A QUARTER PEL FULL SEARCH BLOCK MOTION ESTIMATION ARCHITECTURE FOR H.264/AVC

Choudhury A. Rahman and Wael Badawy
Laboratory for Integrated Video Systems (LIVS)
University of Calgary
Calgary, Alberta, Canada T2N 1N4
{rahman, badawy}@livs.ca

ABSTRACT

This paper presents a novel quarter pel full search block motion estimation architecture for H.264/AVC encoder. The proposed architecture is capable of calculating all 41 motion vectors required by the various size blocks, supported by H.264/AVC, in parallel. The architecture has been prototyped in Verilog HDL, simulated and synthesized for Xilinx Virtex2 FPGA. The experimental result shows that the architecture is capable of processing CIF frame sequences in real time considering 5 reference frames within the search range of -3.75 to +4.00 at a clock speed of 120MHz. The maximum speed of the architecture is around 150MHz.

1. INTRODUCTION

The newest international video coding standard has been finalized in May 2003. It is approved both by ITU-T as Recommendation H.264 and ISO/IEC as International Standard 14496-10 (MPEG-4 part 10) Advanced Video Coding (AVC) [1]. This new standard H.264/AVC is designed for application in the areas such as broadcast, interactive or serial storage on optical and magnetic devices such as DVDs, video-on-demand or multimedia streaming, multimedia messaging etc. over ISDN, DSL, Ethernet, LAN, wireless and mobile networks. Some new features of the standard that enable enhanced coding efficiency by accurately predicting the values of the content of a picture to be encoded are variable block-size, quarter-sample-accuracy and multiple reference picture for motion estimation and compensation [2]. In addition to improved prediction methods, other parts of the design are also enhanced for improved coding efficiency including small block-size transform, hierarchical block transform, exact-match inverse transform, arithmetic entropy coding etc. While the scope of the standard is limited to the decoder by imposing restrictions on the bitstream and syntax, and defining the decoding process of the syntax elements such that every decoder conforming to the standard will produce similar output when given an encoded bitstream that conforms to the constraints of the standard, there is a considerable flexibility in designing an encoder for AVC to optimize implementations in a manner appropriate to the intended application.

The new features such as variable block-size, quarter-sample-accuracy and multiple reference frames increase the complexity and computation load of motion estimation greatly in H.264/AVC encoder. Experimental results have shown that motion estimation can consume 60% for 1 reference frame to 80% for 5 reference frames of the total encoding time of H.264 codec [3]. Due to this reason, in order to get real time performance (30 frames per second) from a H.264 encoder, parallel processing must be exploited in the architecture. So far, there have been a very few VLSI implementations [4,5] for H.264/AVC motion estimation considering variable block size. But none of them is particularly suitable considering real time frame processing, multiple reference frames and fractional pel accuracy. In this paper, a quarter pixel full search variable block motion estimation architecture has been proposed that can process all the required motion vectors for H.264/AVC encoder in parallel. Experimental results have shown that the architecture can process in real time upto 5 reference frames at a clock speed of 120MHz.

The rest of the paper is organized as follows: Section 2 discusses some background of motion estimation. Section 3 presents the proposed architecture. Section 4 shows the simulation and synthesis results. Finally, section 5 concludes the paper.

2. BACKGROUND

Motion estimation is the basic bandwidth compression method adopted in the video coding standards. In H.264/AVC the motion estimation method is further refined with the new features like variable block size, multiple reference frames and quarter pixel accuracy.

0-7803-9332-5/05/$20.00 ©2005 IEEE

Upto 5 reference frames can be used along with 7 block patterns: 16x16, 16x8, 8x16, 8x8, 8x4, 4x8 and 4x4 in AVC as shown in figure 1. Compared to fixed size block and singe reference frame, the new method provides better estimation of small and irregular motion fields and allows better adaptation of motion boundaries resulting in a reduced number of bits required for coding prediction errors.

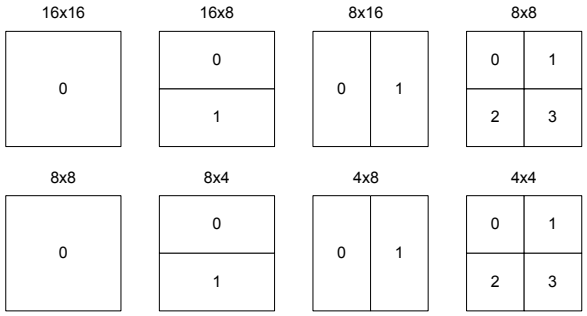

Figure 1. The various block sizes in H.264/AVC

The block matching algorithm (BMA) is the most implemented one in real time for motion estimation [6]. The algorithm is composed of two parts: matching criterion and searching strategy. In our proposed architecture, sum of absolute difference (SAD) and full search (FS) have been chosen for matching criterion and search strategy, respectively. SAD can be expressed in terms of equation as follows:

$$SAD(dx,dy)=\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left|a(i,j)-b(i+dx,j+dy)\right| \quad (1)$$

$$(MV_x,MV_y)=(dx,dy)\Big|_{\min SAD(dx,dy)}$$

In equation (1), $a(i,j)$ and $b(i,j)$ are the pixels of the reference and candidate blocks, respectively. dx and dy are the displacement of the candidate block within the search window. MxN is the size of the reference block and (MV_x, MV_y) is the motion vector pair of the block.

3. THE PROPOSED ARCHITECTURE

The proposed architecture for quarter pel full search block motion estimation is shown in figure 2. The architecture composed of single port block RAMs for search window and 16x16 reference block, 8 processing units, shift registers comparing unit and address generator (AG). The search window size of 92x92 pixels (quarter pel) has been chosen for prototyping for which the motion vector for the 16x16 size block lays between -3.75 to +4.00. So, there are 23x23 integer pel positions for which there are 64(8x8) 16x16 candidate blocks. Therefore, the total number of 16x16 candidate blocks considering quarter pel accuracy is 64x4x4 = 1024. The search window has been partitioned into 23 4x92 size block RAMs for parallel processing. This is shown in figure 3 and it requires a total memory bandwidth of 184(23x8) bits. The address generator generates addresses for the reference and candidate blocks. These addresses are fed into the search window memory, reference block memory and comparing unit.

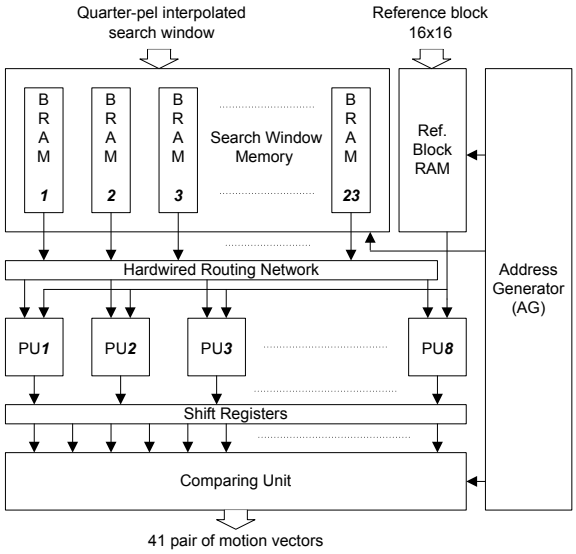

Figure 2. The proposed motion estimation architecture.

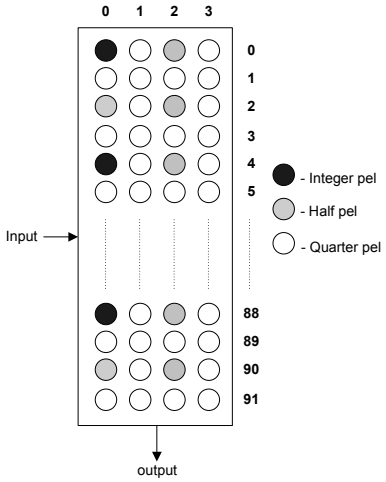

Figure 3. BRAM for search window memory.

The hardwired routing network connects the search window memory with the PUs. The input / output connections of the routing network are shown in table 1. Figure 4 shows the C-type address generation algorithm for the AG. This algorithm generates addresses for the search window memory (SW MEM), reference block memory (REF MEM) and Hx, Vx for the processing units

Table 1. Hardwired routing network's input / output connections.

Input (BRAM)	Output							
	PU1	PU2	PU3	PU4	PU5	PU6	PU7	PU8
1	1							
2	2	1						
3	3	2	1					
4	4	3	2	1				
5	5	4	3	2	1			
6	6	5	4	3	2	1		
7	7	6	5	4	3	2	1	
8	8	7	6	5	4	3	2	1
9	9	8	7	6	5	4	3	2
10	10	9	8	7	6	5	4	3
11	11	10	9	8	7	6	5	4
12	12	11	10	9	8	7	6	5
13	13	12	11	10	9	8	7	6
14	14	13	12	11	10	9	8	7
15	15	14	13	12	11	10	9	8
16	16	15	14	13	12	11	10	9
17		16	15	14	13	12	11	10
18			16	15	14	13	12	11
19				16	15	14	13	12
20					16	15	14	13
21						16	15	14
22							16	15
23								16

fed into the comparing unit. Each (H*x*, V*x*) pair represents motion vector and it addresses the top left corner point of a 4x4 candidate block shown in figure 5. This means the motion vectors of all the possible size blocks can be represented by the combination of these Hx, Vx values.

```
For c = 0 to 31 {
  For add_h = 0 to 3 {
    For add_v = 0 to 15 {

      SW MEM address = c + add_v*4 + add_h*92;
      REF MEM address = add_v;

      Hx for PU(y) = (add_h + x*16 + y*4 - 15)/4;
      Vx for all PU = (c + x*16 - 15)/4;
        //where, x = {0, 1, .. 3} and y = {0, 1, .. 7}
} } }
```

Figure 4. Algorithm for address generator.

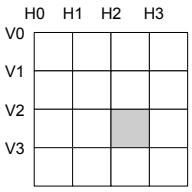

Figure 5. 4x4 blocks within a 16x16 candidate block and their corresponding addresses. For an example, the address of the gray shaded 4x4 block is (H2, V2).

The PU structure is shown in figure 6. It has 16 processing elements (PE) shown in figure 7. The PE is composed of one subtractor, one selectable adder / subtractor and two registers. The subtractor subtracts the values of the candidate and reference block pixels. The MSB of the result of this subtractor selects the functionality of the adder / subtractor unit. If the result of the subtractor unit is negative (MSB = 1), the adder / subtractor unit subtracts the result from the value stored in register R1 and vice versa. After each 4th cycle the accumulated value is loaded into R2 and R1 value is cleared. This means the output of each group of 4 PEs (16 PEs arranged in 4 groups) after summation of the PE outputs in that group gives the SAD value of a 4x4 block.

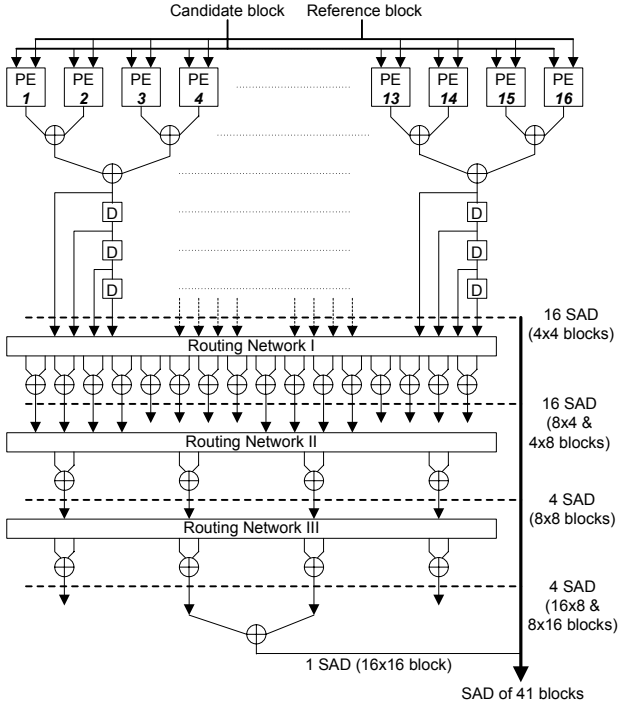

Figure 6. Processing unit (PU).

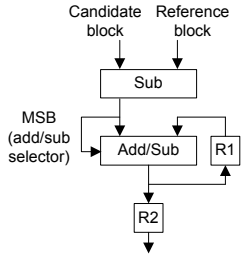

Figure 7. Processing element (PE).

These SAD values are passed through delay registers (D) that are triggered in every 4th cycle. Therefore, after the 16th cycle the SAD values of all the 4x4 candidate blocks are available to the inputs of the routing networks. The routing networks I, II and III are then used to connect these inputs to the four stage adder networks for computing the SAD values of the candidate blocks of other sizes, i.e., 8x4, 4x8, 8x8, 16x8, 8x16 and 16x16.

Therefore, 8 PUs compute all 41 SAD values of 8 16x16 candidate blocks of one row in parallel for each *add_h* value (figure 4). This means, each complete cycle of *add_h* values results all SAD values of 8x4 = 32 16x16 candidate blocks of one row. This is repeated 32 times, controlled by the value of *c* (figure 4) to complete motion estimation of the entire search window. *add_v* in figure 4 controls the row addresses of the reference and candidate block.

There are 41 parallel in serial out shift registers, one of which is shown in figure 8. Each of these takes SAD values of one particular type / size of block from all PUs as inputs and makes them serially available to the comparing unit.

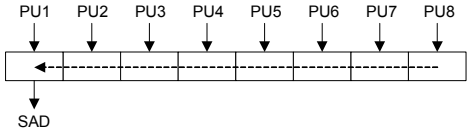

Figure 8. Parallel in serial out shift registers.

The comparing unit is composed of 41 comparing elements (CE), one of which is shown in figure 9. Each shift registers output is connected to one of these CEs. CE is composed of one comparator and two registers, one of which stores the minimum SAD for comparison and the other is triggered for storing the motion vector (H*x*, V*x*) from AG when the input SAD is less than the previous stored minimum SAD value.

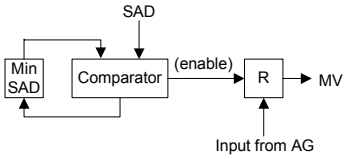

Figure 9. Comparing element (CE).

The min SAD is initialized with the biggest possible SAD value at the beginning of motion estimation for each reference block. So, the output of the comparing unit gives the motion vectors of all possible candidate blocks (41 in total) at the end of search of the search window. The multiplication and division operations in AG (figure 4) are implemented by hardwired shifts except *add_h**92 for which stored pre-computed values are used. The subtraction and division operations are done for sign and quarter pel adjustments, respectively.

4. SIMULATION AND SYNTHESIS RESULTS

The proposed architecture has been prototyped in Verilog HDL, simulated and synthesized by Xilinx ISE development tools for Virtex2 device family. Table 2 summarizes the synthesis results. The maximum speed was found to be around 150MHz. Simulation result conforms real time processing of CIF (352x288) frame sequences. Under a clock speed of 120MHz, the core can compute in real time the motion vectors of all various size blocks with 5 reference frames.

Table 2. Simulation and synthesis results.

# of Slices	# of 4-input LUTs	Gate Count	Speed (MHz)
14.5K	28.5K	225K	149.2

5. CONCLUSION

A novel quarter pel full search variable block size motion estimation architecture has been presented in this paper. The architecture is suitable for FPGA implementation as an IP core for H.264/AVC encoder. The parallel and pipelined design allows the core to run at a speed of 120MHz which is sufficient for real time processing of CIF image sequences.

6. ACKNOWLEDGEMENTS

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canadian Foundation for Innovations (CFI), Micronet R&D Canada, and the Canadian Microelectronics Corporation (CMC) for supporting this research.

7. REFERENCES

[1] "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-G050r1, May 2003.

[2] T. Wiegand, G. J. Sullivan, G. Bjntegaard, 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, pp. 560-576, July 2003.

[3] "Fast integer pel and fractional pel motion estimation for AVC," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-F016, December 2002.

[4] Y. W. Huang *et. al.*, "Hardware architecture design for variable block size motion estimation in MPEG-4 AVC/JVT/ITU-T H.264," *Proceedings of the 2003 International Symposium on CAS, ISCAS '03*, pp. II-796-II-799, May 2003.

[5] S. Y. Yap and J. V. McCanny, "A VLSI architecture for variable block size video motion estimation," *IEEE Transactions on CAS II*, vol. 51, no. 7, July 2004.

[6] P. Kuhn, *Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation*, Kluwer Academic Publishers, Boston, 1999.