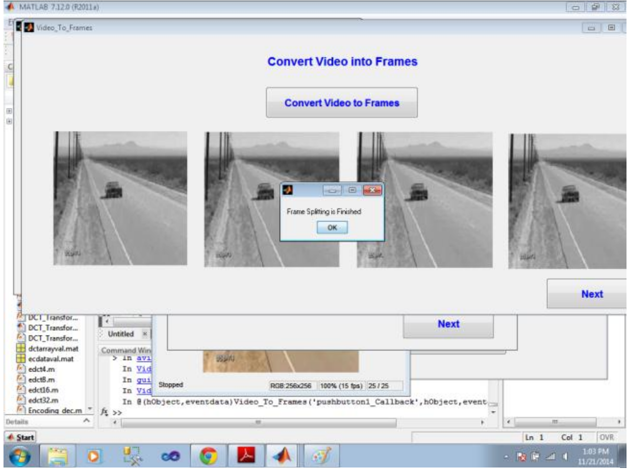

Frame Splitting is Finished

Fig 5: Video to Frames

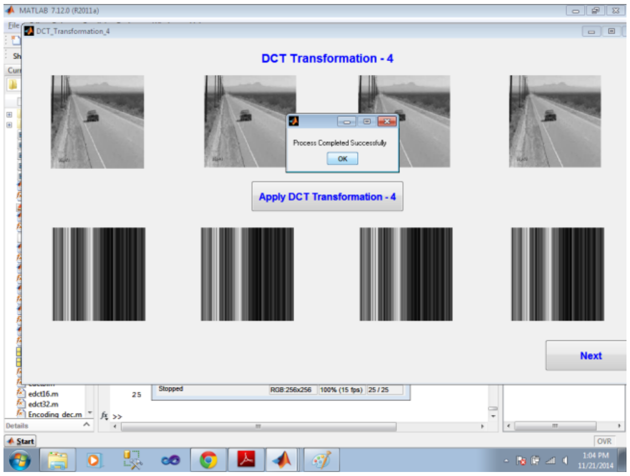
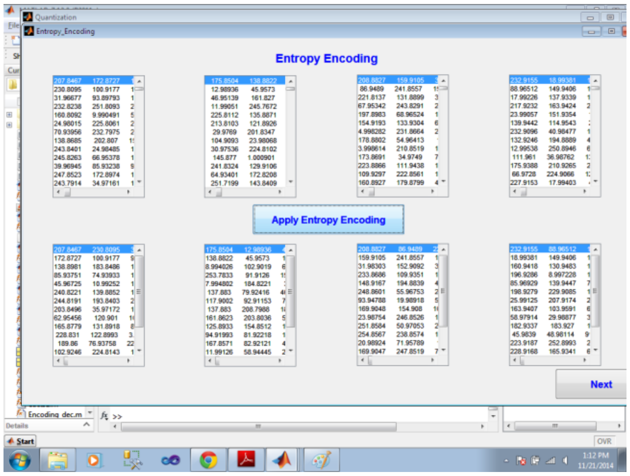

Process Completed Successfully

Fig 6: DCT Transformation – 4 Point



```
207.8467    172.8727
230.8095    100.9177
31.96677    93.89793
232.8238    251.8093
160.8092    9.990491
24.98015    225.8061
70.93956    232.7975
138.8685    202.807
243.8401    24.98485
245.8263    66.95378
39.96945    85.93238
247.8523    172.8974
243.7914    34.97161
```

```
175.8504    138.8822
12.98936    45.9573
46.95139    161.827
11.99051    245.7672
225.8112    135.8871
213.8103    121.8926
29.9769     201.8347
104.9093    23.98068
30.97536    224.8102
145.877     1.000901
241.8324    129.9106
64.93401    172.8208
251.7199    143.8409
```

```
208.8827    159.9105
86.9489     241.8557
221.8137    131.8899
67.95342    243.8291
197.8983    68.96524
154.9193    133.9304
4.998282    231.8664
178.8802    54.96413
3.998614    210.8519
173.8691    34.9749
223.8866    111.9438
109.9297    222.8561
160.8927    179.8799
```

```
232.9155    18.99381
88.96512    149.9406
17.99226    137.9339
217.9232    163.9424
23.99057    151.9354
139.9442    114.9543
232.9096    40.98477
132.9246    194.8889
12.99538    250.8946
111.961     36.98762
175.9388    210.9265
66.9728     224.9066
227.9153    17.99403
```

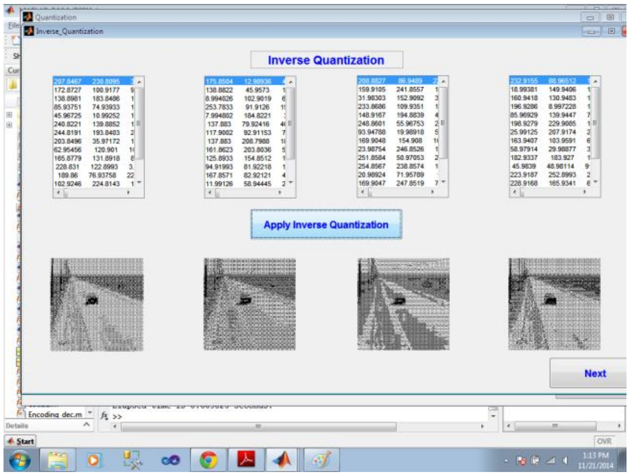
```
207.8467    230.8095
172.8727    100.9177
138.8981    183.8486
85.93751    74.93933
45.96725    10.99252
240.8221    139.8852
244.8191    193.8403
203.8496    35.97172
62.95456    120.901
165.8779    131.8918
228.831     122.8993
189.86      76.93758
102.9246    224.8143
```

```
175.8504    12.98936
138.8822    45.9573
8.994026    102.9019
253.7833    91.9126
7.994802    184.8221
137.883     79.92416
117.9002    92.91153
137.883     208.7988
161.8623    203.8036
125.8933    154.8512
94.91993    81.92218
167.8571    82.92121
11.99126    58.94445
```

```
208.8827    86.9489
159.9105    241.8557
31.98303    152.9092
233.8686    109.9351
148.9167    194.8839
248.8601    55.96753
93.94788    19.98918
169.9048    154.908
23.98754    246.8526
251.8584    50.97053
254.8567    238.8574
20.98924    71.95789
169.9047    247.8519
```

```
232.9155    88.96512
18.99381    149.9406
160.9418    130.9483
196.9286    8.997228
85.96929    139.9447
198.9279    229.9085
25.99125    207.9174
163.9407    103.9591
58.97914    29.98877
182.9337    183.927
45.9839     48.98114
223.9187    252.8993
228.9168    165.9341
```

Fig 7: Entropy Encoding



```
207.8467    230.8095
172.8727    100.9177
138.8981    183.8486
85.93751    74.93933
45.96725    10.99252
240.8221    139.8852
244.8191    193.8403
203.8496    35.97172
62.95456    120.901
165.8779    131.8918
228.831     122.8993
189.86      76.93758
102.9246    224.8143
```

```
175.8504    12.98936
138.8822    45.9573
8.994026    102.9019
253.7833    91.9126
7.994802    184.8221
137.883     79.92416
117.9002    92.91153
137.883     208.7988
161.8623    203.8036
125.8933    154.8512
94.91993    81.92218
167.8571    82.92121
11.99126    58.94445
```

```
208.8827    86.9489
159.9105    241.8557
31.98303    152.9092
233.8686    109.9351
148.9167    194.8839
248.8601    55.96753
93.94788    19.98918
169.9048    154.908
23.98754    246.8526
251.8584    50.97053
254.8567    238.8574
20.98924    71.95789
169.9047    247.8519
```

```
232.9155    88.96512
18.99381    149.9406
160.9418    130.9483
196.9286    8.997228
85.96929    139.9447
198.9279    229.9085
25.99125    207.9174
163.9407    103.9591
58.97914    29.98877
182.9337    183.927
45.9839     48.98114
223.9187    252.8993
228.9168    165.9341
```

Fig 8: Inverse Quantization

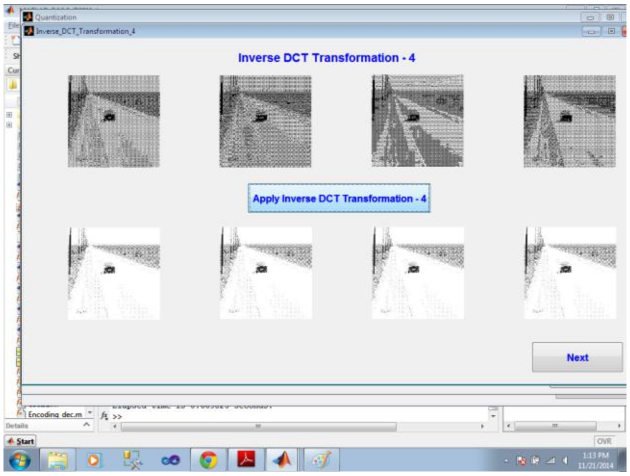
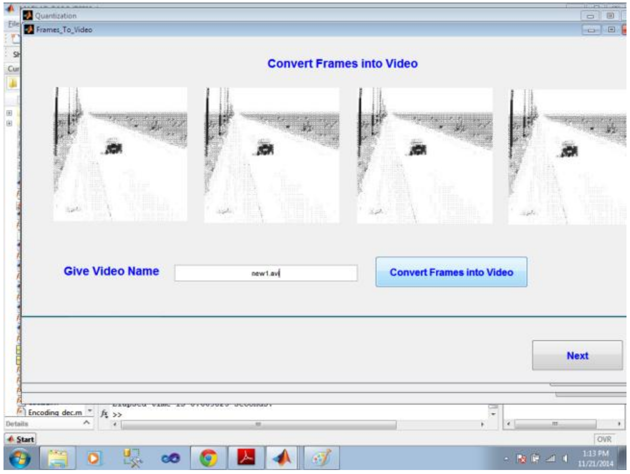

Quantization
Inverse_DCT_Transformation_4
Inverse DCT Transformation - 4
Apply Inverse DCT Transformation - 4
Next
Encoding_dec.m
Start
OVR
1:13 PM
11/21/2014

Fig 9: Inverse DCT Transformation – 4 Point



Give Video Name: new1.avi

Fig 10: Frames to Video

5. Conclusion

The area- and power-efficient architectures for the implementation of integer DCT of different lengths to be used in HEVC is developed. The computation of N-point 1-D DCT involves an (N/2)-point 1-D DCT and a vector-matrix. The design is 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. It is also shown that proposed design could be pruned to reduce the area and power complexity of forward DCT without significant loss of quality, and multiplication with a constant matrix of size (N/2) × (N/2).

In future it can be proposed power-efficient architectures for folded and full-parallel implementations of 2-D DCT, where no data movement takes place within the transposition buffer.

6. References

1. K. R. Rao and P. Yip *Discrete Cosine Transform, Algorithms, Advantages, Applications*, 1990 :Academic Press

2. W. B. Pennebaker and J. L. Mitchell *JPEG Still Image Data Compression Standard*, 1993 :Van Nostrand Reinhold
3. J. L. Mitchell , W. B. Pennebaker and C. E. Fogg *Mpeg Video: Compression Standard (Digital Multimedia Standards Series)*, 1996 :Chapman &amp; Hall
4. TIAN Xiaohua, ZHU Guangxi, WANG Yao, JIN Xin, "Optimization Design of Integer Transform In Video Coding," Computer Applications, vol.24, pp.85-87, May 2004. (in Chinese)
5. DAI Shengkui, LIU Jianguo, WANG Guoyou, ZHU Pingping, "Fast Integer-DCT Implement without Multiplication," MICROELECTRONICS& COMPUTER, vol.25, pp.11-13, May 2008. (in Chinese).
6. Rui Su, Guizhong Liu, Tongyu Zhang, "Fast mode decision algorithm for intra prediction in H.264/AVC with integer transform and adaptive threshold," Signal, Image and Video processing, vol.1, pp.11-27, January 2007.
7. Jun Xin, Anthony Vetro, Huifang Sun, "Coverting DCT Coefficients to H.264/AVC Transform Coefficients," Lecture Notes in computer Science, vol.3332, pp.939-946, 2005.
8. Reeba Korah, J-Raja Paul Perinbam, "FPGA Implementation of Integer Transform and Quantizer for H.264 Encoder," Journal of signal processing systems, vol.53, pp.261-269, March 2008.
9. Bi Houjie, New generation video compression standard-H.264/AVC, POST &TELECOM PRESS, 2005. (in chinese)
10. WANG Zhongyuan, HU Ruimin, TIAN Gang, LI Ming, "The Generic Generating Algorithm for Integer DCT Transform Radix," Journal of Image and Graphics, vol.13, pp.1061-1064, June 2008. (in chinese)
11. R.G. Schyndel, A. Tirkel, and C. Osborne, "A digital watermark," in IEEE Proc. Int. Conf. Image Processing (ICIP), 1994, pp. 86-90.
12. R.B. Wolfgang and E.J. Delp, "A watermark for digital images," in IEEE Proc. Int. Conf. Image Processing, Lausanne, Switzerland, Sept. 1996, vol. 3, pp. 219-222.
13. W.N. Lie, G.S. Lin, C. L. Wu, and T. C. Wang, "Robust image watermarking on the DCT domain," in IEEE Proc. Int. Symposium, Circuits and Systems (ISCAS), Geneva, Switzerland, May 2000, pp. 228-231.
14. A. Lumini and D. Maio, "A wavelet-based image watermarking Scheme," in information technology proc. int. Conf. on Coding and Computing, March 2000, pp. 122-127.
15. Joint Collaborative Team on Video Coding (JCT-VC), "WD4: Working Draft 4 of High-Efficiency Video coding", JCTVC-F803.doc. s