Design of High Efficient Video Encoder H.265

Akash.H.Deepak, Vijaya Prakash A.M
MTech in VLSI and Embedded System, BIT, Bangalore
Professor, Dept of ECE, BIT, Bangalore

***Abstract*: In this project, we have designed an efficient Motion Estimation (ME) processor or engine for High Efficiency Video Coding (HEVC) systems is presented, along with its algorithm and VLSI architecture. The Exhaustive Search Algorithm that is implemented in this project, in particular, dramatically reduces the number of search possibilities while customizing the search space to the characteristics of the video. According to the experimental findings, this algorithm reduces computational complexity by 54% compared to traditional methods with just a slight performance hit of 2.01%. Additionally, this article presents the suggested ME engine's VLSI design and circuit implementation. Illustrated are the system level and gate level optimizations for increasing efficiency and lowering complexity.**

***Keywords*:H.265 Encoder, Verilog, FPGA, Xilinx,Genus**

I. INTRODUCTION

Modern multimedia systems must have an effective video compression, especially when using high-resolution digital video formats. In particular, the adjustment between reducing effectiveness and signal quality is represented by Peak Signal to Noise ratio (PSNR), is the heart of video suppression. The Motion Estimation in HEVC is essential because it eliminates superfluous information by forecasting how the current frame will move in relation to the reference frame.

Motion Estimation generally specifies a range for a particular frame and within this particular frame it identifies the Motion Vectors which specifies the movement of a particular for both the main frame and the reference Frame. A frequency transform is the term used to describe the discrete cosine transform (DCT). It was introduced for Wiener filtering and pattern recognition.

As a transform coder for still photos, moving images, and video compression, it is currently widely utilized. Typically, a DCT expresses a finite set of sequences as values of the cosine transform. As it turns out (mentioned below) that lesser cosine functions are required to simulate a particular data, the usage of conical functions rather than sinusoidal functions is essential for reduction, but the designs based for nonlinear equations represent a particular choice of boundary conditions. In particular, a DCT is a Fourier-related transform that uses only real numbers and is comparable with discrete Fourier transform (DFT). While DFT's are associated to Fourier Series products of only sporadically expanded sequences, DCT's only related to the Fourier Series products of a sporadic and for symmetrically expanded data.

Initiated as a pattern matching scheme, quantization or vector quantization is now widely utilized for data compression in voice, image, video coding, and speech recognition. A video encoder, which is used to compress the digital image, is frequently known as a hardware or firmware component. Video encoders commonly accept High-Definition Multimedia Interface (HDMI) or Serial Digital Interface (SDI) for television broadcasting (HDMI). These are typically the video and audio standards used for coaxial or fibre optic cable transmission. JPEG, MPEG, H.261, H.262, H.263 and H.264 are the previous standards for HEVC, respectively.

The remainder of this study continues with part II, which examines building a digital system by considering optimization of design parameters. Our proposed methodology is found in Section III. The simulation and results are presented in Section IV. In sections V and VI of this article, we draw attention to the conclusions and future scope of H.265 Encoder.

II. EXISTING SYSTEM FOR H.265 CORE BLOCKS

A. Discrete Cosine Transform

The transform domain, essentially aggregates conical equations vibrating at different wavelengths, is

utilized to represent a finite stream of data points. It has been used by most electronic content, including speech coding (like Augmentative and alternative communication, Calypso, and Augustus), digital radio (such as AAC+ and DAB+), digital television (like SDTV, HDTV, and VOD), and image files (such as JPEG and HEIF, where minor high-frequency components can be deleted).

Datasets also are critical for a wide spectrum of many other applications in science and engineering including such digital signal processing, telecommunications equipment, dramatically reducing network bandwidth, and spectral methodologies for the numerical solution of partial differential equations.

Trigonometric factors are vital for reduction since sinusoidal values indicate a limited number of boundary conditions for differential equations, but it turns out (as will be shown below) that fewer cosine functions are necessary to approximate a typical signal than sinusoidal functions.

In particular, a DCT is a Fourier-related transform that is comparable to the discrete Fourier transform and involves only real numbers (DFT). DCTs frequently relate to Fourier Series coefficients of a continuously and symmetrically expanded sequence, while DFTs only relate to Fourier Series coefficients of periodically extended patterns.

DCTs operate on real data with even symmetry and are normally twice as long as DFTs since the Fourier analysis of a real and even function is real and even.

Conversely, in certain modifications, the entry and/or data collected are displaced by twice the average a frame. There are eight common DCT variants, of whose four are conventional.

B. Quantization

In image processing, quantization is a lossy compression approach which compresses a wide range of values together into single quantum value. When there are reduced number of discrete symbols in a stream, it becomes even more compressible. For instance, a digital image's file size can be decreased by lowering the number of colors needed to represent it. Both color quantization and frequency quantization are aspects of quantization. Color quantization reduces the amount of shades employed in an image, enabling some types of images to be successfully reduced and exhibited on monitors which can only display a restricted variety of shades. Current color quantization approaches include several well-known examples among which are the adjacent colored technique (for static panel) the midpoint reduction approach, and an octree-based technique. The human eye is capable of detecting tiny improvements in brightness across a wide radius with maybe some accuracy, but it is difficult to quantify the precise magnitude of a high frequency (rapidly varying) luminance variability. The amount of information necessary gets diminished through this concept's potential exempt high frequency components. Merely divide each frequency domain component by the required constant to accomplish this, then round the result to the nearest integer. It is the major lossy activity inside the operation. The majority of the remaining components are indeed translated into little positive or negative integers, and many of the higher spectral analysis are typically adjusted to zero. In order for a video codec to function, the image must be divided into distinct blocks (8X8 pixels in the case of H.265). The frequency components can then be determined for these blocks using the discrete cosine transform (DCT), both either longitudinally and diagonally.

This normalization vector subsequently splits every generated frame, which is the same size as the initial block, into its constituent elements, rounding each subsequent factor.

C. Inter Prediction

A frame in a video compression stream that is represented in terms of one or more adjacent frames is known as an inter frame. The term's "inter" component alludes to the application of Inter frame prediction. This type of prediction seeks to benefit from the temporal overlap between nearby frames, which allows for higher compression rates. An inter coded frame can really be partitioned into segments utilizing macroblocks. An inter coded frame can really be partitioned into segments utilizing macroblocks. The encoder will then attempt to locate a block that is equivalent. Instead of specifically recording the underlying data point for every frame, the encoder overlays the frame to also be encoded over a reference frame that has previously been encoded. Block matching is the implementation method used in this case. If the encoder finds the block after a successful search, it may encode the block with

a motion vector, or vector, indicating the location of the matching block in the reference frame.

Even though the block retrieved is probably not an exact match to the block being encoded, the encoder will typically be successful. The encoder will detect their variations as a result. Before being sent to the decoder, these residual values, generally termed as the prediction error, must be altered. To summarize, the encoder will simultaneously receive a prediction error and a motion vector pointing at the matched block if it is capable of finding a matching block on a reference frame. Using both components, the decoder will be able to have the block's unfiltered pixels.

D. Motion Estimation

Motion estimate has been the procedure for figuring out the motion vectors which characterize overall change from same video frame to the next. It is frequently done using neighboring frames in a video series. Since the movement occurs in three dimensions but the visuals are a projection of the image into two dimensions, the issue is inadequately defined. The motion vectors could pertain to each individual pixel, each rectangular block, each randomly selected area, or even to the entire image (global motion estimate). A linear version or a number of alternative models, such as those that rotate, translate, and zoom in all three dimensions, can represent and estimate the motion vectors.

The indicated 16-bit limit is specified by high efficient video coding with both the azimuth and elevation original images. Together with the mvLX characteristics, this was presented forth during the HEVC meeting in July 2012.The HEVC horizontal/vertical MV range of 32768 to 32767 generates an MV range of 8192 to 8191.75 luma samples due to the quarter pixel resolution used by HEVC. Maximum MV ranges supported by H.264/MPEG-4 AVC, in contrast, include 2048 to 2047.75 luma samples for the horizontal MV range and 512 to 511.75 luma samples again for vertical MV range.

These main normalization mechanisms represented by High Efficient Video Coding are merger mode and advanced Motion Vector Prediction (AMVP). AMVP may incorporate information from surrounding prediction blocks as well as information from the reference picture. When employing the merging technique, the normalized values could be inherited from surrounding prediction blocks.

With multiple enhancements, HEVC's merge mode is comparable to the "skipped" and "direct" motion inference modes of H.264/MPEG-4 AVC. The first improvement is that HEVC chooses one choice from a variety of plausible options by using index data. The utilization of data from the reference image index and reference picture list by HEVC represents the main improvement.

III.PROPOSED METHODOLOGY FOR H.265 ENCODER

The H.265 Encoder's suggested design architecture will be built for an 8x8 block size and can be upgraded to a 64x64 block size. The proposed architecture can be implemented on Artix-7 series and higher for an FPGA-based design. The implementation of the sampling block makes use of a butterfly structure. The implementation of the Quantization block takes into account either a 50% or a 90% image matrix. By creating a Controller block, the Inter Prediction block is put into practise. We may run both the Motion Estimation and Inter Prediction processes since the inter prediction block will include both IME and FME. Using the CABAC procedure, the prediction's output will be encoded.

The indicated sampling procedure will be carried out with the aid of DCT. A butterfly-shaped implementation of the DCT will be used. The three stages of the butterfly structure are as follows. The DCT transformation matrix and the 8x8 image matrix are combined to provide the inputs that will be used initially. The second stage will include partial products that are added to the values of the other partial products. The output, which consists of the sum of the partial products, will be the last stage.

DCT will be used to carry out the sampling process as previously indicated. A butterfly structure will be used to implement the DCT. Three stages will make up the butterfly structure. In the beginning, we will give the inputs that are created by multiplying the DCT transformation matrix by the 8x8 picture matrix. The second stage will be made up of partial products that will be combined with the values from the other partial products. The output, which is the culmination of the incomplete products, will be the last stage. For the H.265 Encoder, quantization is regarded as a highly outstanding and beneficial feature. We shall

take into consideration the DCT outputs in this stage. The outputs produced will have varied degrees of image compression, and the quality is achieved by choosing particular quantization levels. This gives the user the option to select quality levels between a minimum of 1 and a maximum of 100. Here, 1 stands for the lowest quality image with the highest level of compression, while 100 represents the highest quality image with the lowest level of compression. Both the IME and the FME approaches are used in inter prediction. The data image values of the IME block will range from 8x8 to 64x64. In this particular project, an 8x8 image will be taken into consideration, and IME and FME will both be applied. The Full Search method will be used to implement the IME and FME techniques involved. Each 8x8 macro block partition will be partitioned into a 4x4 image sub-macroblock when we consider an 8x8 picture. Regarding this, we can see that there are 41 motion vectors present at the same time. After interpolating half and quarter pixels from reference frames, the FME projects fractional components. The MVs and their distortions are developed to determine the mode for each MB. The MB's mode, reference indexes, and MVs are transferred to the MC and subsequent units to encapsulate the encoding information. The MC obtains information from the motion estimate modules, such as data on the chosen mode, reference indices, and MVs, to re-build the projected MB. With this data, MC develops predictions about the current MB using values from reference frames. The residual values are also determined for the encoding process that follows. The IME makes use of the full-search algorithm. This method offers the highest level of motion estimate accuracy, although it is expensive to compute. Furthermore, the full-search algorithm of the H.264/AVC standard additionally offers concurrent variable block-size motion estimates for all modes. The overlapping rule of two neighboring search windows, that aligns the search windows with the current MB, saves at least 66 percent of the data bandwidth when reading data from external RAM with a search range of 48 X48.

Since mode determination is already incorporated into IME, the FME merely needs to use fractional components to refine the motion vector. In order to be used again as anticipated values, the interpolated pixels are kept. As a result, after the FME process is complete, we incorporate the MC of the luma component. Thus, we maximize the memory capacity and lower the delay of regenerating sub-pixels. Because the luma compensation is projected inside of FME, the chroma motion compensation solely reconstructs the chroma components from search windows. The Macroblock Mode is transferred to the CABAC Coding block after being packed identical to the standard but with a different motion vector.

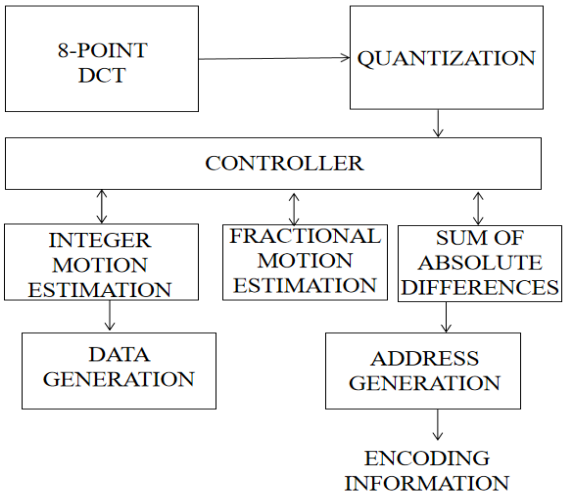


Figure 1: Design of Proposed H.265 Encoder

A. Full Search Algorithm

For the purpose of video processing, similar macroblocks can be found in a variety range digital image sequences using a thorough search method. It is assumed that the patterns associated with the foreground and objects in a frame of a video series travel inside the frame in order to construct the relevant objects on the following frame. This can be done by defining the contents of a macroblock by reference to the contents of a known macroblock that is minimally different in order to improve the performance of inter-frame video compression. This can be used to detect temporal duplication in the video sequence.

Using a Full Search approach, which divides the current frame of the video into macroblocks, each macroblock is compared to a selected cell and its associates in such a prior frame of the movie (sometimes just the previous one). A vector models the movement of a macroblock from one location to another. This movement, which is determined for each of the frame's macroblocks, is the motion that is estimated in a frame.

The search region for a successful macroblock match

is determined by the "lookup factor," p, which is the total number of pixels on all four sides of the pertinent macro-block within reference frame. The search parameter is a measure of motion. As the p value rises, so do the potential motion and chances of finding a good match. The exhaustive search of all feasible blocks, however, is a computationally expensive procedure.

By limiting the variations between reference and candidate block matching, the full search (FS) technique for video coding, which is based on block matching, discovers the optimum motion vectors. Due to its straightforward and straightforward hardware implementation, the FS algorithm has been frequently employed in video coding applications. However, a significant obstacle to achieving quick real-time video coding has been thought to be the FS algorithm's high computational cost due to its extremely broad search area.

By creating a specialized processor with an address generator, data generator, storage element, and memory controller, the Inter Prediction and Motion Estimation processes are concurrently conducted.

IV.RESULTS

The proposed architecture for H.265 Encoder simulation outputs are produced using vivado 2018.2 platform, and its synthesisusing the FPGA Artix-7 6 series results in reports on device utilization and timing summaries.

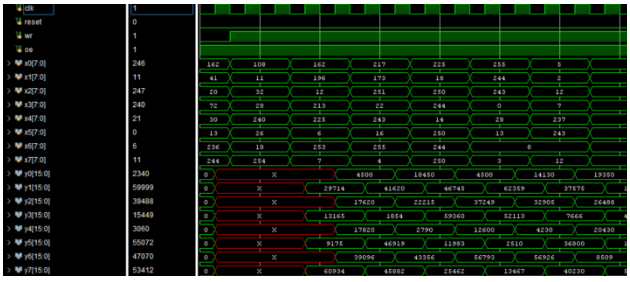


Figure 2: Simulation Output for DCT

Following waveforms show the simulations of designed DCTs that were implemented using Butterfly Structure and adders and multipliers. The following pictures show the simulation outputs for an approximate 8X8 bit pixel image or image transformation matrix.

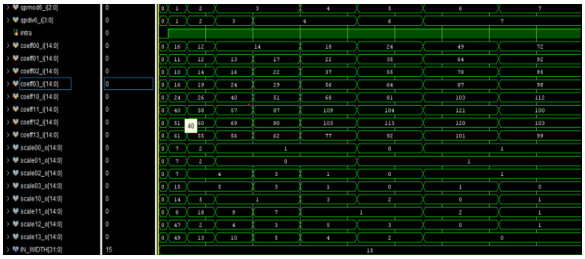


Figure 3: Simulation Output for Quantization

The following waveforms show the simulations of the designed quantization. The approximate 8X8 bit values from the simulation outputs are retrieved from the DCT output, accordingly.

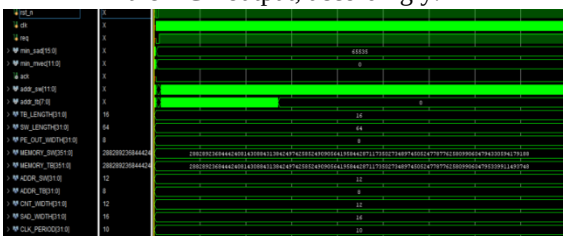


Figure 4: Output for Inter Prediction

The following waveforms show the results of designed Quantization simulations. The data from the reference frame and the approximate 8 bit values that are acquired from the simulation outputs in relation to each other.

```
Timing Report

Slack:                    inf
  Source:                 M2C1_reg[0]/C
                            (rising edge-triggered cell FDRE)
  Destination:            X3_reg[14]/D
  Path Group:             (none)
  Path Type:              Max at Slow Process Corner
  Data Path Delay:        6.238ns  (logic 3.660ns (58.673%)  route 2.578ns (41.327%))
  Logic Levels:           9  (CARRY4=5 FDRE=1 LUT1=1 LUT2=1 LUT3=1)

  Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                       FDRE                         0.000     0.000 r  M2C1_reg[0]/C
                       FDRE (Prop_fdre_C_Q)         0.478     0.478 f  M2C1_reg[0]/Q
                       net (fo=4, unplaced)         0.335     0.813    M2C1_reg_n_0_[0]
                       LUT1 (Prop_lut1_I0_O)        0.295     1.108 r  X5[4]_i_7/O
                       net (fo=2, unplaced)         0.344     1.452    X5[4]_i_7_n_0
                       CARRY4 (Prop_carry4_CYINIT_CO[3])
                                                    0.595     2.047 r  X3_reg[1]_i_11/CO[3]
                       net (fo=1, unplaced)         0.009     2.056    X3_reg[1]_i_11_n_0
                       CARRY4 (Prop_carry4_CI_O[1])
                                                    0.337     2.393 r  X3_reg[12]_i_20/O[1]
                       net (fo=2, unplaced)         0.622     3.015    X3_reg[12]_i_20_n_6
                       LUT3 (Prop_lut3_I0_O)        0.306     3.321 r  X3[12]_i_14/O
                       net (fo=2, unplaced)         0.650     3.971    X3[12]_i_14_n_0
                       CARRY4 (Prop_carry4_DI[1]_O[3])
                                                    0.629     4.600 r  X3_reg[12]_i_11/O[3]
                       net (fo=1, unplaced)         0.618     5.218    PCIN[12]
                       LUT2 (Prop_lut2_I1_O)        0.307     5.525 r  X3[12]_i_3/O
                       net (fo=1, unplaced)         0.000     5.525    X3[12]_i_3_n_0
                       CARRY4 (Prop_carry4_S[3]_CO[3])
                                                    0.376     5.901 r  X3_reg[12]_i_1/CO[3]
                       net (fo=1, unplaced)         0.000     5.901    X3_reg[12]_i_1_n_0
                       CARRY4 (Prop_carry4_CI_O[1])
                                                    0.337     6.238 r  X3_reg[15]_i_1/O[1]
                       net (fo=1, unplaced)         0.000     6.238    X30[14]
                       FDRE                                       r  X3_reg[14]/D
```

Figure 5: Timing Report for DCT

The procedure is faster overall because to the proposed butterfly structure implementation for DCT, which acts on the number of bits given in the design and produces partial products. The design

computation can be finished more quickly because to the usage of adders and multipliers in this model. Overall, it improves the DCT's performance.

A synthesis report is produced by Xilinx Vivado upon the completion of the synthesis procedure, and it contains all the accurate and detailed data required for the timing delay evaluation. 6.238 ns (3.660 ns logic delay and 2.578 ns routing delay) is the timing period analysis summary for an eight point DCT operating at a frequency of 160.308 MHz, as illustrated in the figure 8 above. The delay values are around 58.673 percent and 41.327 percent, respectively, when calculating the percentage occupation of the logic and the routing interconnections.

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	791	0	53200	1.49
LUT as Logic	791	0	53200	1.49
LUT as Memory	0	0	17400	0.00
Slice Registers	687	0	106400	0.65
Register as Flip Flop	687	0	106400	0.65
Register as Latch	0	0	106400	0.00
F7 Muxes	0	0	26600	0.00
F8 Muxes	0	0	13300	0.00

Figure 6: Area Report for DCT

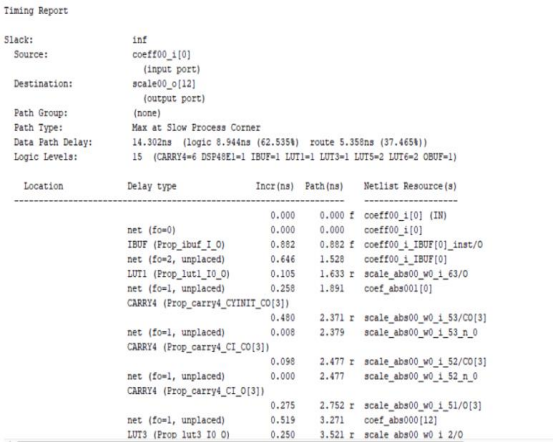
The device design summary generates the data for the 8-point DCT's designed area utilization. The hardware resources of the FPGA hardware that can be taken into consideration in the area report are in fact represented by the slice representation in terms of LUTs and flipflops. These slice occupation values are taken into account when examining how the DCT uses the available space.

The hardware space consumption summary for the 8-point DCT is depicted where the required number of slices is 791, slice flip-flops are 687, and the required number of six input LUTs is similarly 791.

Ambient Temp (C)	25.0
ThetaJA (C/W)	11.5
Airflow (LFM)	250
Heat Sink	none
ThetaSA (C/W)	0.0
Board Selection	medium (10"x10")
# of Board Layers	8to11 (8 to 11 Layers)
Board Temperature (C)	25.0

Figure 7: Power Report for DCT

The total power usage for DCT is equivalent to 1.34884mw. The temperature will be 0.115mw when the ambient temperature of 0.25mw is reached.



```
Timing Report

Slack:                  inf
  Source:               coeff00_i[0]
                          (input port)
  Destination:          scale00_o[12]
                          (output port)
  Path Group:           (none)
  Path Type:            Max at Slow Process Corner
  Data Path Delay:      14.302ns  (logic 8.944ns (62.535%)  route 5.358ns (37.465%))
  Logic Levels:         15  (CARRY4=6 DSP48E1=1 IBUF=1 LUT1=1 LUT3=1 LUT5=2 LUT6=2 OBUF=1)

    Location            Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                     0.000     0.000 f  coeff00_i[0] (IN)
                      net (fo=0)                     0.000     0.000    coeff00_i[0]
                      IBUF (Prop_ibuf_I_O)           0.882     0.882 f  coeff00_i_IBUF[0]_inst/O
                      net (fo=2, unplaced)           0.646     1.528    coeff00_i_IBUF[0]
                      LUT1 (Prop_lut1_I0_O)          0.105     1.633 r  scale_abs00_w0_i_63/O
                      net (fo=1, unplaced)           0.258     1.891    coef_abs001[0]
                      CARRY4 (Prop_carry4_CYINIT_CO[3])
                                                     0.480     2.371 r  scale_abs00_w0_i_53/CO[3]
                      net (fo=1, unplaced)           0.008     2.379    scale_abs00_w0_i_53_n_0
                      CARRY4 (Prop_carry4_CI_CO[3])
                                                     0.098     2.477 r  scale_abs00_w0_i_52/CO[3]
                      net (fo=1, unplaced)           0.000     2.477    scale_abs00_w0_i_52_n_0
                      CARRY4 (Prop_carry4_CI_O[3])
                                                     0.275     2.752 r  scale_abs00_w0_i_51/O[3]
                      net (fo=1, unplaced)           0.519     3.271    coef_abs000[12]
                      LUT3 (Prop_lut3_I0_O)          0.250     3.521 r  scale_abs00_w0_i_2/O
```

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	868	0	134600	0.64
LUT as Logic	868	0	134600	0.64
LUT as Memory	0	0	46200	0.00
Slice Registers	0	0	269200	0.00
Register as Flip Flop	0	0	269200	0.00
Register as Latch	0	0	269200	0.00
F7 Muxes	0	0	67300	0.00
F8 Muxes	0	0	33650	0.00

Figure 9: Area Report for Quantization

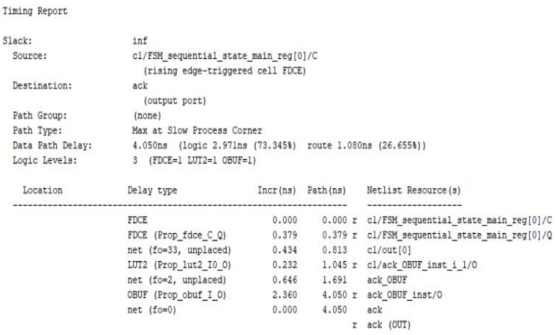
The device design summary generates the reports for the quantization block's designed area use. The hardware resources of the FPGA hardware that can be taken into consideration in the area report are in fact represented by the slice representation in terms of LUTs and flip flops. These slice occupation values are taken into account when quantifying the area usage. The hardware area consumption summary for the quantization is depicted where the required number of slices is 868, the required number of DSPs is 8, and the required number of the six input LUTs is similarly 868.

Ambient Temp (C)	25.0
ThetaJA (C/W)	1.9
Airflow (LFM)	250
Heat Sink	medium (Medium Profile)
ThetaSA (C/W)	3.4
Board Selection	medium (10"x10")
# of Board Layers	12to15 (12 to 15 Layers)
Board Temperature (C)	25.0

Figure 10: Power Report for Quantization

The total power used for quantization is 0.99361 mw, respectively. The temperature will be 0.019 mw when the ambient temperature of 0.25 mw is

reached.



```
Timing Report

Slack:                  inf
  Source:               c1/FSM_sequential_state_main_reg[0]/C
                          (rising edge-triggered cell FDCE)
  Destination:          ack
                          (output port)
  Path Group:           (none)
  Path Type:            Max at Slow Process Corner
  Data Path Delay:      4.050ns  (logic 2.971ns (73.345%)  route 1.080ns (26.655%))
  Logic Levels:         3  (FDCE=1 LUT2=1 OBUF=1)

    Location            Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                        FDCE                         0.000     0.000 r  c1/FSM_sequential_state_main_reg[0]/C
                        FDCE (Prop_fdce_C_Q)         0.379     0.379 r  c1/FSM_sequential_state_main_reg[0]/Q
                        net (fo=33, unplaced)        0.434     0.813    c1/out[0]
                        LUT2 (Prop_lut2_I0_O)        0.232     1.045 r  c1/ack_OBUF_inst_i_1/O
                        net (fo=2, unplaced)         0.646     1.691    ack_OBUF
                        OBUF (Prop_obuf_I_O)         2.360     4.050 r  ack_OBUF_inst/O
                        net (fo=0)                   0.000     4.050    ack
                                                                     r  ack (OUT)
```

Figure 11: Timing Report for Inter Prediction

A synthesis report comprising all the precise and detailed information required for the timing delay evaluation is generated following the synthesis procedure in Xilinx Vivado. Inter Prediction has a timing period analysis summary of 4.050 ns (2.971 ns logic delay and 1.080 ns routing delay) with a 246.91 MHz frequency operation, as indicated in the figure 17 above. The delay values are around 73.345 and 26.655 percent, respectively, depending on the fraction of logic that is occupied.Figure 8: Timing Report for Quantization

The proposed quantization method uses the number of bits provided in the design to produce the desired outputs, hence accelerating the process as a whole.

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	5528	0	134600	4.11
LUT as Logic	5400	0	134600	4.01
LUT as Memory	128	0	46200	0.28
LUT as Distributed RAM	0	0		
LUT as Shift Register	128	0		
Slice Registers	7330	0	269200	2.72
Register as Flip Flop	7330	0	269200	2.72
Register as Latch	0	0	269200	0.00
F7 Muxes	0	0	67300	0.00
F8 Muxes	0	0	33650	0.00

Figure 12: Area Report for Inter Prediction

The device design summary generates the reports for the area usage of the Inter Prediction block created. The slice representation does, in fact, represent the hardware resources of the FPGA hardware that can be taken into consideration in the area report created, both in terms of LUTs and flipflops. For the Inter Prediction's analysis of the area's utilization, these slice occupancy values are taken into account. Figure 17 depicts the hardware space consumption breakdown for the Inter Prediction, where the required number of slices is 5528, LUTs are 5400, and the required number of input LUTs is 6 which is also 128.

On-Chip	Power (W)	Used	Available	Utilization (%)
Slice Logic	12.261	17451	---	---
LUT as Logic	10.146	5400	134600	4.01
CARRY4	1.461	1039	33650	3.09
Register	0.648	7330	269200	2.72
BUFG	0.006	1	32	3.13
LUT as Shift Register	<0.001	128	46200	0.28
Others	0.000	549	---	---
Signals	13.258	9581	---	---
I/O	11.598	68	400	17.00
Static Power	0.712			
Total	37.829			

Figure 13: Power Report for Inter Prediction

The entire power usage for Inter Prediction is equal to 0.378291mw. The temperature will be equivalent to 0.019mw from the ambient temperature of 0.25mw.

V. CONCLUSION

In this specific project, we have created an encoder that can support data or pixels up to a maximum width of 64x64, starting at 8x8. With the use of DCT, data from the original frame is sampled. The DCT outputs are taken into account while evaluating the quantization procedure. Here, we note that the inputs will consist of both signed and unsigned numbers, and the generated DCT values will be divided according to the acquired values.

Inter prediction is carried out by comparing the data relating to the reference frame and the original frame. It is not necessary to develop a specific ME block for this design since both IME and FME operations are executed in this specific process. The inter prediction operation is developed using the controller block, which comprises of a control unit, an address generator, and a data generator. Up to 1080p resolutions, as well as 4K, 8K, and 4K-UHD resolutions, are handled by the H.265 Encoder and will be broadcast or transmitted at either 30 or 60 frames per second.

VI. FUTURE SCOPE

One of the significant characteristics that dramatically improves the coding efficiency of H.265 (HEVC), especially for high resolution formats, is the use of configurable block sizes for coding and transformations. In contrast to H.264 (AVC), that employs a macroblock with only a fixed size of 16x16, H.265 (HEVC) establishes a Coding Tree Unit (CTU) with just a maximum size of 64x64. Each CTU can be subdivided into smaller in a hierarchical fashion and can represent smaller blocks of size 4x4 (known as a

quad-tree). The H.265 (HEVC) transforms can also be of various sizes, ranging from 4x4 to 32x32.

The planar mode, the DC [discrete cosine] mode, and 33 directional (or "angular") modes are really only a couple of minor 35 prediction modes provided by HEVC's intra coding. AVC seems to have a total of 9 modes, along with the DC mode and 8 directional modes.

HEVC contains certain extra coding modes in which the transform step and, occasionally, the quantization stage are completely bypassed. It also has greater flexibility and adaptability in its transform and quantization design. A portion of the coded image can be encoded losslessly thanks to HEVC.

REFERENCE

[1] G. J. Sullivan, J.-R. Ohm, W. J. Han, and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) standard,” IEEE Trans. Circ. Syst. Video Technol., vol. 22, no. 12, pp. 1648–1667, Dec. 2012.

[2] M. Grellert, M. Shafique, M. U. K. Khan, L. Agostini, J. C. B. Mattos, and J. Henkel, "An adaptive workload management scheme for HEVC encoding," IEEE Inter. Conf. Image Processing (ICIP) 2013.

[3] J. Ohm, G. J. Sullivan, H. Schwarz, T. T. Keng, and T. Wiegand, “Comparison of the coding efficiency of video coding standards including high efficiency video coding (HEVC),” IEEE Trans. Circ. Syst.Video Technol., vol. 22, no. 12, pp. 1669–1684, Dec. 2012.

[4] H. Yun, J. Ostermann, M. Domański, Oscar C. Au, N. Ling, “Introduction to the Issue on Video Coding: HEVC and Beyond,” IEEE Journal of Selected Topics in Signal Processing, vol. 7, pp. 931-933, Dec. 2013.

[5] C.-Y. Lung and C.-A. Shen, “Design and implementation of a highly efficient fractional motion estimation for the HEVC encoder,” Journal of Real-Time Image Processing, pp. 1-17, Dec. 2016.

[6] S. Y. Jou, S. J. Chang and T. S. Chang, “Fast Motion Estimation Algorithm and Design for Real Time QFHD High Efficiency Video Coding,” IEEE Trans. Circ. Syst. Video Technol., vol. 25, pp. 1533-1544, Sep. 2015.

[7] Y. Xu, J. Liu, Z. Zhang and R. K. F. Teng, “A High-Performance VLSI Architecture for Integer Motion Estimation in HEVC,” IEEE Inter. Conf. ASIC 2013, pp. 1-4.

[8] Medhat, A. Shalaby, M. S. Sayed, M. Elsabrouty and F. Mehdipour, “Fast Center Search Algorithm with Hardware Implementation for Motion Estimation in HEVC Encoder,” IEEE Inter. Conf. Electr. Circ. and Syst. 2014.

[9] 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.

[10] 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.

[11] 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 or Video Technology, vol. 24, no. 1, pp. 168- 178, Jan. 2014.

[12] 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.

[13] 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.

[14] ITU-T Recommendation H.265, “High efficiency video coding, “International Telecommunication Union, Apr. 2013.

[15] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 12, Dec. 2012, pp. 1649-1668.