FPGA Based Architecture of Video Compression Standard for Space Applications

M.Tech. Thesis

By

YAGNIK KHUSHBU NALINKANT

(Roll No. 1502102011)

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JULY 2017

FPGA Based Architecture of Video Compression Standard for Space Applications

A THESIS

Submitted in partial fulfillment of the requirements for the award of the degree
of

Master of Technology

by

YAGNIK KHUSHBU NALINKANT

(Roll No. 1502102011)

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JULY 2017

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **FPGA BASED ARCHITECTURE OF VIDEO COMPRESSION STANDARD USING TEMPLORAL PREDICTION METHOD FOR SPACE APPLICATION** in the partial fulfillment of the requirements for the award of the degree of **MASTER OF TECHNOLOGY** and submitted in the **DISCIPLINE OF ELECTRICAL ENGINEERING, Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from June, 2016 to June, 2017 under the supervision of Dr. Vivek Kanhangad, Associate Professor, Indian Institute of Technology Indore and Shri Thakar Lalitkrushna J., Scientist-SF, PDMG , ISAC, ISRO

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

(YAGNIK KHUSHBU NALINKANT)

Date:

(Roll No. 1502102011)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Signature of the Supervisor — Signature of the Supervisor

SHRI THAKAR LALITKRUSHNA J. — **Dr. VIVEK KANHANGAD**

YAGNIK KHUSHBU NALINKANT has successfully given his M.Tech. Oral Examination held on _____.

Signature of Supervisor of M.Tech. thesis	Convener, DPGC
Date:	Date:
Signature of PSPC Member #1	Signature of PSPC Member #2
Date:	Date:

`

ACKNOWLEDGEMENTS

At first, I would like to express my sincere gratitude to my supervisors, Dr. Vivek Kanhangad and Shree Thakar Lalitkrushna, for their encouragement, guidance and support during my post-graduation study and thesis work.

I would also like to express my gratitude to all the faculty members of Electrical Engineering discipline of IIT Indore who taught me during my course work, which laid a strong foundation for pursuing my project.

I would also like to thank Mr. Anish, Mr. Ashish, Mr. Saurabh, Ms. Parual, Ms. Neetu, Ms. Sushree, Mr. Anuj, Ms. Ashima and all the other colleagues at IIT Indore for providing a wonderful company and for all the productive discussions we had.

And last but certainly not the least, I wish to thank my family members for their absolute love. I could not have accomplished my ambition without their support. I dedicate this thesis to my parents for their countless sacrifices and unbounded love.

Yagnik Khushbu Nalinkant

mt1502102011

M.Tech. (Communication and Signal Processing)

Discipline of Electrical Engineering, IIT Indore

`

Dedicated to My Parents

`

ABSTRACT

Compression is a main technique to reduce the bandwidth requirement for space applications as well as to reduce the storage requirement for data. Mass, volume and power are the most important parameters for space applications. Therefore, compression is the most efficient way to control all these parameters.

Indian space research organization (ISRO) uses consultative committee for space data system (CCSDS) based image compression standard for the image data as well as video data. But CCSDS standard is not able to fulfill the requirement of high compression ratio. Therefore, ISRO planned to implement a real-time video compression standard for space applications. ISRO decided to implement the H.264 video compression standard with baseline profile for space applications. This standard fulfills the requirement of high compression standards with good quality of video. The video has 512 × 512 frame size for space applications. This algorithm can be used to implement the video compression for 512 × 512 resolution video and 352 × 288 resolution video.

Field programmable gate array (FPGA) based design is selected to implement this standard. The complete implementation of this standard within a single FPGA is required to avoid the frequency and power issues. Xilinx and Microsemi are only two companies, which provide space graded and radiation hardened FPGAs with very limited series. This project is targeted on the Virtex 5Q - xq5vfx130t FPGA. This project is placed and routed on this FPGA successfully. It uses 88% of total slices of FPGA and 64% of total block random access memory (RAM) of FPGA. It can work up to 60 MHz input frequency. The basic H.264 algorithm and hardware implementation are explained within this thesis. Simulation waveforms, the compression ratio and the peak signal to noise ratio (PSNR) are discussed in this thesis.

`

TABLE OF CONTENTS

`

`

LIST OF FIGURES

`

LIST OF TABLES

`

CHAPTER - 1

INTRODUCTION AND LITERATURE REVIEW

The space based systems have a bandwidth limitation. Therefore, compression of data is necessary for these systems. This chapter describes the main objective and the reasons for the selection of H.264 video compression standard. The organization of the thesis is described at the end of this chapter.

1.1 Problem statement

ISRO is planning for human spaced mission, interplanetary mission and docking mission. ISRO requires communication between crew members and ground stations. ISRO requires docking event observation also. These activities should be transmitted to earth station in bandwidth and power limited environment. This calls for compression of image or video data to reduce data rate. The image compression is used in satellite. But high compression ratio cannot be achieved by image compression algorithms. Therefore, ISRO requires video compression for above mentioned missions.

Various compression standards are available as software module or hardware chips in the commercial domain. But this solution will not work for the harsh space environment where the temperature is in the range of -55 degrees to 125 degrees and various radiations are present in the environment. These problems are not common in the ground systems. Power and mass are also crucial parameters for the space mission. These requirements call for the development of the space qualified video compression standard.

1.2 Motivation

Various video compression standards were studied for best bandwidth performance and least resource requirement. H.264 video compression standard outperforms in terms of compression ratio, speed and resource utilization with respect to other standards such as moving pictures expert group (MPEG)-2, MPEG-4, H.263, H.261

`

etc. Therefore, ISRO selected H.264 algorithm to implement a video compression standard.

In the space environment, there are resource constraints in terms of computational power, logic resources, mass and volume. Currently available processor (LEON3) works up to 50 million instructions per second. Therefore, this platform is not suitable for implementation of a video compression for space applications and FPGA based design is selected for implementation of the H.264 standard.

1.3 Literature overview

Mass, volume and power are constraints for space applications. Main approach is to implement the algorithm, which reduces these main constraints. Richardson [1] explains the ideas of H.264, its linguistic structure, H.264 prediction, H.264 transform and its execution. Wiegand *et al.* [2] explains the features of H.264, profiles and its applications. Chen *et al*. [3] describes the very large scale integration (VLSI) implementation of CAVLC. Keshaveni *et al*. [4] explains an implementation of CAVLC. Keshaveni *et al*. [5] describes the implementation of integer transform and quantization process for H.264 using FPGA. Kuo *et al*. [6] describes motion estimation and prediction algorithm for H.264. Shi et al. [7] describes different image and video compression standards. Raja *et al*. [8] compares performance of H.264 with H.263 and H.263+ in terms of PSNR. Zhang *et al.* [9] describes the optimization in joint source channel rate distortion for H.264 video compression standard in noisy environment.

1.4 Advantages of H.264 standard

There are many standards for video compression such as MPEG-2, MPEG-4, H.263, H.264, etc. H.264 has many variable parameters, which can be used to get high compression ratio.

H.264 can save bit rate up to 50% compared to MPEG-2 or MPEG-4 with simple profile. H.264 algorithm provides good quality video with high compression ratio. If there is packet loss in the wireless network, video cannot be lost completely. A frame, which is in the lost packet, cannot be recovered. Other frames can be recovered.

`

These features make it more suitable for video compression standard for space applications.

There are many different standards for video compression such as H.261, MPEG 1, MPEG 2 etc. All standards and its applications are given in Table 1.1.

Table 1.1 List of video coding standard

Standard	Main applications	Year
H.261	Video conferencing	1990
H.262	SDTV	1995
H.263 (Baseline profile)	Videophone	1998
MPEG 1	Video CD	1992
MEPG 2	SDTV, HDTV, DVD	1995
MPEG 4 version 2	Interactive video	1999
MPEG 7	Multimedia content description interface	2001
MPEG 21	Multimedia frame work	2002
H.264/ MPEG 4 Part 10	Advance video coding	2003

SDTV, HDTV, DVD and CD indicate standard definition television, high definition television, digital video disc and compact disc, respectively.

1.5 Thesis organization

The thesis is organized as follows-

In Chapter 3, the basic video codec is explained as well as H.264 algorithm is discussed.

In Chapter 4, the implemented hardware block, radiation hardened FPGA for space applications and targeted FPGA are discussed.

In Chapter 5, the simulation results of modules are shown. PSNR and compression ratio are also discussed in this chapter.

In Chapter 6, the future work and conclusion are described.

`

CHAPTER - 2

H.264 OVERVIEW

The main objective and problem statement are discussed in the previous chapter. In this chapter, main H.264 video codec function is described. The basic information of color space, macroblock and submacroblock are also discussed.

2.1 Block diagram of H.264 algorithm

The main block diagram of H.264 is shown in Fig. 2.1. The main function of this block diagram is the conversion of input YUV video data to H.264 bit stream.

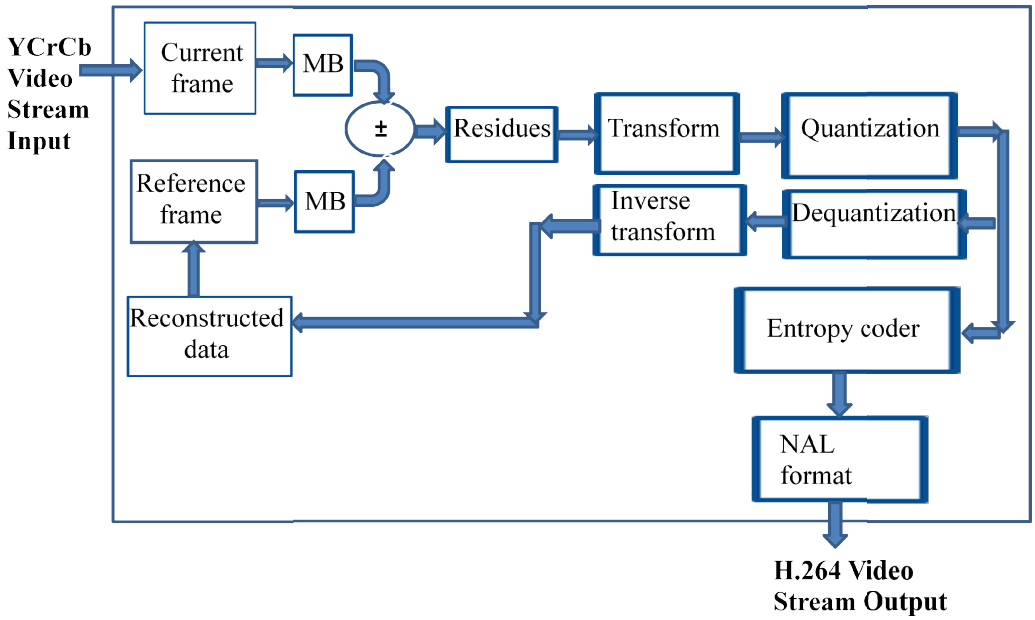


Fig. 2.1 Block diagram of H.264 algorithm

Luminance or brightness is only one component in a monochrome image. But there are two components for color images. These two components are brightness and color.

Video is a collection of images. A pixel is the basic sample of each image. Each pixel has its own color and luminance component. Red, green and blue are the basic colors. H.264 uses YUV color space. Red, green and blue colors have the same resolution in RGB format, which requires more space to store that color image or video. The

`

human visual system is more sensitive to luminance component compared to chrominance component. Therefore, YUV color space is used more than RGB color space. Y indicates the luminance component. Cr, Cb and Cg indicate chrominance components. Y, Cr and Cb can be calculated from RGB using equations (2.1), (2.2) and (2.3). YUV to RGB conversion is done using equations (2.4), (2.5) and (2.6). YUV color space has 4:4:4, 4:2:2 and 4:2:0 sampling format.

$$Y = 0.299R + 0.587G + 0.114B \quad (2.1)$$

$$Cb = 0.564(B - Y) \quad (2.2)$$

$$Cr = 0.713(R - Y) \quad (2.3)$$

$$R = Y + 1.402Cr \quad (2.4)$$

$$G = Y - 0.344Cb - 0.714Cr \quad (2.5)$$

$$B = Y + 1.772Cb \quad (2.6)$$

2.1.1 Macroblock

The image is divided into block size of 16 × 16 pixels, which is known as the macroblock as shown in Fig. 2.2.

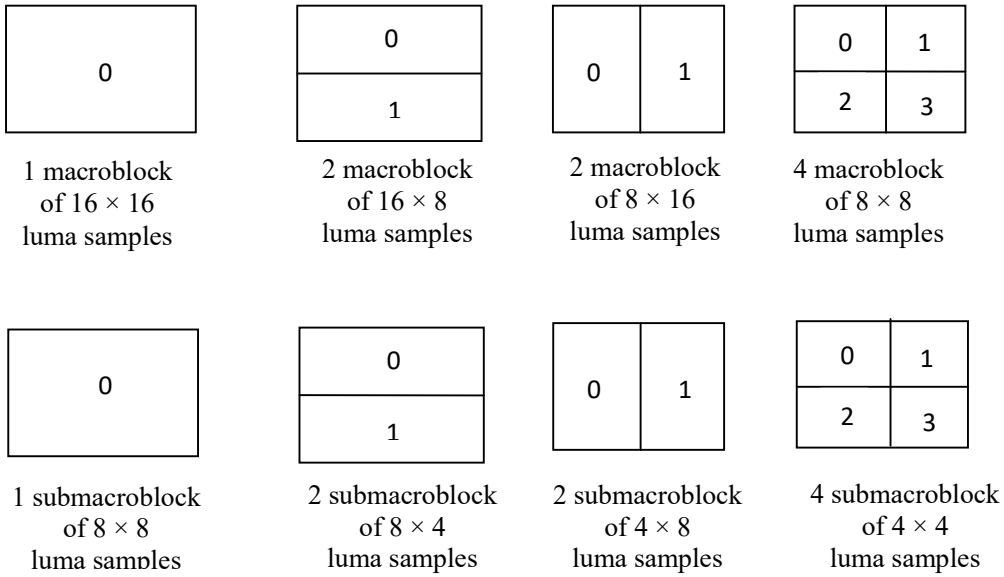


Fig. 2.2 Macroblock and submacroblock partition

Here, luma macroblock can be size of 16 × 16, 16 × 8, 8 × 16 and 8 × 8 pixels. Chroma macroblock size is according to sampling format. If size of luma macroblock is 8 × 8 pixels, it can be further divided to submacroblock. The size of submacroblock of chroma components is according to sampling format. Minimum submacroblock

`

size of luma and chroma component can be 4 × 4 pixels. Here, submacroblock is scanned in a zigzag manner such as 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15 as shown in Fig. 2.3.

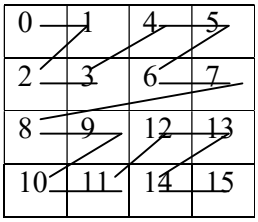


Fig. 2.3 Zig-zag scanning for submacroblock

2.1.2 Prediction

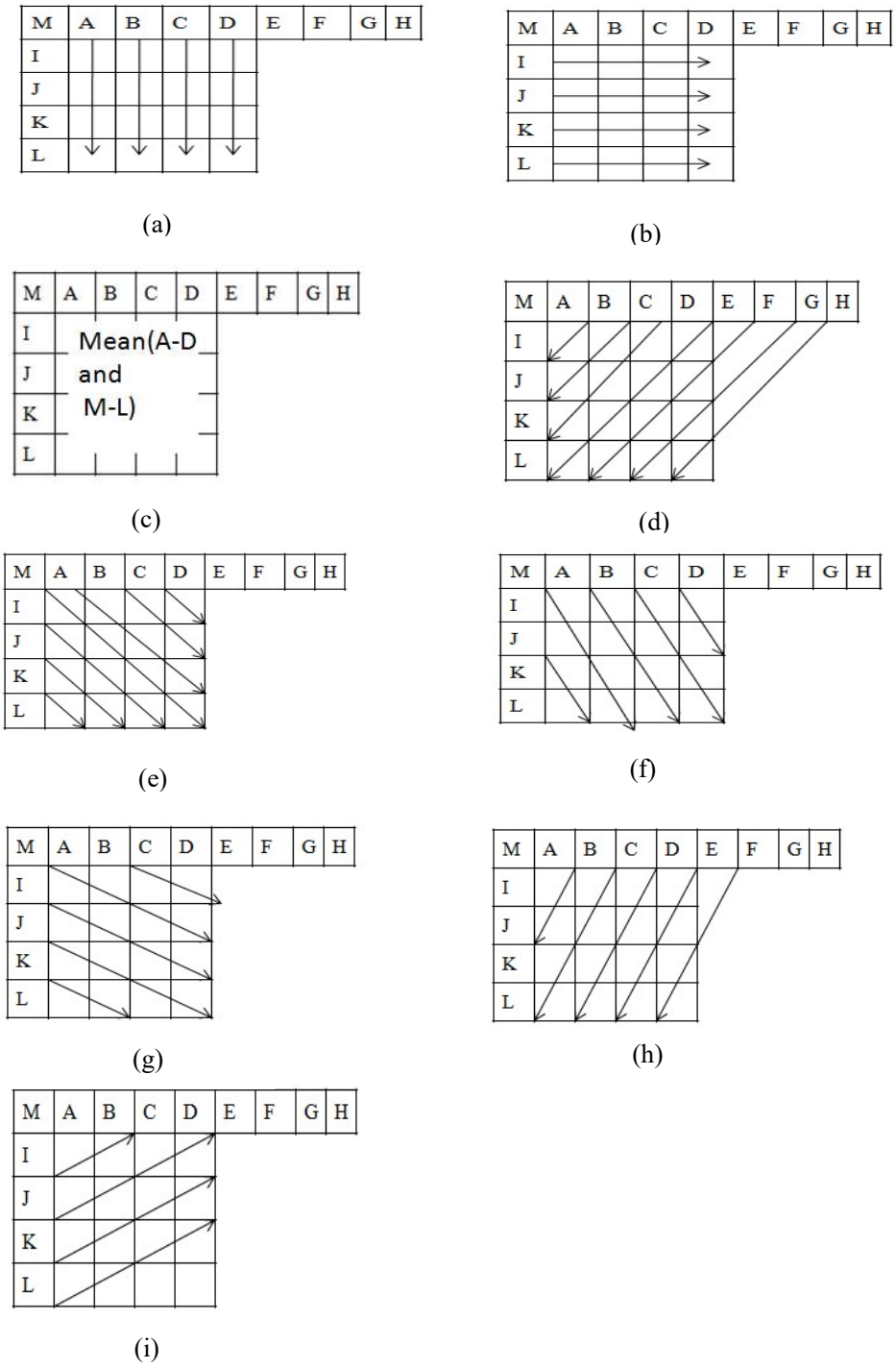
Prediction helps to achieve a high compression ratio in H.264 algorithm. Intra prediction and inter prediction are two types of predictions, which are used in H.264 algorithm.

Intra prediction is also known as spatial prediction. Intra prediction is done within the frame. The pixels of current block are predicted using pixel values of current frame, which are already encoded. The pixels of current macroblock are highly correlated to neighbor pixel values. Therefore, this previously coded neighbor pixel values are used to predict the pixel values of current block. Fig. 2.4 shows the block of the size of 4 × 4 pixels. Here, A, B, C, D, E, F, G, H, I, J, K, L and M are previously coded pixels in Fig. 2.4. The current block pixels are a, b, c, d, e, f, g, h, i, j, k, l, m, n, o and p in Fig. 2.4.

M	A	B	C	D	E	F	G	H
I	a	b	c	d				
J	e	f	g	h				
K	i	j	k	l				
L	m	n	o	p				

Fig. 2.4 Sample pixels of intra prediction

`



(a) vertical mode (b) horizontal mode (c) DC mode (d) diagonal left down mode (e) diagonal down right mode (f) vertical left mode (g) horizontal down mode (h) vertical right mode (i) horizontal up mode.

Fig. 2.5 Intra prediction modes of 4 × 4 pixel block

`

In mode 0 (vertical), upper pixels are extrapolated as shown in Fig. 2.5 (a). The left pixels are extrapolated horizontally for mode 1 (horizontal) operation, which is shown in Fig. 2.5 (b). In mode 2 (DC), the current pixel block is extrapolated by average of A...D and I...L, which is shown in Fig. 2.5 (c). In mode 3 (diagonal left down), extrapolation of pixel at an angle of 45 degrees between down left and upper right is performed, which is also shown in Fig. 2.5 (d). Extrapolation of pixels at an angle of 45 degrees down and to right for mode 4 (diagonal down right) is done, which is also shown in Fig. 2.5 (e). Fig. 2.5 (f) shows mode 5 (vertical left). The pixels are extrapolated by an angle of 26.6 degrees to the left of the vertical in mode 5. Fig. 2.5 (g) shows mode 6 (horizontal down). The pixels are extrapolated by an angle of 26.6 degrees below horizontal in mode 6. In mode 7 (vertical right), the pixels are extrapolated or interpolated by an angle of 26.6 degrees to the right of the vertical, which is shown in Fig. 2.5 (h). The pixels are interpolated by an angle of 26.6 degrees above horizontal for mode 8 (horizontal up), which is shown in Fig. 2.5 (i).

The intra prediction method does not give high compression ratio for highly correlated frames. Therefore, the inter prediction method is used for compression as well as for good quality of video. The inter prediction is known as temporal prediction. The inter prediction is done using the correlation property between two frames. The neighboring frames are highly correlated to each other. Therefore, this redundancy is used to achieve a high compression ratio.

The temporal prediction can be done using one or more previous or next frames. These frames are called as the reference frames. The accuracy can be improved using compensation of motion between the reference frames and the current frames. In the simple temporal prediction method, the previous frame is used as the reference frame for the current frame. Movement of the blocks of the current frame is used for motion estimation and compensation. The process for block of M × N samples is described in Fig. 2.6. The block of M × N samples in the current frame and the same size of block in the reference frame are compared. All or some possible combinations of same size block in the search area are taken for comparison with the current block. The residue energy of each combination is calculated by subtracting the reference frame block from the current frame block for each combination. Sum of absolute energy (SAE) is also calculated for each combination by adding the absolute value of residual energy

`

of all samples. The combination, which has the minimum value of SAE among all combinations, is the best match. This process of finding the best match is known as the motion estimation. Best match block is the predictor block for the current frame block. The process of finding the residues for the best match case is known as the motion compensation. This block of residues is used for further process such as transformation, quantization, encoding, etc. The difference between the current block position and the predicted block position is transmitted, which is known as a motion vector.

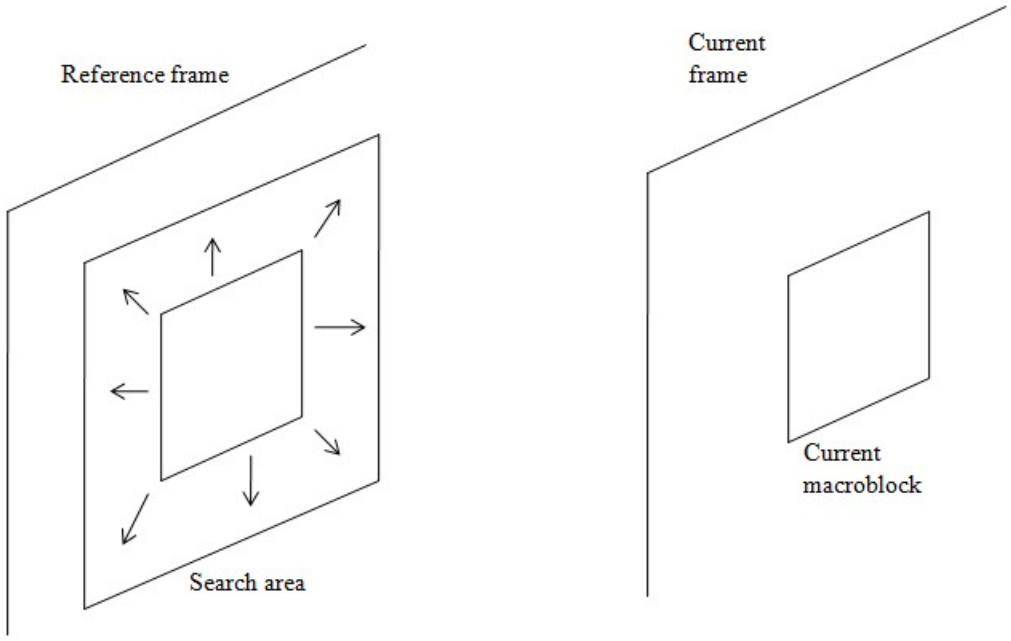


Fig. 2.6 Motion estimation

The decoder has the residues and the motion vector values. It finds out the predictor block from the previously decoded frame using motion vector value. This predictor block and the residual block are added to reconstruct values of the current pixels. The blocks are used for the motion estimation because it is simple in calculation and it has less complexity. It fits easily with a rectangular frame. DCT is applicable to block of residues.

P prediction and B prediction are two prediction types in H.264 algorithm. In P prediction type, previously coded frames are used for the inter prediction as shown in Fig. 2.7.

`

In this Fig. 2.7, block A in a frame *n* is coded from *n*-1 frame. Block B in a frame *n* is coded from *n*-2 frame.

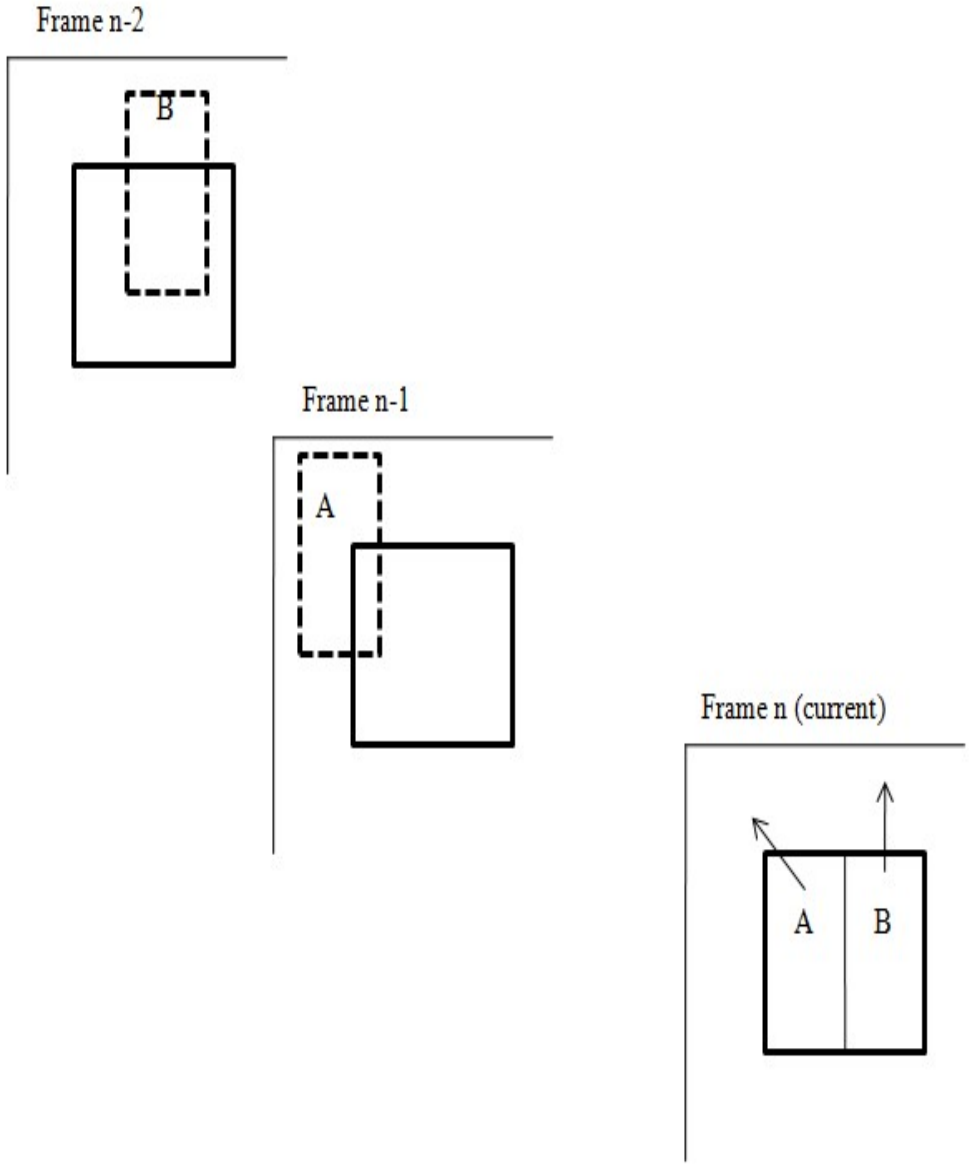


Fig. 2.7 P prediction

In B type of prediction, both the previous and the future frames are used for the motion estimation and the prediction, which are shown in Fig. 2.8. In this Fig. 2.8, block 0 in the frame *n* is predicted from frame *n*-1. Block 1 in the frame *n* is predicted from frame *n*-1 and frame *n*+1.

`

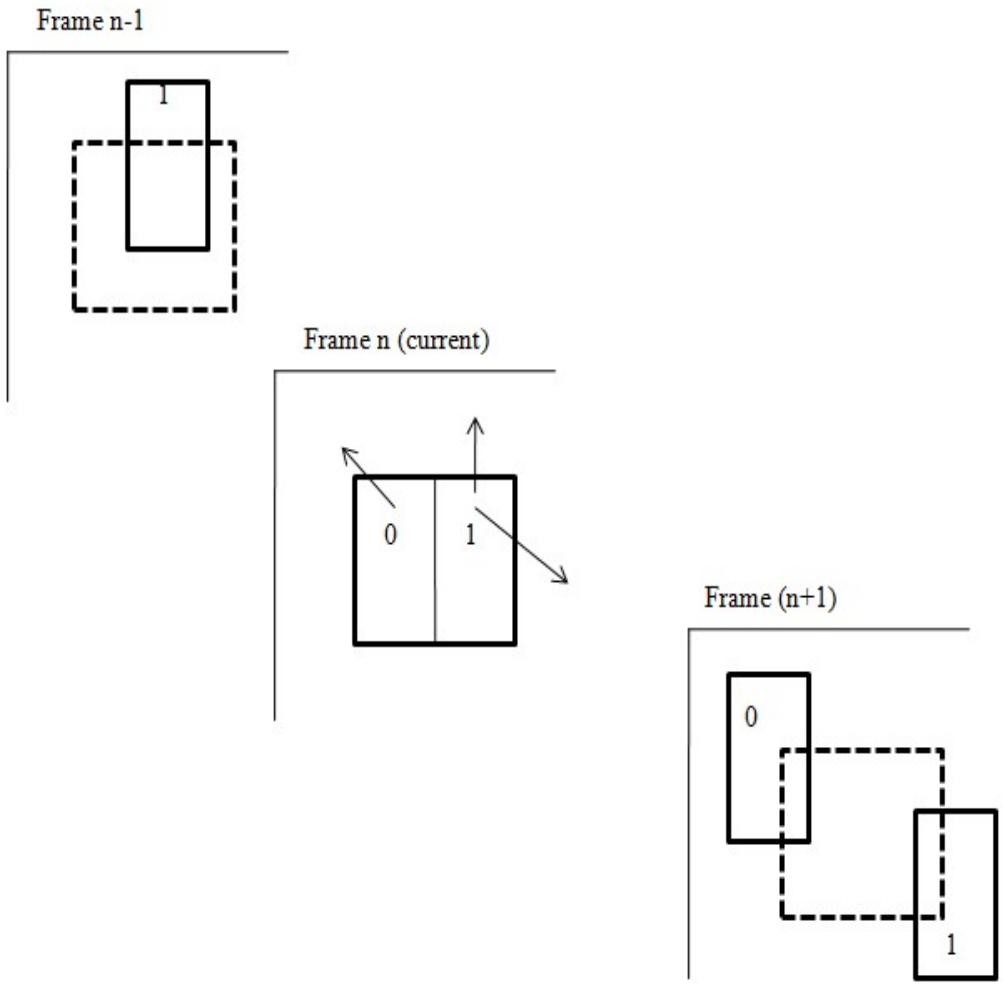


Fig. 2.8 B prediction

2.1.3 Transformation and quantization

The best predicted residues are transformed to another domain and quantized to compress the data. DCT is used in H.264 standard. To minimize calculation complexity, DCT and quantization are combined together. Core-transform or integer transform is used for fixed point arithmetic. Fig. 2.9 shows the combined procedure of DCT and quantization.

Core-transform is used for H.264 compression standard. The equation (2.7) is core-transform conversion equation. QP indicates step size parameter.

`

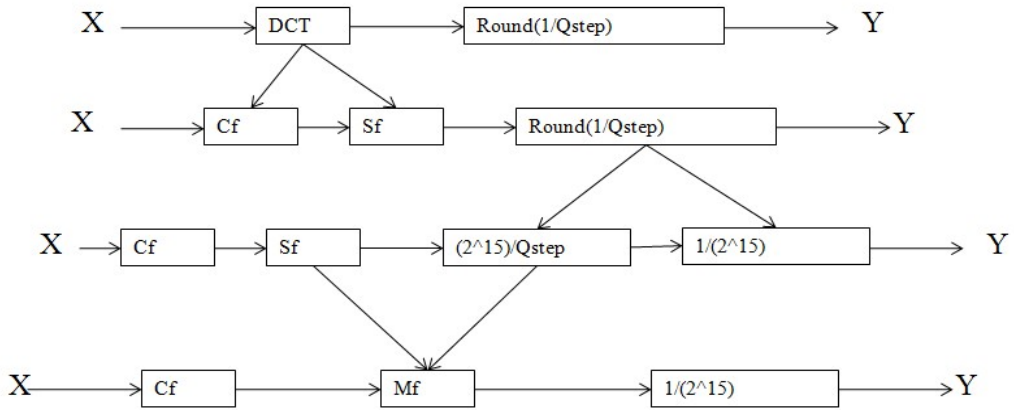


Fig. 2.9 Combination process of transform and quantization

$$Y = C_{f4} X C_{f4}^{T} \tag{2.7}$$

$$\text{Where, } C_{f4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \tag{2.8}$$

DC transform is used in H.264 compression standard. DC transform for 2 × 2 input matrix can be calculated using equation (2.9).

$$Y_D = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} [w] \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.9}$$

M_{f4} is the derived matrix, which is shown in Fig. 2.9. QP value for M_{f4} can be calculated using equation (2.10). This matrix denotes various values depending upon the value of step size parameter QP. Different QP value has different M_{f4} matrix. This matrix value can be decided by using Table 2.1 and equation (2.11).

$$QP = QP \,\%\, 6 \tag{2.10}$$

where, % indicates modulo operator.

$$M_{f4} = \begin{bmatrix} m(QP,0) & m(QP,2) & m(QP,0) & m(QP,2) \\ m(QP,2) & m(QP,1) & m(QP,2) & m(QP,1) \\ m(QP,0) & m(QP,2) & m(QP,0) & m(QP,2) \\ m(QP,2) & m(QP,1) & m(QP,2) & m(QP,1) \end{bmatrix} \tag{2.11}$$

`

Table 2.1 M_{f4} values for different QP values

QP	**Positions (0,0),(2,0) (0,2),(2,2)**	**Positions (1,1),(1,3) (3,1),(3,3)**	**Other Positions**
0	13107	5243	8066
1	11916	4660	7490
2	10082	4194	6554
3	9362	3647	5825
4	8192	3355	5243
5	7282	2893	4559

2.1.4 Inverse transformation and dequantization

H.264 uses DCT and quantization. Here, dequantization and inverse transform is combined together, which is shown in Fig. 2.10. The reverse process is done in H.264 to reconstruct the coded frame, which is used for the prediction of the next frame. IDCT indicates inverse DCT in Fig. 2.10.

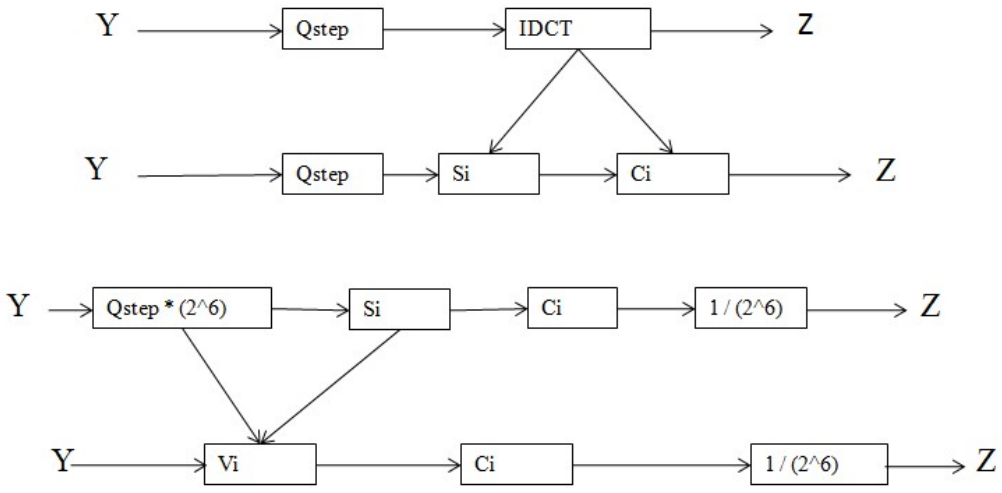


Fig. 2.10 Combination process of inverse transform and dequnatization

Y is the input for the inverse core transform in equation (2.12). Z is the output of this inverse core-transform in equation (2.12). Inverse core-transform can be calculated using equation (2.12).

$$Z = C_{i4}^{T} Y C_{i4} \tag{2.12}$$

`

$$\text{where, } C_{i4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix} \tag{2.13}$$

The inverse DC transform for 2 × 2 matrix is calculated using equation (2.14).

$$W_{QD} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} [Z_D] \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.14}$$

V_{i4} is shown in Fig. 2.10. QP value for V_{i4} can be calculated using equation (2.10). There are different values of V_{i4} according to QP value. Different V_{i4} can be calculated using Table 2.2. and equation (2.15).

$$V_{i4} = \begin{bmatrix} v(QP,0) & v(QP,2) & v(QP,0) & v(QP,2) \\ v(QP,2) & v(QP,1) & v(QP,2) & v(QP,1) \\ v(QP,0) & v(QP,2) & v(QP,0) & v(QP,2) \\ v(QP,2) & v(QP,1) & v(QP,2) & v(QP,1) \end{bmatrix} \tag{2.15}$$

Table 2.2 V_{i4} values for different QP values

QP	**Positions (0,0),(2,0) (0,2),(2,2)**	**Positions (1,1),(1,3) (3,1),(3,3)**	**Other Positions**
0	10	16	13
1	11	18	14
2	13	20	16
3	14	23	18
4	16	25	20
5	18	29	23

2.1.2.6 Entropy encoder

Inverse transform, dequantization and inverse DC transform are used for the reconstruction of the frame in the reverse path. In the forward path, quantized residues go to entropy coder block. Context adaptive variable length coding (CAVLC) and context adaptive binary arithmetic coding (CABAC) are two coding algorithms, which are supported by H.264 standard. The baseline profile supports CAVLC encoding algorithm. The higher profiles support CABAC algorithm in H.264. Here, context adaptive algorithms are used because CAVLC and CABAC both depend on

previous block coefficients. Therefore, it gives more compression compared to other entropy coding algorithms such as huffman coding, run length encoding etc.

2.1.6 Header

After entropy encoding, the header should be placed in its appropriate position such as slice header should be placed at the beginning of the slice. In H.264 standard, network abstraction layer (NAL) is used to pass the header information to the decoder. The header syntax structure is shown in Fig. 2.11. There are sequence parameter set (SPS) and picture parameter set (PPS). These include information such as resolution of video, profile, level etc.

SPS is common to video sequence. PPS can be more than one to a video sequence. The slice header indicates the type of slice such as I slice, P slice, B slice, etc. Slice data denote the macroblocks of that slice.

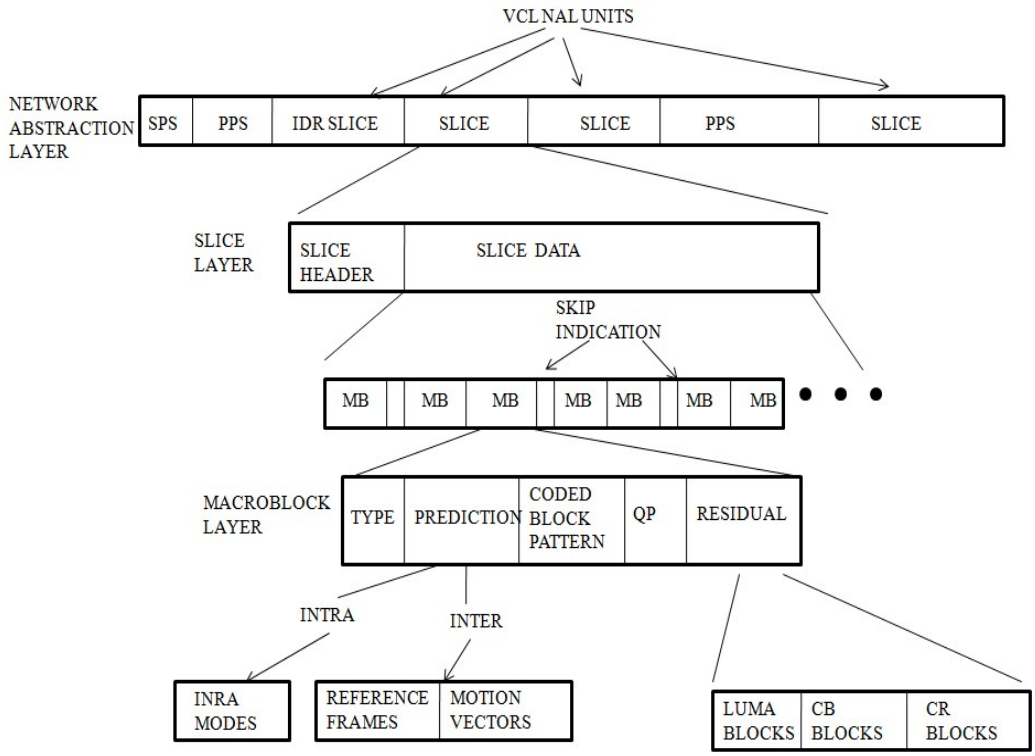


Fig. 2.11 Syntax structure of header

Macroblock type denotes the type of macroblock such as I, P or B macroblock. Prediction indicates intra prediction or inter prediction for that particular macroblock. Intra indicates modes for that particular macroblock, which are discussed earlier. Inter

`

denotes reference frame numbers and motion vector for macroblock. Coded block pattern (CBP) provides information about the nonzero residual coefficients of luma and chroma block. QP provides the information about QP (step size parameter) for that particular macroblock. This field denotes residues of luma and chroma blocks. The order of these residues is luma blocks, Cb blocks and Cr blocks.

`

CHAPTER - 3

HARDWARE IMPLEMENTATION

The basic algorithm of H.264 is explained in the previous chapter. Very high speed integrated circuit (VHSIC) hardware description language (VHDL) is used for the hardware implementation on FPGA. The small modules are discussed in this chapter. The radiation hardened FPGA for the space applications and targeted FPGA for this algorithm are also explained in this chapter.

3.1 Hardware implementation of H.264

The hardware implementation of H.264 algorithm is shown in Fig. 3.1. Here, the output of the camera is in the bayer format. Therefore, it is converted to RGB and RGB is converted to YUV format. Here, the baseline profile is used. Therefore, YUV is converted to 4:2:0 sampling format. There is one selection signal for the intra module and the inter module. Intra module to inter module selection ratio is 1:10 in this project. Therefore, the input (YUV 4:2:0) goes to intra or inter module as per the selection signal. Here, luma and chroma are processed simultaneously in both the inter module and the intra module. DC transform is used for the chroma components. DC transform is not used for the luma components in this project. The output of the luma module and the output of the chroma module go to the core-transform module. An average of chroma residues goes to the DC transform module for each submacroblock. The outputs of the core-transform module and the DC transform module go to the quantization module using multiplexer.

In the reverse path, dequantization, inverse transform, inverse DC transform and reconstruction occur. This reconstructed output is stored in the memory. The inter module uses reconstructed pixel values of the previous frame for the prediction of the current frame block. The intra module uses the reconstructed value of the previous pixels of the current frame for the prediction of current block of the current frame.

`

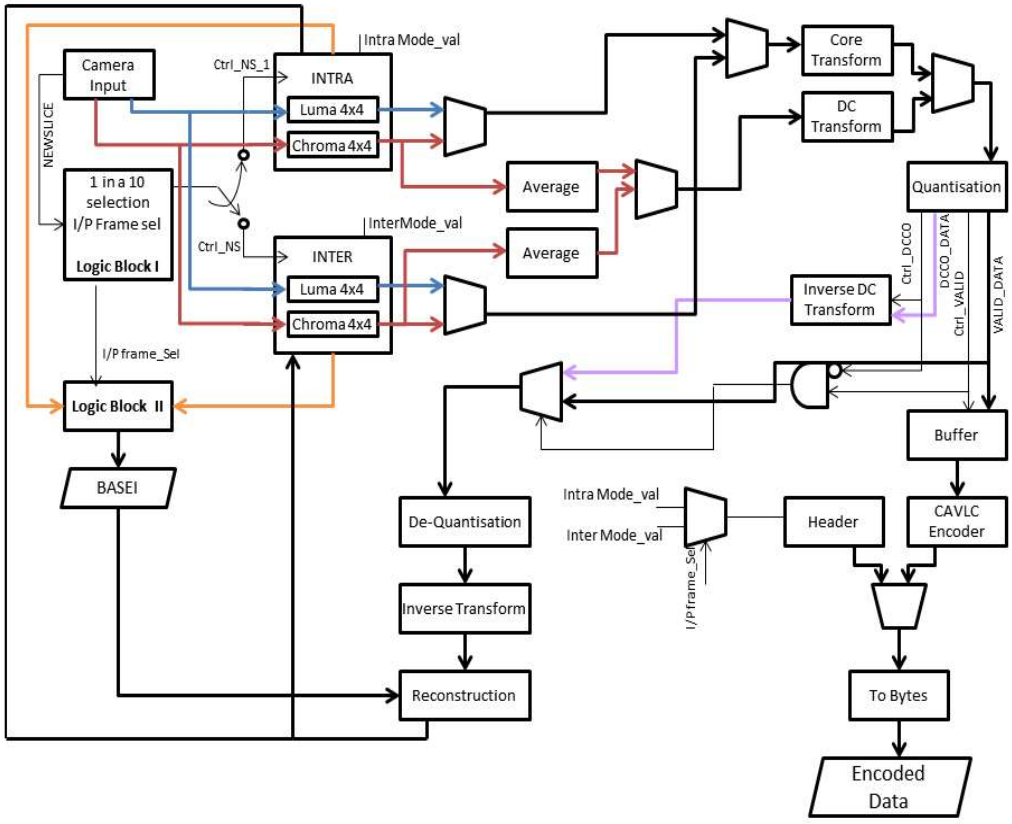


Fig. 3.1 Block diagram of hardware implementation of H.264

In the forward path, the output of the quantization process goes to the buffer module, which is used to rearrange the output in the desired format. This format will be discussed later in this chapter. Then the output of the buffer module goes to the CAVLC encoder. There is a header module, which is used to generate the slice header, macroblock header, SPS header, PPS header, etc. The output of the header block and the encoder module go to the multiplexer. The multiplexed output goes to the tobytes module. The tobytes module is used to generate byte format. The output of the tobytes module is H.264 bit stream. The logic block I and the logic block II are used for the selection of the intra or inter prediction for the core-transform module and the reconstruction module, respectively.

`

3.2 FPGA architecture overview

3.2.1 FPGA

FPGA is used mainly for the hardware applications. Microsemi and Xilinx are only two companies, which manufacture the space graded FPGAs. Microsemi RTAX 4000 has 20,160 register cells. It has 40,320 combinational cells, which are used to implement the combinational logics. There are 40,320 flip flops, which are used for sequential logics. It has 540 kb core RAM. Microsemi RTG4 has 1,51,824 logic elements, 5.2 Mb synchronous random access memory (SRAM). Xilinx Virtex 5Q-xq5vfx130t has 20,480 configurable logic blocks (CLB) slices, 320 DSP48E slices and 10,728 kb block RAM.

3.2.2 Targeted FPGA

If more than one FPGA is used for this implementation, the desired frequency of operation and power will be affected. A single chip solution is required to avoid these problems. Initially this algorithm was targeted to RTAX 4000 because board complexity and cost are less of this FPGA compared to other FPGA such as Virtex 5Q and RTG4. But numbers of resources are not sufficient in RTAX 4000 to implement entire algorithm in a single FPGA.

RTG4 was also targeted after the failure in RTAX 4000 because static power in RTG4 is less than Virtex 5Q. But the resources are not sufficient in RTG4 for this complete algorithm. Then Virtex 5Q series was targeted. The algorithm is implemented in this FPGA successfully. The source utilization report is shown below.

66% of slice registers, 49% look up table (LUT) and 88% slices are occupied by this algorithm. 64% of block RAM is used by this algorithm after place and route operation.

3.3 Explanation of modules

H.264 supports video of YUV format. Here, the input data is in the serial and byer format. Therefore, this data is converted to YUV (4:2:0) format. The pixel values go to the inter module after this conversion. This conversion is the input interface of this entire algorithm.

`

3.3.1 Inter module

Inter_luma_module and inter_chroma_module are included in the inter module. The inter_luma_module is used for the inter prediction of luma samples and the inter_chroma_module is used for the inter prediction of chroma samples. The inter_luma_module is shown in Fig. 3.2.

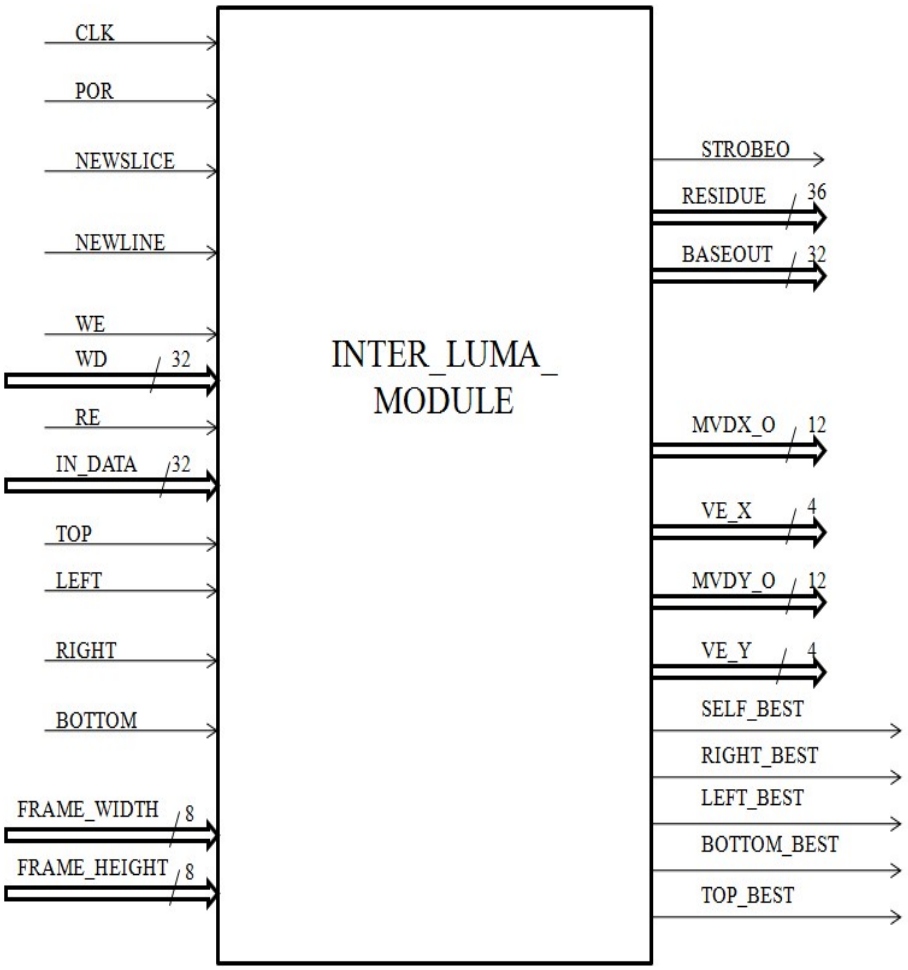


Fig. 3.2 Block diagram of the inter_luma_module

Here, the input signals are NEWSLICE, NEWLINE, WE, WD RE, IN_DATA, TOP, LEFT, RIGHT, BOTTOM, FRAME_WIDTH and FRAME_HEIGHT for the inter_luma_module. WE becomes high when reconstructed luma data is available at WD. The data at WD is written to memory for the prediction of the next frame. RE becomes high when data at IN_DATA is available. IN_DATA is the current frame data, which is predicted using the previous frame data. TOP, LEFT, RIGHT and

`

BOTTOM are control signals, which are used to decide appropriate calculation according to the position of macroblock in the current frame, i.e. if the position of current macroblock is at the right edge corner of a frame, left, bottom and self comparisons are possible with the previously coded data of the previous frame.

The output signals are STROBEO, RESIDUE, BASEOUT, MVDX_O, MVDY_O, VE_X, VE_Y, SELF_BEST, RIGHT_BEST, TOP_BEST, BOTTOM_BEST and LEFT_BEST for the inter_luma_module. When STROBEO becomes high, data at RESIDUE and data at BASEOUT are valid data. The data at RESIDUE are used for further calculation such as core-transform, quantization, entropy coding etc. The data at BASEOUT are used for the reconstruction process in the backward path. MVDX_O and MVDY_O are the motion vectors for the x-direction and y-direction, respectively. VE_X and VE_Y show the number of valid bits at MVDX_O and MVDY_O, respectively. These motion vectors are given to the header module. SELF_BEST, RIGHT_BEST, TOP_BEST, BOTTOM_BEST and LEFT_BEST go to the inter_chroma_module to calculate residuals. For the baseline profile, the motion vectors for chroma component and luma component are the same. The inter_chroma_module is shown in Fig. 3.3.

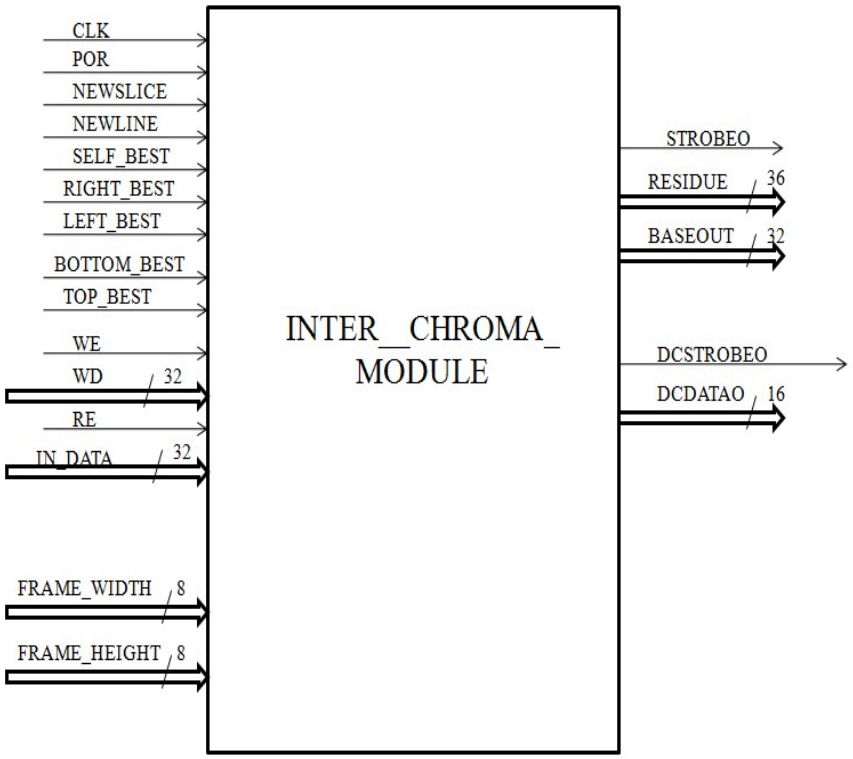


Fig. 3.3 Block diagram of the inter_chroma_module

`

The input signals are CLK, POR, NEWSLICE, NEWLINE, SELF_BEST, RIGHT_BEST, LEFT_BEST, BOTTOM_BEST, TOP_BEST, WE, WD, RE, IN_DATA, FRAME_WIDTH and FRAME_HEIGHT for the inter_chroma_module. SELF_BEST, RIGHT_BEST, LEFT_BEST, BOTTOM_BEST and TOP_BEST are connected to the output of the inter_luma_module. These signals are used to calculate the residues for chroma components. When WE becomes high, data at WD is valid data, which are reconstructed data. These data are stored in the memory for the prediction of the next frame. When the input of current frame comes, RE becomes high and data at IN_DATA are the current frame data, which are predicted using the SELF_BEST, RIGHT_BEST, LEFT_BEST, BOTTOM_BEST and TOP_BEST signals.

The output signals are STROBEO, RESIDUE, BASEOUT, DCSTROBEO and DCDATAO for inter_chroma_module. When STROBEO becomes high, data at RESIDUE and data at BASEOUT are valid data. The data at RESIDUE are used for further process such as core-transform, quantization, encoding, etc. The data at BASEOUT are used for the reconstruction process. When DCSTROBEO becomes high, data at DCDATAO are valid data, which are connected to the input of the DCTRANSFORM module.

3.3.2 Core-transform module

Core-transform module is shown in Fig. 3.4. The input signals are POR, CLK, ENABLE and XXIN in Fig. 3.4. When ENABLE goes high, data at XXIN indicates valid data for operation. The input of this module is multiplexed output of the inter_chroma_module and the inter_luma_module.

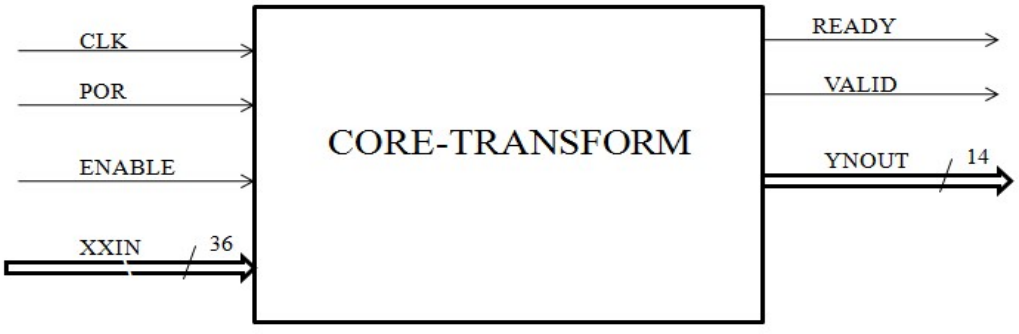


Fig. 3.4 Block diagram of the core-transform module

`

Here, the output signals are VALID, READY and YNOUT in Fig. 3.4. When VALID is high, the output data at YNOUT is valid data. A READY signal used for handshaking. When the READY signal becomes high, it means that the core-transform process on previous data is completed and this module can process new data.

3.3.3 DC transform module

DC transform module is shown in Fig. 3.11. CLK2, POR, RESET, READYO, ENABLE and XXIN are the input signals. When ENABLE becomes high, data at XXIN show valid data. READYO signal used to get the output. When user wants output, the user has to keep the READYO signal high. Therefore, the output can be obtained to the next clock pulse. RESET is used to RESET the DC transform module.

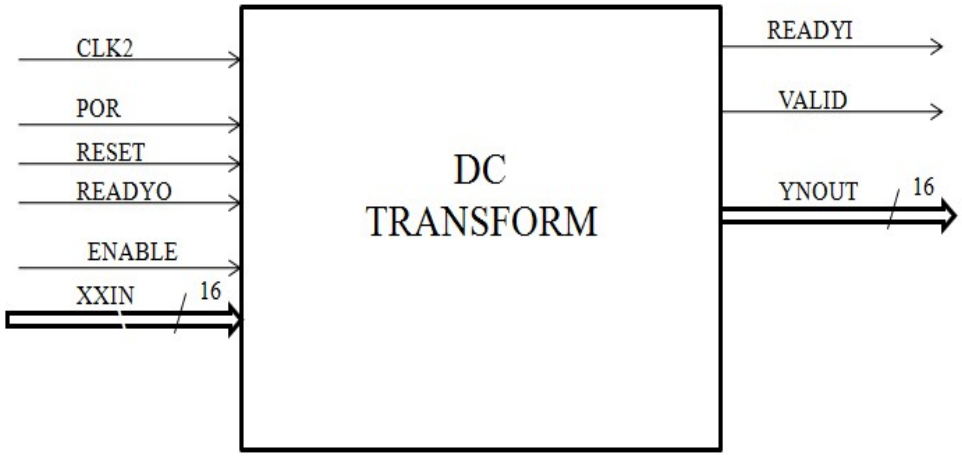


Fig. 3.5 Block diagram of the DC transform module

READYI, VALID and YNOUT are the output signals. When VALID becomes high, data at YNOUT indicate valid data. When READYI becomes high, it shows that the DC transform process of previous data is completed and this module is ready to process other data.

3.3.4 Quantization module

Fig 3.6 shows the quantization module. CLK, POR, QP, DCCI, ENABLE and YNIN are the input signals. The output of the DC transform module and the core-transform module is multiplexed. The output of this multiplexer becomes the input of the quantization module. Therefore, DCCI signal indicates that data at YNIN signal are

`

the output from the DC transform module. When ENABLE becomes high, data at YNIN show valid data. QP decides the step size.

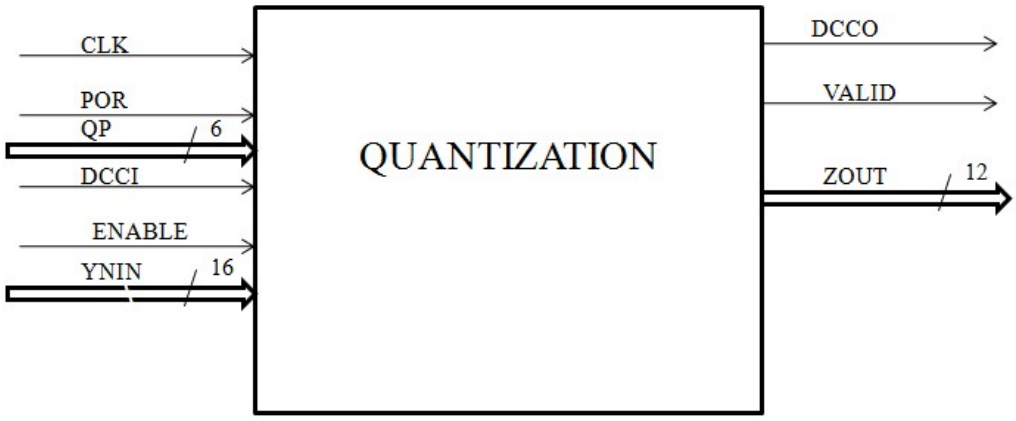


Fig. 3.6 Block diagram of the quantization module

Here, DCCO, VALID and ZOUT are the output signals in Fig. 3.6. When VALID becomes high, data at ZOUT show valid output data, which goes to the next module. When DCCO becomes high, the output data at ZOUT corresponds to the input of the DC module.

3.3.5 Inverse DC transform module

In this module, the input and output signals are the same as the DC transform module. The output of the DC transform module is quantized and this value is the input to the inverse DC transform module.

3.3.6 Dequantization module

Fig. 3.7 shows the block diagram of the dequantization module. CLK, POR, QP, DCCI, ENABLE, and ZIN are the input signals in Fig.3.7. When ENABLE becomes high, data at ZIN indicate valid data. The outputs of the inverse DC transform module and the quantization module go to multiplexer. The output of this multiplexer becomes the input of the dequantization module.

`

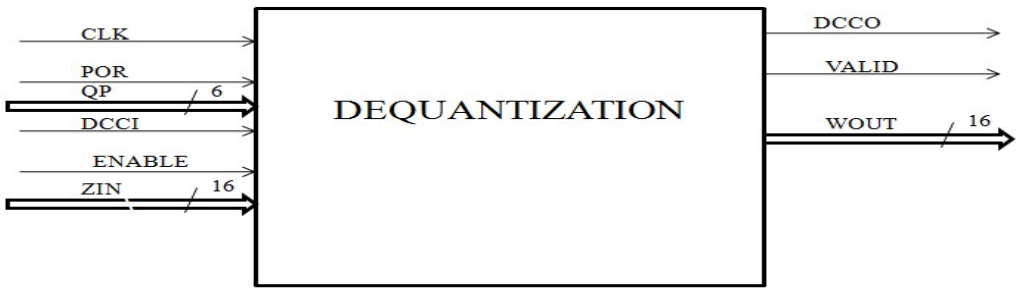


Fig. 3.7 Block diagram of the dequantization module

DCCO, VALID and WOUT are the output signals in Fig.3.7. When VALID becomes high, data at WOUT are valid data. When DCCO becomes high, data at WOUT indicate dequantized DC data.

3.3.7 Inverse core-transform module

Fig. 3.8 shows the block diagram of the inverse core-transform module. CLK, POR, ENABLE and WIN are the input signals in Fig. 3.8. When ENABLE is high, the input data at WIN are valid data. The input of this module is the output of the dequantization module.

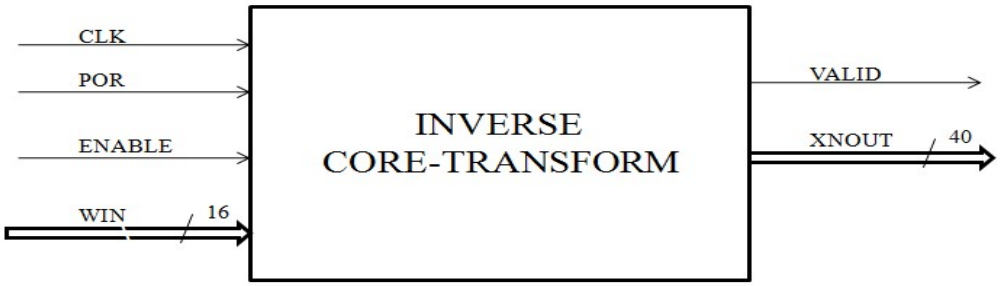


Fig. 3.8 Block diagram of the inverse core-transform module

The outputs of this module are VALID and XNOUT. When VALID becomes high, data at XNOUT are valid data, which go to the next module.

3.3.8 Reconstruction module

Fig.3.9 shows the reconstruction module with the input and output signals. CLK2, POR, NEWSLICE, STROBEI, BSTROBEI, DATAI, BCHROMAI and BASEI are the input signals in Fig. 3.9. Here, DATAI input come from the inverse core-

`

transform module. The STROBEI becomes high when data at DATAI are valid data. The BASEI input come from the inter module. The data at BASEI are the reference data, which are subtracted from the current data.

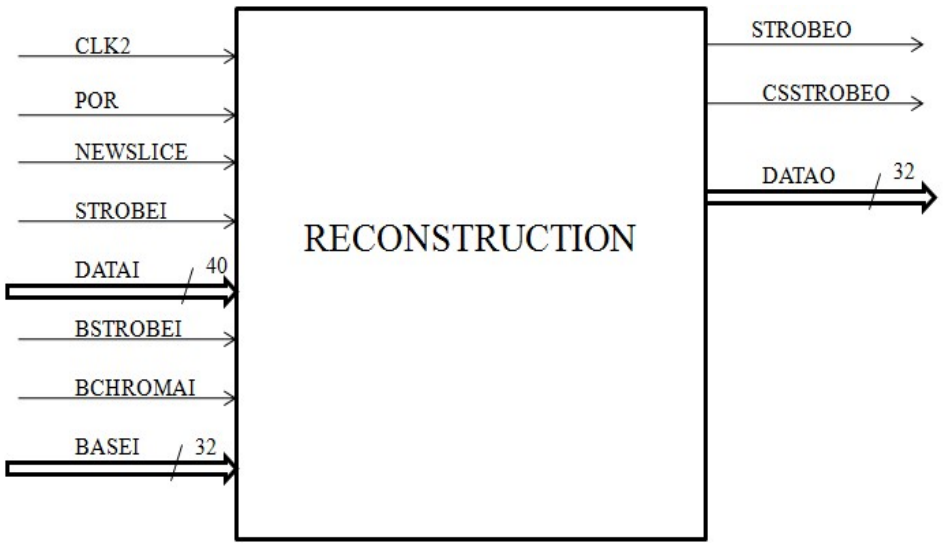


Fig. 3.9 Block diagram of the reconstruction module

When the data at BASEI is from the CHROMA module, BCHROMAI and BSTROBEI become high. BSTROBEI becomes high only when data at BASEI from the inter luma module. The NEWSLICE is used to reset the memory of the module. This memory used to store the data of BASEI.

The output signals of this module are STOBEO, CSTROBEO and DATAO. When CSTROBEO becomes high, the output data at DATAO is for chroma component. If STROBEO becomes high, output data at DATAO is for luma component.

3.3.9 Buffer module

Fig. 3.10 shows the block diagram of the buffer module. The buffer is used to rearrange the input format before giving to the CAVLC decoder. The input order for one macroblock is: luma0, chromadcA, chromaA0, luma1, luma2, chromaA1, luma3, luma4, chromaA2, luma5, luma6, chromaA3, luma7, luma8, chromadcB, chromaB0, luma9, luma10, chromaB1, luma11, luma12, chromaB2, luma13, luma14, chromaB3, luma15. Luma0 to luma 15, chromaA0 to chromaA3 and chromaB0 to chromaB3 are shown in Fig. 3.11.

`

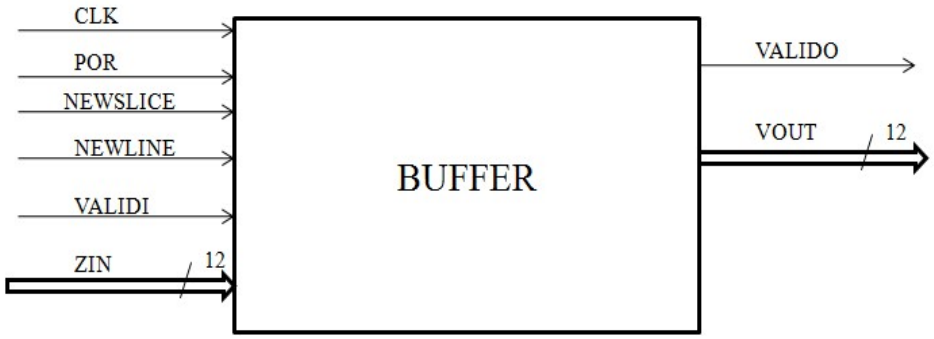


Fig. 3.10 Block diagram of the buffer module

The required output format for one macroblock is: luma0...luma15, chromadcA, chromadcB, chromaA0...3, chromaB0...3.

0	1	4	5
2	3	6	7
8	9	12	13
10	11	14	15

(a)

0	1
2	3

(b)

0	1
2	3

(c)

(a) Luma_compnent luma0, luma1---luma15 (b) Cr_component chromaA0---chromaA3 (c) Cb_component chromaB0---chromaB3

Fig. 3.11 Macroblock presentation of luma and chroma samples

POR, CLK, NEWSLICE, NEWLINE, VALIDI and ZIN are the input signals in Fig. 3.10. If VALIDI becomes high, data at ZIN are valid data. This data are stored in the memory of the buffer module. It arranges the input in the desired manner and gives the output at the desired manner. Here, VALIDO and VOUT are the output signals in Fig. 3.10. When VALIDO becomes high, data at VOUT are valid data, which are given to the next module.

3.3.10 Header module

The block diagram of the header module is shown in Fig. 3.12. Here, the input signals are CLK, POR, NEWSLICE, SINTRA, MINTRA, LSTROBE, MVDX_O, VE_X, MVDY_O, VE_Y, QP, PMODE, RMODE, PTYPE and PSUBTYPE in Fig. 3.12. This module is used to generate the different level header, which are discussed in the

`

previous chapter. SINTRA indicates the type of slice. MINTRA indicates the type of macroblock. PMODE and RMODE are used to store the different intra modes. The PTYPE and PSUBTYPE are used to code the size of macroblock and submacroblock, respectively. MVDX_O and VE_X are used to generate the motion vector for x-direction. MVDY_O and VE_Y are used to generate the motion vector for y-direction. These motion vector values come from the inter_luma_module. Here, the output signals are VALID, VE and VL. When VALID becomes high, data at VE and VL are valid data.

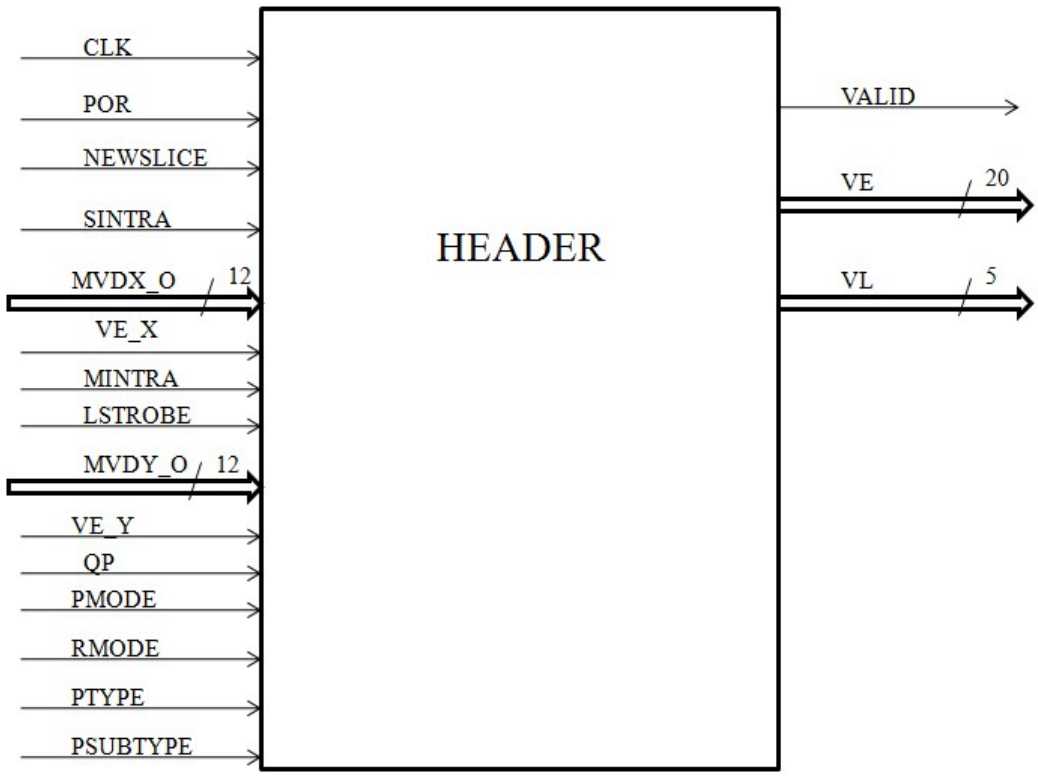


Fig. 3.12 Block diagram of the header module

The data at VE show the output header data. The values at VL show the number of valid bits in VE data. The output values go to the next tobytes module.

3.3.11 CAVLC module

CAVLC module is shown in Fig. 3.13. CLK, CLK2, POR, ENABLE, VIN and NIN are the input signals in Fig. 3.13. It takes the nonzero coefficients of the previous

`

block as an input NIN to encode the current macroblock coefficients. Therefore, it is context adaptive variable length coding.

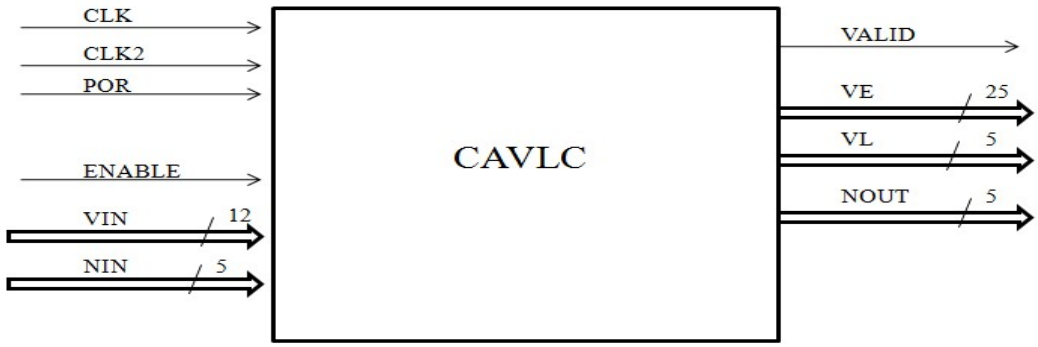


Fig. 3.13 Block diagram of the CAVLC module

When ENABLE becomes high, data at VIN are valid data. NOUT indicate the number of nonzero coefficients for this macroblock. When VALID becomes high, the output data at VE and VL are valid data. VE indicate the output encoded data and VL indicate the number of valid bits in VE.

3.3.12 Tobytes module

Tobytes module is shown in Fig. 3.14. The output of the CAVLC module (VL and VE) and the header module (VL and VE) are multiplexed. This multiplexed output becomes the input (VL and VE) of the tobytes module.

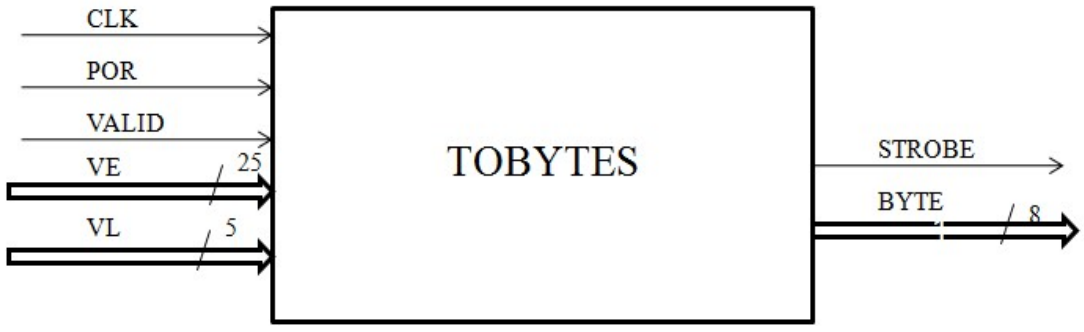


Fig. 3.14 Block diagram of the tobytes module

When VALID becomes high, the input data at VL and VE are valid data, which is converted to byte data. The data at VL shows the number of bits of VE, which should be converted to byte data. Here, BYTE is the output data in Fig. 3.14. When STROBE becomes high, data at BYTE signal is valid output data.

`

3.3 Calculation of minimum frequency of operation

The FPGA implemented H.264 algorithm can support up to 60 MHz frequency operations. The calculation of minimum frequency depends on the data-rate and the numbers of clock cycles are required for the processing of one macroblock. The Table 3.1 shows the video frame resolution, which are supported by this algorithm. The resolution of the supported video frame can be calculated by using the equation (3.1).

Table 3.1 Minimum frequency of operation according to datarate

Frmae_Width (FW)	Frame_Height (FH)	Fram rate (FR)	Data rate in Mbps for 4:2:0	Minimum frequency of operation in MHz
352	288	24	29.196288	9.161856
512	512	12	37.748736	11.845632
512	512	24	75.497472	23.691264
720	480	24	99.5328	31.2336

$$\text{Minimum frequency of operation (for 4: 2: 0)} = \frac{12 * \text{FW} * \text{FH} * \text{FR} * \text{clk}}{3072} \qquad (3.1)$$

where, FW, FH and FR indicate Frame_Width, Frame_Height and Frame rate, respectively. Clk shows number of required clock cycles for processing of one macroblock, which is set to 964. The number of bits in one macroblock is 3072.

`

CHAPTER - 4

SIMULATION RESULTS

The H.264 encoding algorithm and its hardware implementation are described in the previous chapter. The frequency is set to get the simulation result of the modules according to the minimum and the maximum frequency of operation, which is discussed in the previous chapter. These simulation waveforms of the different modules are discussed in this chapter. PSNR and compression ratio are discussed in this chapter.

4.1 Simulation waveforms

The simulation waveforms of the modules are shown here. These waveforms are captured using questa simulator and ISE 13.2 software.

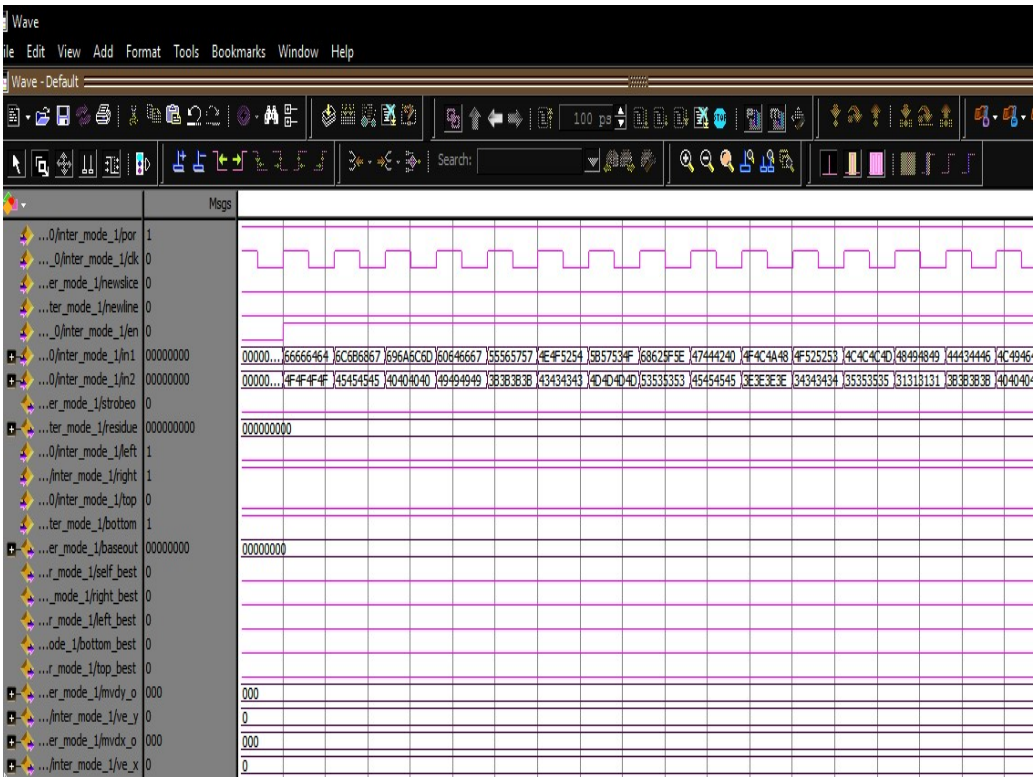


Fig. 4.1 Simulation waveform with the input data values of the inter prediction luma module

`

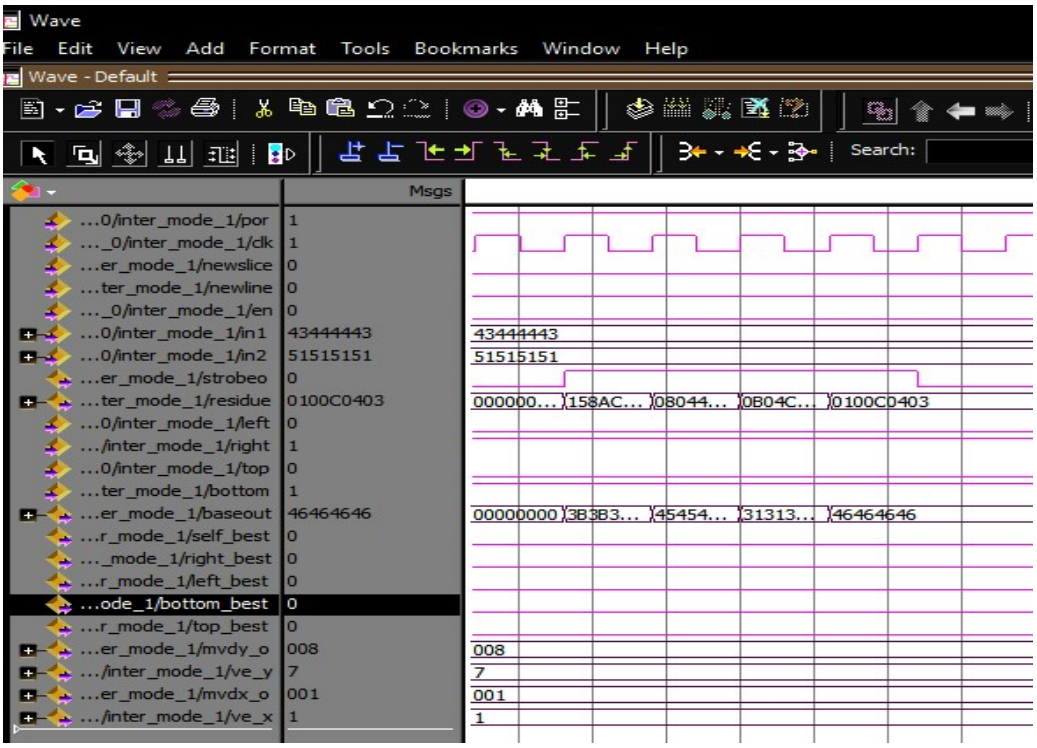


Fig. 4.2 Simulation waveform with the output data values of the inter prediction luma module

The input signals are por, clk, newslice, newline, en, in1, in2, left, right, top and bottom for the inter prediction luma module. The output signals are strobeo, residue, self_best, right_best, left_best, bottom_best, top_best, mvdx_o, ve_x, mvdy_o and ve_y for the inter prediction luma module. The inputs of a macroblock are shown in Fig. 4.1. The outputs of one submacroblock are shown in Fig. 4.2. The motion vector of x-direction and y-direction are shown for that macroblock in Fig. 4.2.

Here, four pixels are processed simultaneously. Each pixel denotes 8 bit. Therefore, four pixels denote 32 bit data. 32 bit data comes in one clock cycle. Therefore, 64 clock pulses are required to read one macroblock data. The width for residue signal is 36 bit. The width indicates the number of bits of a signal in one clock cycle.

`

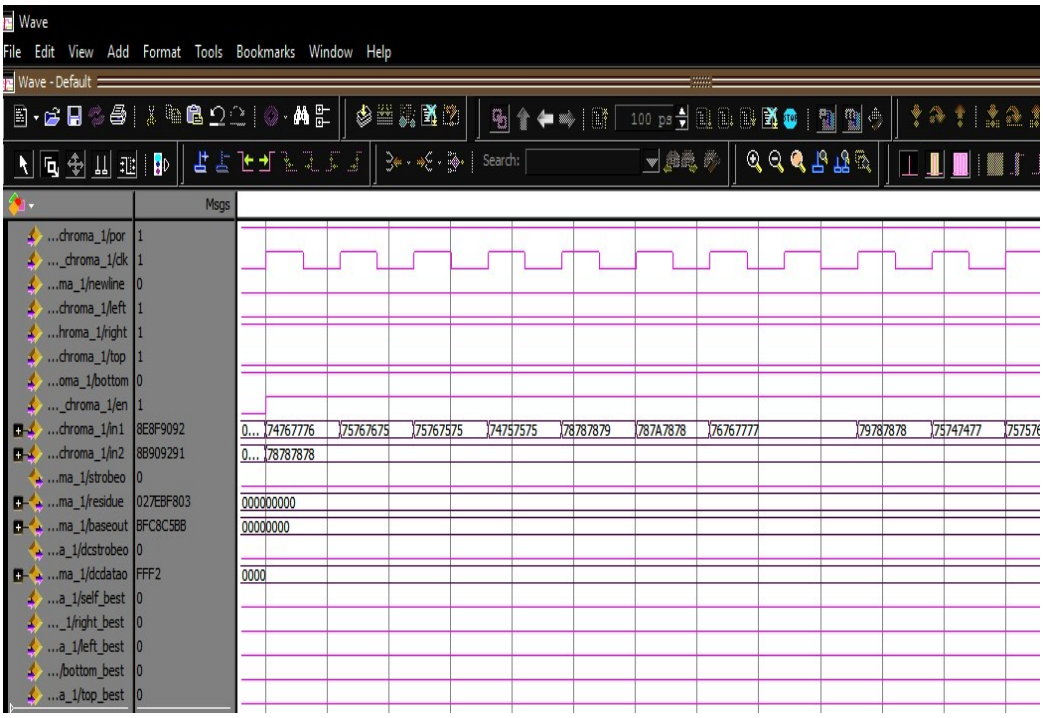


Fig. 4.3 Simulation waveform with the input data values of the inter prediction chroma module

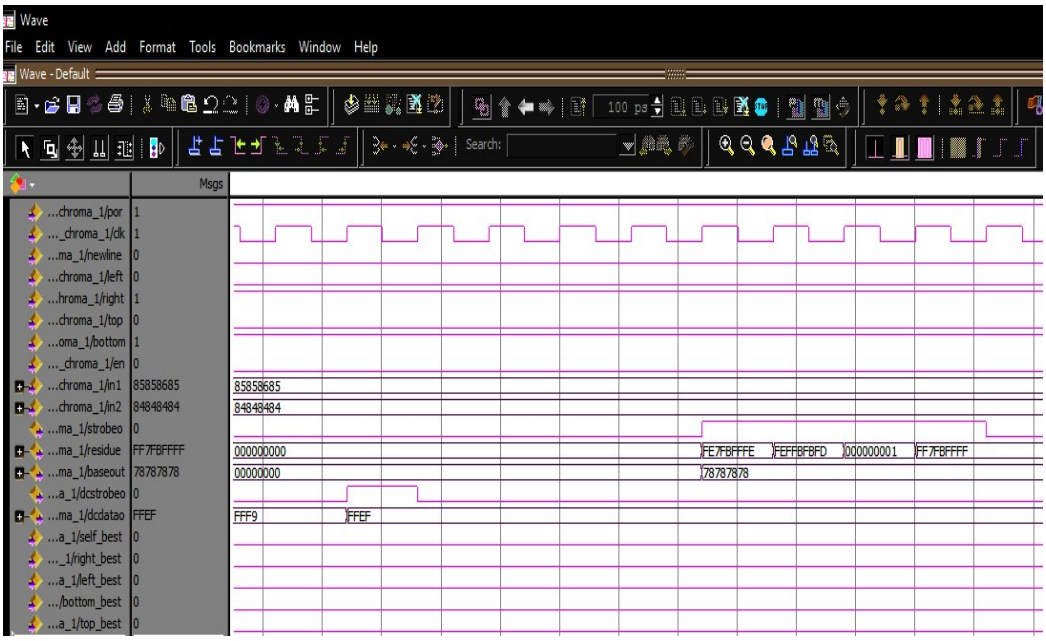


Fig. 4.4 Simulation waveform with the output data values of the residues and the dc data value for the inter prediction chroma module

`

The input data values are shown in Fig. 4.3. Fig. 4.4 shows the output for one submacroblock and a dcdatao value.

Here, 32 clock pulses are required to read a macroblock for the chroma module. The width of residue signal is 36 bit. 964 clock cycles are required to process one complete macroblock (luma and chroma components). The new data value come after 964 clock cycles of first input.

In the reverse path, the reconstruction module is used to reconstruct the data value. The input signals are CLK2, POR, NEWSLICE, STROBEI, DATAI, BSTROBEI, BCHROMAI and BASEI for the reconstruction module. The output signals are STROBEO, CSTROBEO and DATAO for the reconstruction module. Fig. 4.5 shows BASEI input, DATAI input and DATAO output for luma samples. Fig. 4.6 shows BASEI input, DATAI input and DATAO output for chroma samples.

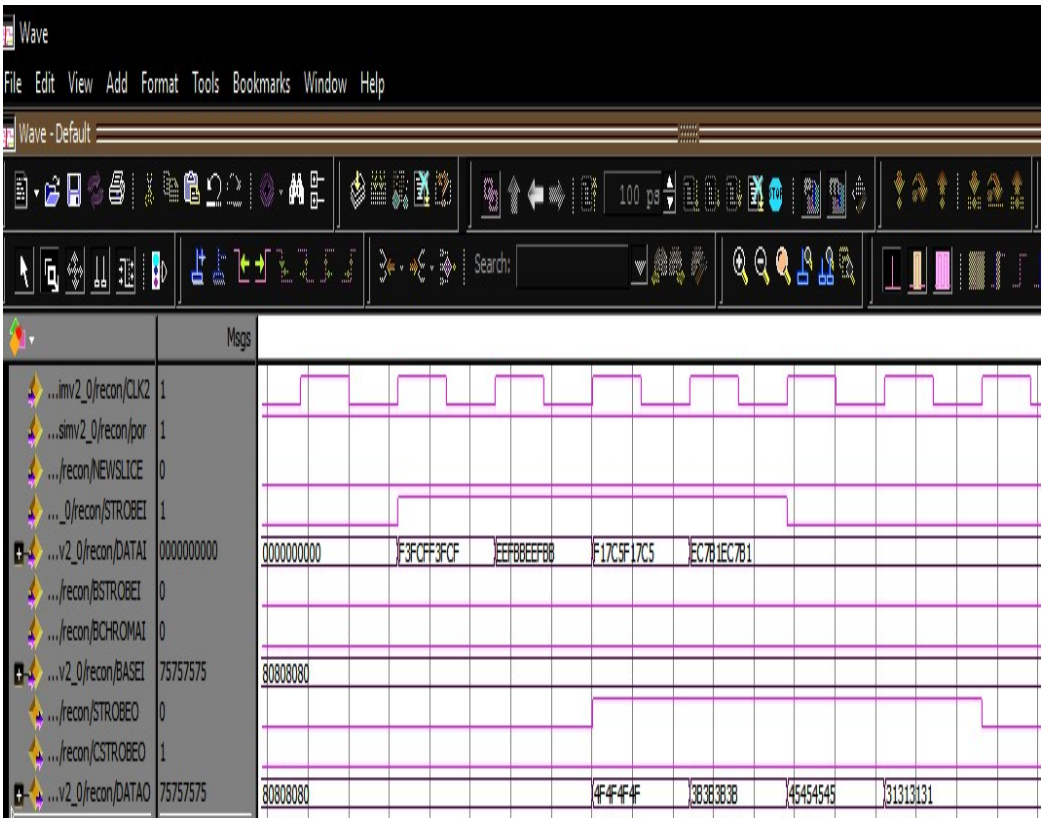


Fig. 4.5 Simulation waveform with the input and output data values of the reconstruction module for luma samples

`

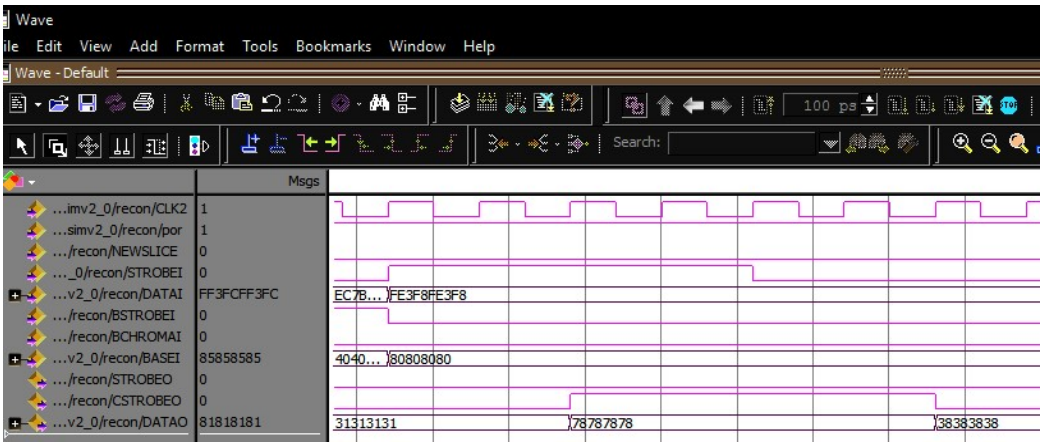


Fig. 4.6 Simulation waveform with the input and output data values of the reconstruction module for chroma samples

The width of BASEI signal and DATAI signal are 32 bit and 36 bit, respectively. The width of DATAO signal is 32 bit.

The output of the quantization module goes to the buffer module where the input is rearranged in the desired format. The output of the buffer module goes to the CAVLC module where the input data is encoded. The input signals are CLK, CLK2, POR, ENABLE, VIN and NIN for the CAVLC module. The output signals are VALID, VE and VL for the CAVLC module. Fig. 4.7 shows the input and output results of the CAVLC module.

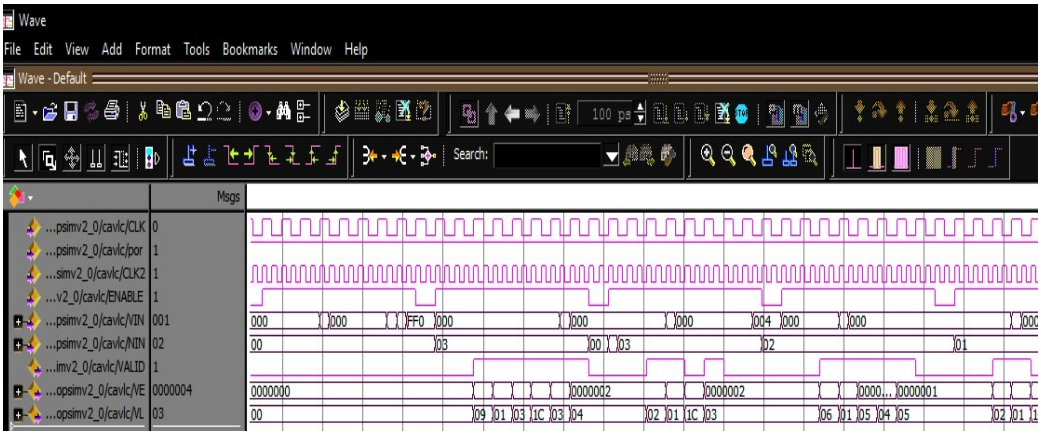


Fig. 4.7 Simulation waveform of the input and output signals of the CAVLC module

`

Fig. 4.8 Simulation waveform with the input data value of the CAVLC module

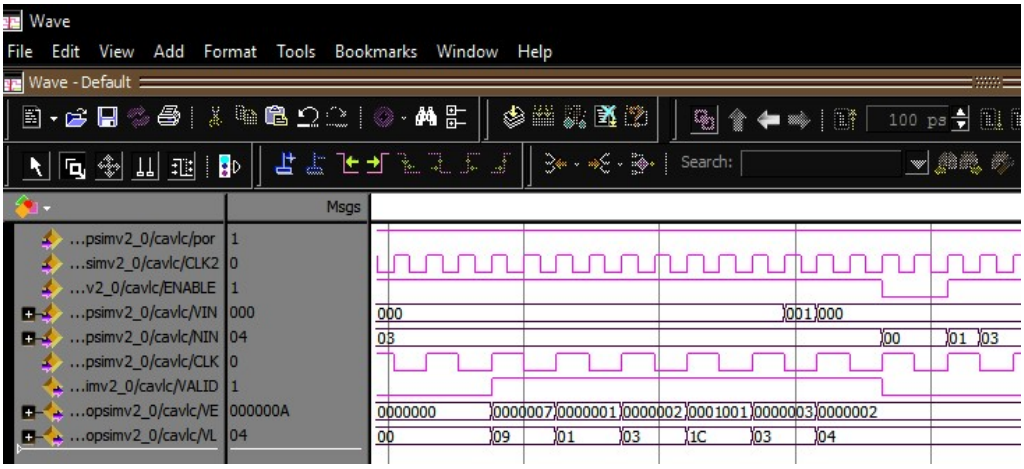


Fig. 4.9 Simulation waveform with the output data value of the CAVLC module

Fig. 4.8 shows the input of one submacroblock. Fig. 4.9 shows the output of a submacroblock. The CAVLC encoder requires minimum 9 clock cycles and maximum 18 clock cycles to encode a complete macroblock data. The required numbers of clock cycles depend on the input data values. The width of signal VE and VL are 25 bits and 5 bits, respectively.

The header module adds the information about video data such as frame height, frame width, QP value, etc.

`

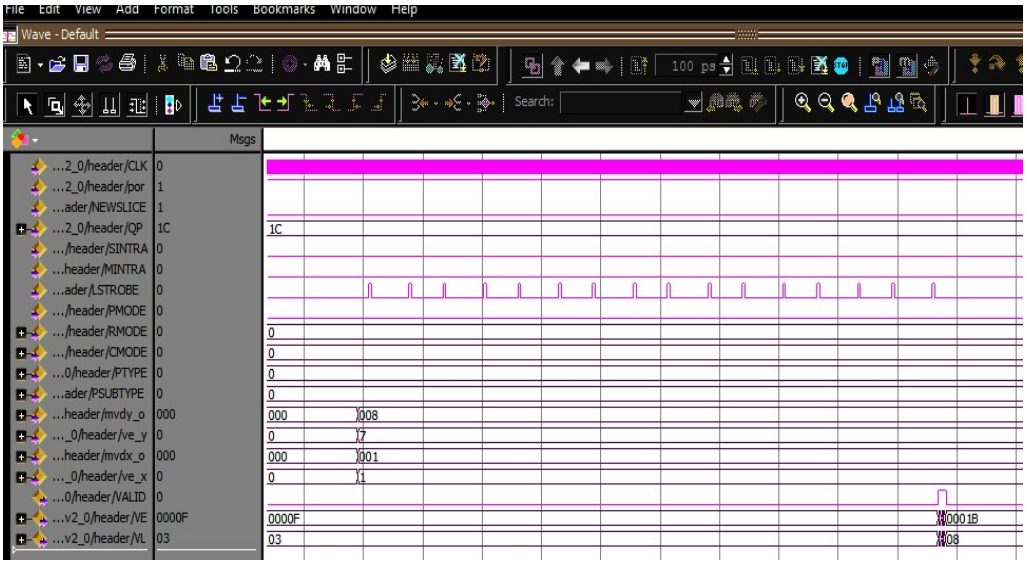


Fig. 4.10 Simulation waveform of the input and output of the header module

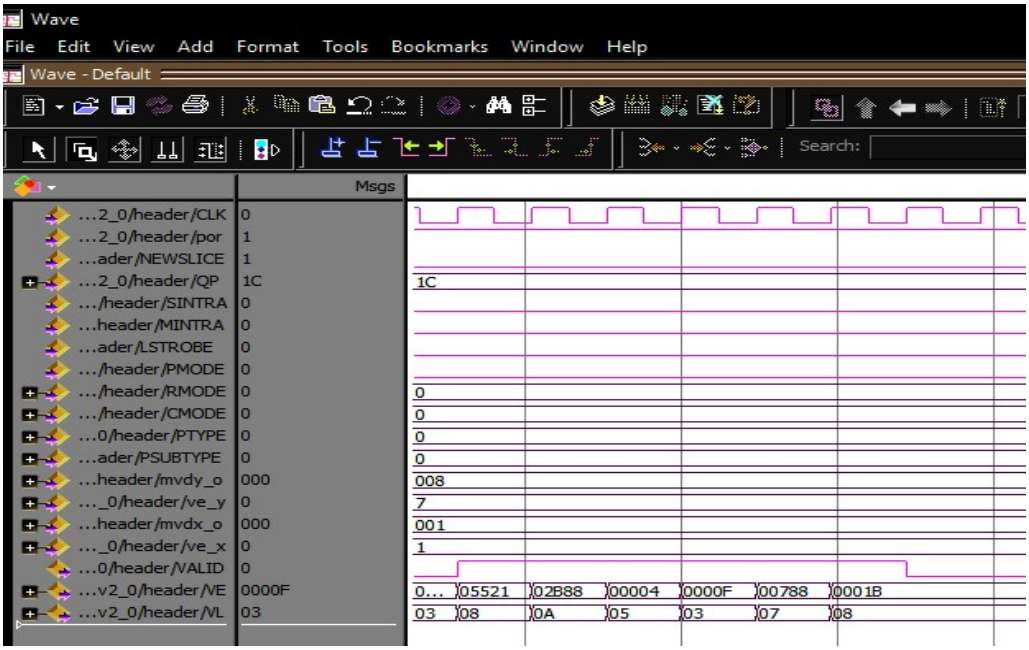


Fig. 4.11 Simulation waveform with the output data value of the header module

The input signals are CLK, POR, NWESLICE, QP, SINTRA, MINTRA, LSTROBE, PMODE, RMODE, CMODE, PTYPE, PSUBTYPE, MVDY_O, VE_Y, MVDX_O, and VE_X for the header module. The output signals are VALID, VE and VL for the header module. Fig. 4.10 shows the input and output for the header module. Fig. 4.11 shows the output of the the header module. The output from the CAVLC module and

`

the header module go to the tobytes module as shown in Fig. 4.12. The width of VE signal and VL signal are 25 bits and 5 bits, respectively.

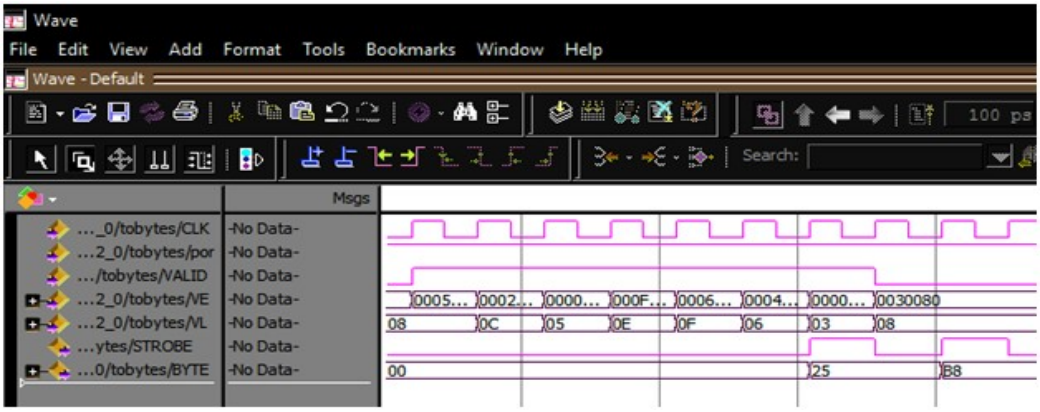


Fig. 4.12 Simulation waveform of the input and output of the tobytes module

The input signals are CLK, POR, VALID, VE and VL for the tobytes module. The output signals are STROBE and BYTE for the tobytes module. Fig. 4.12 shows the input and output of the tobytes module. Here, the input bits are converted to byte format. The output of the tobytes module is the final H.264 bit stream.

4.2 PSNR and compression ratio

Minimum one reference frame is required to implement the inter module If the reference frame is sent without any prediction, high compression ratio cannot be achieved. If the reference frame is sent using the intra prediction method, high compression ratio can be achieved. Therefore, the reference frame is sent using the intra prediction method. Here, the intra prediction frame to the inter prediction frame ratio is 1:10. Therefore, overall prediction is achieved due to the inter prediction method.

`

Table 4.1 Comparison of the output of H.264 hardware implementation and the output of JM software in terms of PSNR and compression ratio

TEST_ VIDEOS	H.264 HARDWARE			JM_SOFTWARE		
	PSNR (dB)		COMPRE-SSION RATIO	PSNR (dB)		COMPRE-SSION RATIO
	Luma	Chroma U/V		Luma	Chroma U/V	
akiyo_cif (352 × 288)	40	41.21/ 42.37	112.3	40.25	42.32/ 43.71	114.23
mother-daughter_cif (352 × 288)	39.2	42.97/ 43.61	93.58	39.33	43.33/ 44.17	94.18
Space_video (512 × 512)	43.22	44.75/ 44.93	53.2	36.5	39/ 38.3	40

Table 4.2 Comparison of the output of H.264 algorithm and the output of image compression algorithm in terms of PSNR and compression ratio

TEST_ VIDEOS	H.264 ALGORITHM			IMAGE COMPRESSION ALGORITHM		
	PSNR (dB)		COMPRE-SSION RATIO	PSNR (dB)		COMPRE-SSION RATIO
	Luma	Chroma U/V		Luma	Chroma U/V	
akiyo_cif (352 × 288)	40	41.2/ 42.3	112.3	40.47	42.46/ 43.77	19.97
mother-daughter_cif (352 × 288)	39.2	42.97/ 43.61	93.58	39.95	43.2/ 43.74	22.15
Space_video (512 × 512)	43.22	44.75/ 44.93	53.2	44	45.3/ 45.9	33.6

Table 4.3 Comparison of the output of H.264 with 15 fps and the output of H.264 with 30 fps in terms of PSNR and compression ratio

TEST_ VIDEOS	H.264 ALGORITHM					
	Frame rate : 30 fps			Frame rate : 15 fps		
	PSNR (dB)		COMPRE-SSION RATIO	PSNR (dB)		COMPRE-SSION RATIO
	Luma	Chroma U/V		Luma	Chroma U/V	
CITY_cif (352 × 288)	35.66	42.1/ 43.01	16.69	35.59	41.9/ 42.65	9.8
CREW_cif (352 × 288)	37.56	40.23/ 39.65	15.96	37.31	40/ 39.4	12.01

`

In Table 4.1, the PSNR and the compression ratio of the output of this FPFA implemented H.264 algorithm are compared with the PSNR and compression ratio of the output of JM software of H.264 to verify the algorithm. It is seen that the compression ratio and PSNR using both the modules are approximately the same. Therefore, the desired H.264 algorithm is successfully implemented on FPGA.

In Table 4.2, PSNR and compression ratio of the H.264 are compared with PSNR and compression ratio of the image compression algorithm. Here, image compression algorithm indicates H.264 algorithm with intra prediction frames only. All frames are predicted using the intra prediction algorithm.

In Table 4.3, fps indicates frames per second. In Table 4.3, PSNR and compression ratio of the H.264 with different frame rate are compared. Here, 15 fps and 30 fps are taken for comparison. This comparison shows that the number of fps increases, compression ratio increases, but PSNR decreases. This decrement in PSNR is not significant compared to increment in compression ratio. The inter prediction method gives more compression ratio if number of the fps increases.

`

CHAPTER - 5

CONCLUSION AND FUTURE WORK

The simulation results and comparison of PSNR and compression ratio are discussed in the previous chapter. In this chapter, conclusion and the future work of this project are presented.

5.1 Conclusion and future work

The main objective of this project is the implementation of real-time video compression standard to achieve the compression ratio in the range of 30 to 40 and good quality video with PSNR in the range of 30 to 40 dB for space applications. This aim is accomplished by using H.264 algorithm with the baseline profile. The system, which can compress the video of 512 × 512 frame size, is required for the space applications. This requirement is fulfilled by the implementation of H.264 algorithm. The algorithm is implemented on a single Xilinx FPGA of Virtex 5Q series successfully.

This project is targeted for the video of 512 × 512 and 352 × 288 frame size. This algorithm is also synthesized with 1024 × 1024 frame size video sequence but it was not implemented on the desired FPGA due to limitation of FPGA resources. In this project, external memory interface is not used. If the user wants to compress the video of 1024 × 1024 frame size, this project can be implemented with external memory interface. Some input and output ports require for accessing the external interface, which can be implemented because 600 input and output ports are available after implementation of this project. Therefore, the expansion of this project for the video of large frame size is possible. A different H.264 standard configuration can also be implemented with a higher profile and levels to get higher compression.

`

REFERENCES

[1] I. E. Richardson. "The H.264 Advanced Video Compression Standard, 2nd edition," *Vcodex limited, UK*.

[2] T. Wiegand, G. J. Sullivan, G. Bjonteguard and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 13, no. 7, July 2003.

[3] T.C. Chen, Y. W. Hung, C. Y. Tsai, " Architecture Design of Context-Based Adaptive Variable-Length Coding for H.264/AVC*," IEEE transactions on Circuits and Systems—ii: Express Briefs*, vol. 53, no. 9, September 2006

[4] N. Keshaveni, S. Ramachandran and Gurumurthy Kargal, "Implementation of Context Adaptive Variable Length Coder for H.264 Video Encoder," *International Journal of Recent Trends in Engineering*, vol. 2, No. 5, November 2009

[5] N. Keshaveni, S. Ramachandran and K. S. Gurumurthy, "Design and Implementation of Integer Transform and Quantization Processor for H.264 Encoder on FPGA," *International Conference on Advances in Computing, Control, and Telecommunication Technologies,* 2009.

[6] T. Y. Kuo and C. H. Chan, "Fast Variable Block Size Motion Estimation for H.264 Using Likelihood and Correlation of Motion Field**,"** *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 10, October 2006.

[7] Y.Q. Shi and H. Sun, "Image and video compression for multimedia engineering: Fundamentals, algorithms and standards," Boca Raton, FL: CRC Press, 2000.

[8] G. Raja and M. J. Mirza, "Performance Comparison of Advanced Video Coding H.264 Standard with Baseline H.263 and H.263+ Standards," *International Symposium on Communications and Information Technologies,* 2004 (ISCIT 2004), Japan, October 26- 29, 2004.

[9] Y. Zhang, W. Gao, Y. Lu, Q. Huang, and D. Zhao, "Joint source-channel rate-distortion optimization for H.264 video coding over error-prone networks*," IEEE Transactions on Multimedia*, vol. 9, no. 3, April 2007

`

[10] A. Ahmad, N. Khan and S. Masud, Maud, “Performance Evaluation of Advanced Features of H.26L Video Coding Standard,” proceedings *IEEE in Mic* 2003.

[11] Joint Video Team (JVT), “Editor’s Proposed Modifications to Joint Committee Draft (CD) of Joint Video Specification” (ITU-T Rec. H.264 I ISO/IEC 14496-10 AVC), 4th JVT meeting, Klagenfurt, Austria, 22-26 July, 2002.

[12] Xiph.org Video Test Media [derf's collection] https://media.xiph.org/video/derf/ (September 2016)

[13] Index ftp://ftp.tnt.uni-hannover.de/pub/svc/testsequences/ (September 2016)