Design and FPGA Implementation of Systolic Array Architecture for Full Search Block Matching Algorithm

S. Prakash[1], R. Selva Kumar[2], Vasudeva Murthy[3]
1 - M. Sc. [Engg.] Student, 2 - Senior Lecturer, 3 - Senior Lecturer, VLSI System Design Centre,
M.S. Ramaiah School of Advanced Studies, Bangalore 560 054

Abstract

In digital video coding applications, adjacent frames look similar and changes are due to the movement of the object or the camera. So their spatial and temporal redundancy must be exploited to obtain reduced data to store and transmit. The motion between two consecutive images can be estimated using Full Search Block Matching Algorithm (FSBMA). This algorithm determines motion by pixel-by-pixel basis. Systolic array architecture can be used to implement the Full Search Block Matching Algorithm due to their advantageous properties like regularity, modularity and local communication.

The FSBMA has been designed using two dimensional systolic array architecture. The blocks of this architecture are processing element, shift register arrays, address generators for search area value ,reference block value, enable signal generator, candidate region monitor and best match selection unit. All these blocks and the top module have been designed. In this work the two dimensional systolic array architecture with serial input is used to perform Full Search Block Matching Algorithm. The FSBMA using Systolic Array has been modeled in Verilog Hardware Description Language and simulated against its functional specifications. This is then synthesized using Xilinx synthesis tool with constraints and has been implemented on FPGA. The implemented design has been verified.

The system has a single clock and reset. The designed system has an image resolution of 176x144 and works at a frequency of 45.792MHz. It takes serial inputs and performs parallel processing to reduce IO pin counts. The design has been implemented on Spartan3E FPGA and the following resources have been utilized - 2095 out of 4656 slices (44%), 2816 out of 9312 slice flip-flops (30%) and 63 Input Output blocks (IOBs). The minimum period of one clock pulse is 21.838ns.

Key Words: Motion Estimation, Full Search Block Matching Algorithm, Systolic Array

Abbreviations

AG	Address Generators
BMSU	Best Match Selection Unit
CRM	Candidate Region Monitor
FPGA	Field Programmable Gate Array
FSBMA	Full Search Block Matching Algorithm
LUT	Look Up Table
MAD	Mean of Absolute Difference
ME	Motion Estimation
PE	Processing Element
QCIF	Quarter Common Intermediate Format
RSR	Reference Block Shift Register
SAV	Search Area Value

1. INTRODUCTION

Digital video codecs for video telephone and video teleconferencing systems needs to process video resolutions of Quarter Common Intermediate Format (QCIF) 176*144 or Common Intermediate Format (CIF) 352*288 with 20 or 25 frames per second. But encoding each frame separately generates a huge amount of digital data to be stored or transmitted. The bandwidth allocated for video conferencing is limited and will be inefficient for transmission of large amounts of data. So the spatial and temporal redundancy within the frame and between the consecutive frames can be used for efficient data transmission. Motion Estimation is used to find temporal redundancy between frames. Motion Estimation is mostly performed using Full Search Block Matching Algorithm. The FSBMA performs motion estimation on pixel-by-pixel basis. The basic video encoder which uses Motion Estimation is shown in figure 1.

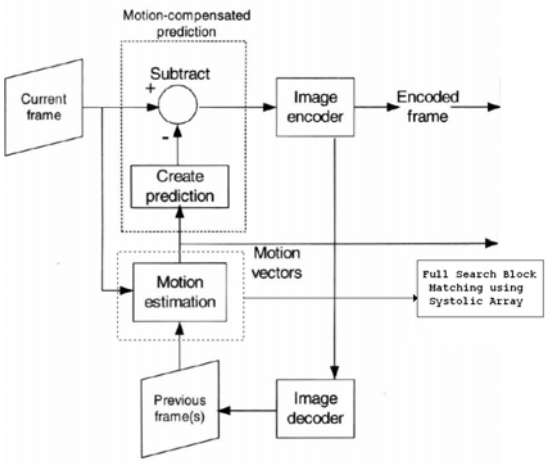

Fig. 1 Video encoder [1]

The motion estimation unit takes in the current frame and previously encoded and decoded frame as inputs and estimates motion between two frames and provides the motion vector as the best matching point. Full search block matching algorithm compares macro block with all the blocks in the search area of previous block. A systolic array is used here to perform FSBMA due to its similarity of processors, synchronous data flow and parallel processing. The figure 2 below

illustrates the block matching between two adjacent frames.

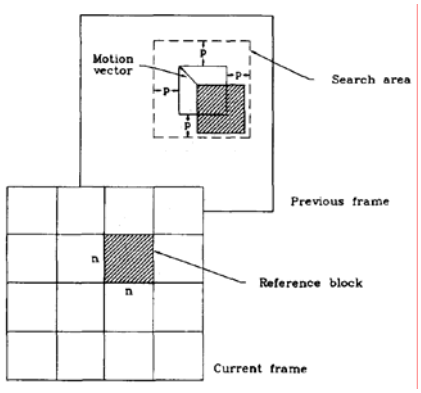

Fig. 2 Block matching process [2]

The block matching algorithm estimates the amount of motion on a block-by-block basis. In a typical block matching algorithm, a current frame is divided into a block of pixels, say, of size (n x n). The block of pixels (called a reference block) is compared with the corresponding blocks (called candidate blocks) within a search area of size (n+2p)*(n+2p) in the previous frame, where p is the maximum displacement allowed. The displacement (motion vector) can be obtained when the best match candidate block is found [2]. Among several possible searching methods the full search block matching algorithm (FSBMA) gives the optimal solution and the low control overhead. The basic idea of FBMA is to search all the possible displaced candidate blocks within the search area in the previous frame to find the best match block.

In Full Search Block Matching Algorithm, for each reference block, there are $(2p+1)^2$candidate blocks to be matched, and for each block matching, N^2 subtractions, N^2 magnitude operations and N^2 accumulations are needed. Obviously, the Full Search Block Matching Algorithm requires huge computations and cannot be performed in real time by a single processor. Recently, many general purpose processors, such as DSP's or multiprocessors, have been presented to overcome the computational bottleneck. However, some drawbacks exist in this way of implementation: cost of implementation and the difficulty in mapping the target algorithm into the general-purpose processors in an optimum way. The systolic array implementation is another attractive approach due to its advantageous properties like the regularity, modularity, and local communication.

Systolic Array architecture for implementing Full Search Block Matching Algorithm is presented. This system consists of three main pipeline stages constructed by a systolic array combined with shift register arrays, a parallel adder formed by a tree of adders and a best-match selection unit. The image data enter the first stage serially and are shifted in it. The systolic array calculates the absolute differences in parallel and then sends the partial sums of the absolute differences to the parallel adder to compute the mean absolute difference of each block matching. The mean absolute differences are fed serially into the best match selection unit to complete the computation. The proposed architecture has the following advantages it allows serial data inputs saving the pin counts but performs parallel processing, it is flexible in adaptation to the dimensional change of the search area via simple control, it can operate in real time for videoconference applications, and it is a simple and modular design and thus is suitable for FPGA implementation.

2. LITERATURE REVIEW

2.1 Introduction

Motion estimation reduces the energy in the motion-compensated residual frame and can dramatically improve compression performance. Motion estimation can be very computationally intensive and so this compression performance may be at the expense of high computational complexity [1].Out of the many algorithms like Full search Block Matching Algorithm, Three-step Search, Logarithmic search, Cross search, one at a time search and so on, In which only Full search block matching algorithms search for all possible displacement in search area and provides best match. Due to its high computational complexity systolic arrays can be used to perform these calculations, which will be discussed in the literature.

2.2 Motion Estimation

Motion estimation creates a model of the current frame based on available data in one or more previously encoded frames i.e. reference frames. These reference frames may be 'past' frames i.e. earlier than the current frame in temporal order or 'future' frames i.e. later in temporal order. The design goals for a motion estimation algorithm are to model the current frame as accurately as possible, since this gives better compression performance whilst maintaining acceptable computational complexity [1]. The basic idea of motion estimation is that the adjacent frames are similar and changes are due to object or camera motion [3], which is shown in the figure 3.

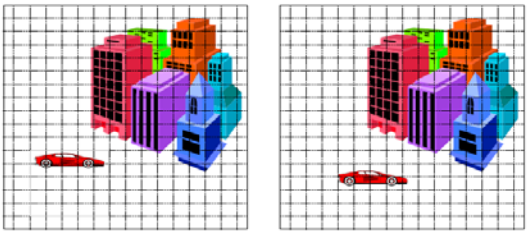

Fig. 3 Adjacent frames [3]

This is the temporal redundancy of adjacent frames that can be exploited to obtain reduced memory for storage and transmission. Motion estimation calculates the motion by comparing a block (usually called macro block of size 8*8 or 16*16) with the search area in the previous frame and calculates the estimation. The search area is area where the reference-block (which needs to be searched) and some adjacent pixels around the area are present. This is called motion compensated prediction.

2.3 Full Search Block Matching Algorithm

The full search block matching algorithm finds the best matching region in the previous frame. In theory it is necessary to carry out a comparison of the reference block with every possible region of the previous frame [3]. This is usually impractical because of the large number of comparisons required. In practice, a good match for the reference block (Macro Block) can usually be found in the immediate neighborhood of the block position in the previous frame if a match exists. Hence in practical implementations, the search for a matching region is limited to a 'search window', typically centered on the current block position. The optimum size of search window depends on several factors: the resolution of each frame (a larger window is appropriate for a higher resolution), the type of scene (high motion scenes benefit from a larger search window than low-motion scenes) and the available processing resources (since a larger search window requires more comparison operations and hence more processing). The full search block matching algorithm calculates the comparison criterion such as Mean of Absolute Differences (MAD) at each possible location in the search window. The example in figure 4 shows full search block matching algorithm.

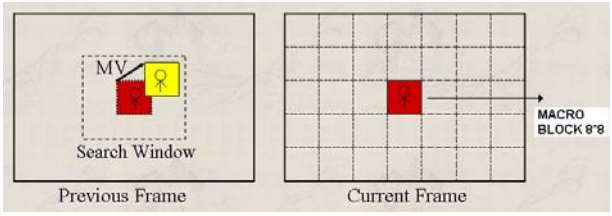

Fig. 4 Block matching motion estimation process [3]

The macro block is compared with all the possible matching positions in search area and finds the Mean of absolute differences, All the MAD values are compared and the position where we get minimum MAD value is the best matching position.
The MAD criterion is shown in equation 1,

$$MAD(u,v) = \sum_{i=1}^{n}\sum_{j=1}^{n} |S(i+u, j+v) - R(i,j)|, \quad -p \le u, v \le p \longrightarrow 1$$

where,
R(i,j) = reference block of size n*n at coordinates (i,j).
S(i+u,j+v) = candidate block within a search area in the previous frame
(u,v) = Candidate displacement vector.

The motion vector is determined by the least MAD (u,v) for all possible displacements within search area.

2.4 Systolic Arrays

A systolic array is an arrangement of processors in an array where data flows synchronously across the array between neighbors, usually with different data flowing in different directions [4]. Each processor at each step takes in data from one or more neighbors (e.g. North and West), processes it and, in the next step, outputs results in the opposite direction (South and East). Systolic arrays are specialized form of parallel computing, where processors are connected by short wires. An example of two dimensional systolic array is given in figure 5.

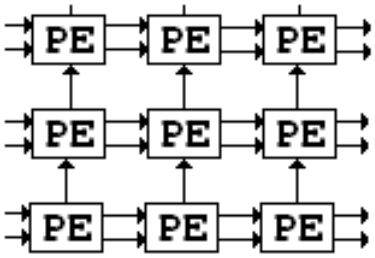

Fig. 5 Example of two dimensional systolic array [4]

The array takes in parallel inputs performs parallel processing and outputs the result. Systolic arrays do not loose their speed due to their connection unlike any other parallelism. Cells i.e. Processing Elements (PE), compute data and store it independently of each other. Each cell (PE) is an independent processor and has some registers and Arithmetic and Logic Units (ALUs). The cells (Processing Elements) share information with their neighbors, after performing the needed operations on the data. For example, when multiplying two 3*3 matrix we need N^3 operations according to the given formula

```
For I = 1 to N
   For J = 1 to N
      For K = 1 to N
         C[I,J] = C[I,J] + A[J,K] * B[K,J];
      End
   End
End
```

But using systolic arrays [4] it can be done in only 9 clock pulses.

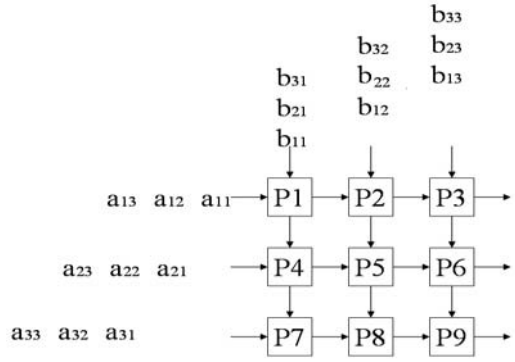

Fig. 6 Example of systolic array processing [4]

Example of Systolic Array is shown in figure 6. Here each cell takes in inputs from top and left, multiplies those two numbers and stores in the local register which is inside each Processing Element. After 9 clock pulses the result would be stored in each processing elements. In the full search block matching it needs N^2 subtractions, N^2 magnitude operations and N^2 magnitude accumulations are needed. Hence systolic arrays can be used to perform these operations due to their advantageous properties like regularity, modularity and local communication.

3. DESIGN OF BUILDING BLOCKS

3.1 Introduction

The Systolic Array architecture for full search block matching algorithm has number of sub modules and the top module. The architecture and design of all these sub modules and top module is discussed. According to the design specification it needs 8*8 processing elements, address generators for macro block (reference block) and search area value, one parallel adder, enable signal generator, candidate region monitor and a best match selection unit. Architecture and design of all these parts is described in this section.

3.2 Design of Processing Element

The architecture of processing element is shown in figure 7.

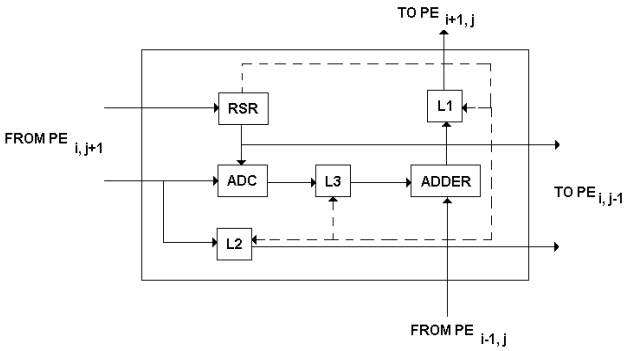

Fig. 7 Architecture of processing element [2]

Each PE consists of a Reference Block Shift register, an absolute difference calculator, adder and shift registers L1, L2 and L3. The reference block shift register, registers L2 and L3 are 8 bit parallel in parallel out shift register. The shift register L1 is a 15 bit shift register. Each PE is divided into a two stage pipeline by register L3. The first stage is constructed by RSR (reference block pixel register) and ADC (absolute difference calculator) and the second one by the adder. ADC computes the absolute difference (AD) between the search pixel datum shifted from the left-neighbour PE or from SRA (shift register array) and the reference pixel R(i, j) stored in the RSR. After being delayed one clock period by the register L3 the AD value is added to the result shifted from the down-neighbour PE and the sum is delayed one clock period by the register L1 and then passed to the up-neighbour PE. The search pixel datum shifted from left PE is delayed one clock period by the register L2 and then passed to the right neighbour or SRA (shift register array) [2].

3.3 Shift Register Array

For the two dimensional systolic array architecture the length of shift register array should be 2P-1, where P is maximum displacement in the search area, hence here the length of shift register array is 2*8-1=7 [2]. The structure of shift register array is given in figure 8.

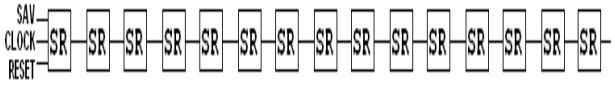

Fig. 8 Shift register array [2]

Each SR shown in Figure 8 is an 8 bit Parallel In Parallel Out shift register which is used to shift the search area value and make 15 clock pulse delay.

3.4 Parallel Adder

The parallel adder takes in inputs from first row of Processing Elements (PE) and outputs the result to best match selection unit. It has eight 15 bit inputs and one 18 bit output. It takes one clock pulse to output the result.

3.5 Enable Signal Generator

The enable signal generator is to enable the RSR (reference block shift register) for first 64 pulses to take in the inputs and store it on RSR. After 64 pulses it will be disabled i.e. en=0 for 514 pulses when absolute difference calculations and addition is performed and MAD values are calculated and given to best match selection unit.

3.6 Address Generator for RSR and SAV

The address generator for Reference block shift register and Search Area Value generates address in a way that it takes in the values from memory properly and inputs to the design unit (Processing Element).

The post Place & Route Simulation result is shown in the Figure 8. It calculates motion vector output if the enable signal from candidate region monitor is one and parallel adder value is lesser than previous value, output will be address value of search area – 1242. The best match selection unit (i.e. output of the FSBMA) value will be kept same if the parallel adder value is greater than previous value.

4. RESULTS AND DISCUSSION

4.1 Synthesis Results

Hardware utilization: The design is synthesized for and implemented in Spartan3E FPGA.

The post place & route simulation result is shown in the figure 8. It calculates motion vector output if the enable signal from candidate region monitor is one and parallel adder value is lesser than previous value, output will be address value of search area – 1242. The best match selection unit (i.e. output of the FSBMA) value will be kept same if the parallel adder value is greater than previous value.

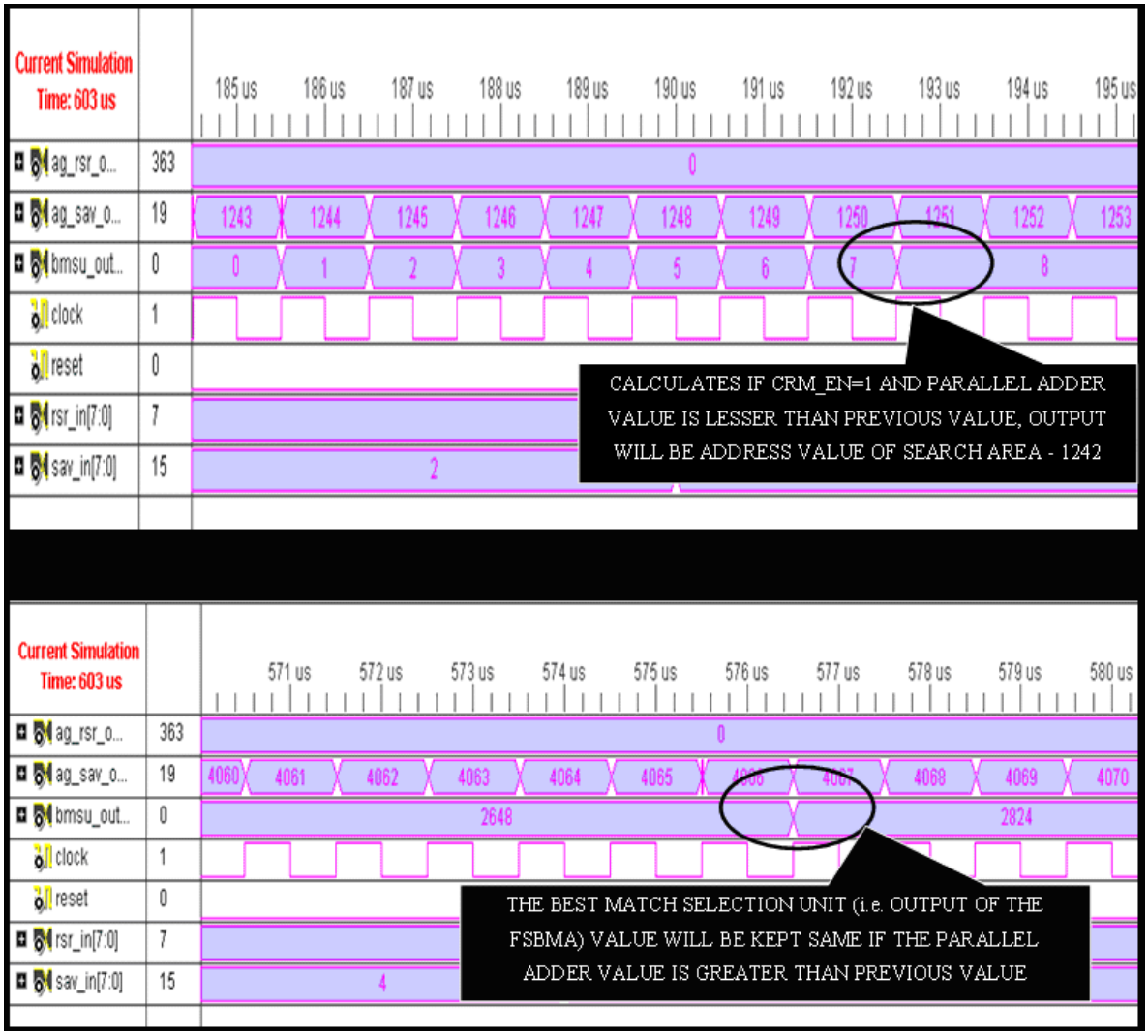

Fig. 8 Post place and route simulation result

Hardware utilization: The design is synthesized for and implemented in Spartan3E FPGA.

Number of Slices:	2095 out of 4656 44%
Number of Slice Flip Flops:	2816 out of 312 30%
Number of 4 input LUTs:	2993 out of 9312 32%
Number used as logic:	2913
Number used as Shift registers:	80
Number of IOs:	63
Number of bonded IOBs:	63 out of 190 33%
Number of GCLKs:	1 out of 24 4%

Table 1 Post place and route timing report

Constraint	Check	Worst Case Slack	Best Case Available	Timing Errors	Timing Score
OFFSET = OUT 15 ns AFTER COMP "clock"	MAXDELAY	2.946ns	12.054ns	0	0
OFFSET = IN 15 ns BEFORE COMP "clock"	SETUP	8.305ns	6.695ns	0	0
TS_clock = PERIOD TIMEGRP "clock" 40 ns HIGH 50%	SETUP HOLD	18.162ns 0.697ns	21.838ns	0	0

Synthesis Report: Minimum period: 16.998 ns (Max. Freq: 58.830 MHz) Minimum input arrival time before clock: 8.686 ns Maximum output required time after clock: 7.671 ns

After applying timing constraint and then synthesizing it produced better timing results. The Post Place & Route Timing Report is shown in table 1.

Post Place & Route Timing Report:

Minimum period: 21.838 ns (Max. Freq: 45.792 MHz)
Minimum input required time before clock: 6.695 ns
Minimum output required time after clock: 12.054 ns

5. CONCLUSION

The systolic array architecture for full search block matching algorithm has been modeled in Verilog Hardware Description Language and simulated against its functional specifications. It is then synthesized with constraints and implemented on FPGA. The implemented design has been verified against its functional specifications. It operates at a maximum frequency of 45.792 MHz. The resolution of the frame is 176 x 144. It takes inputs serially to reduce IO pin counts and processes data in parallel. The address generators are provided to accept inputs from memory

and process data. The best match selection unit outputs the address value which is the top left corner of the best match position for the present reference block.

REFERENCES

[1] Iain E. G. Richardson, "*Video Codec Design*", John wiley & Sons, Ltd

[2] Chaur-Heh Hsieh and Ting-Pang Lin, "*VLSI Architecture for Block-Matching Motion Estimation Algorithm*", IEEE Transactions on circuits and systems for video technology, vol. 2, no. 2, June 1992

[3] Yao Wang, "*Motion Estimation for Video Coding*", Yao Wang Politechnique University, Brooklyn

[4] Jonathan Break, "*Systolic Arrays & Their Applications*"

[5] Mohammed Mahdi Azadfar, "*Implementation of A Optimized Systolic Array Architecture for FSBMA using FPGA for Real Time Applications*", IJCSNS International Journal of Computer Science and Network Security, vol.8 No.3, pp. 46-51, March 2008

[6] T. Shanableh and M. Ghanbari, "*Heterogeneous video transcoding tolower spatio-temporal resolutions and different encoding formats*", IEEE Transaction on Multimedia, vol.2, no.2, pp. 101–110, Jun. 2000

[7] L. De Vos and M. Stegherr, "*Parameterizable VLSI architectures for the full-search block-matching algorithm*", IEEE Transaction on Circuits and Systems, vol. 36, pp. 1309–1316, Oct. 1989

[8] R. Srinivasan and K. R. Rao, "*Predictive coding based on efficientmotion estimation*", IEEE Transaction on Comm., vol. 2, no. 2, pp. 888–896,1985

[9] M. Mohammadzadeh, M.Eshghi and M.M.Azadfar, "*Parameterizable Implementation of Full Search Block Matching Algorithm using FPGA for Real-time Applications*", IEEE ICCDCS, pp.200-203, NOV.2004

[10] Samir Palnitkar, "*Verilog HDL, A Guide to Digital Design and Synthesis*", Pearson Education, ISBN: 81-7758-918-0

[11] Trac D. Tran, "*Video Coding: Block Motion Estimation*", The Johns Hopkins University, Baltimore.