A Novel Spiral-Type Motion Estimation Architecture for H.264/AVC

Naoyuki Hirai*, Tian Song, Yizhong Liu*, and Takashi Shimamoto****

***Abstract*—New features of motion compensation, such as variable block size and multiple reference frames are introduced in H.264/AVC. However, these new features induce significant implementation complexity increases. In this paper, an efficient architecture for spiral-type motion estimation is proposed. First, we propose a hardware-friendly spiral search order. Then, an efficient processing element (PE) architecture for ME is proposed to achieve the proposed search order. The improved PE enables one-pixel-move of the reference pixel data to top, bottom, right, and left by four ports for input and output. Moreover, the parallel calculation architecture to calculate all block size with the SAD of 4x4 is introduced in the proposed architecture. As the result of hardware implementation, the hardware cost is about 145k gates. Maximum clock frequency is 134 MHz in the case of FPGA (Xilinx Vertex5) implementation.**

***Index Terms*—Spiral-type motion estimation, H.264/ AVC**

I. INTRODUCTION

H.264/AVC is a sophisticated video coding standard in which some novel algorithms have been introduced to achieve high coding efficiency [1]. However, the computational complexity brought about by these novel coding tools has increased significantly. In particular, the motion estimation (ME) process, which accounts for about 80% of the total processing complexity of H.264/AVC encoder, becomes the determining impact factor for H.264/AVC encoder implementation.

In the ME process of H.264/AVC, 7 types of selective block size motion estimation algorithms have been introduced with sizes of 16x16, 16x8, 8x16, 8x8, 8x4, 4x8 and 4x4; these algorithms are collectively called the variable-block-size motion estimation (VBSME) algorithm. This algorithm produces extremely high computational complexity for pre-encoding all candidate modes and calculating the rate distortion optimization (RDO) cost for each block size to determine the optimum coding mode. Therefore, the reduction of the computation complexity and high-speed processing by dedicated hardware are necessary for real-time encoding implementation of H.264/AVC. Moreover, fast processing and the pipeline architecture of the ME architecture are indispensable in the implementation of real-time encoding for high definition applications.

To reduce the complexity of the VBSME algorithm, several fast mode decision algorithms have been proposed [2-4]. These previous works are efficient at decreasing the redundant candidate modes with the sacrifice of picture quality. Some other previous works focus on reducing the search points to realize fast motion estimation [5-7]. In these works, several search patterns are proposed by estimating the search point that with high probability. This methodology is excellent at the complexity reduction, but not based on hardware-friendly considerations. A ME algorithm without considering the hardware implementation may increase the hardware and power consumption.

Numbers of hardware architectures for ME are also proposed to reduce the computation complexity from the view point of LSI architecture, and search range selection [8-14]. Because most of the block-based ME algorithms

Manuscript received Oct. 1, 2009; revised Dec. 18, 2009.
* College of Systems Innovation Engineering, Graduate School of Advanced Technology and Science, The University of Tokushima, Tokushima, 770-0814, Japan
** Computer Systems Engineering, Institute of Technology and Science, Graduate School of Engineering, The University of Tokushima, Tokushima, 770-0814, Japan
E-mail : ra-i1020@ee.tokushima-u.ac.jp

are based on computing the sum of absolute differences (SAD) between corresponding elements in the candidate and reference blocks, the processing element array (PEA) which is used to calculate the SAD becomes the challenging design point. Several fast PEA architectures concerning the fast calculating of SAD are proposed [8-11]. However, all of these PEA architectures based on the continuous macroblock (MB) order.

Another important issue about the complexity reduction of motion estimation is the search range refinement. In these years, some researches found that efficient search range can help to significantly reduce the search point and several algorithms are proposed. Some proposals which concentrate on how to restrain the search range are proposed including our previous work in which a new concept of the optimum search range is proposed [13, 14]. However, these previous works did not provide an efficient architecture to support spiral-type search pattern.

In this paper, an efficient architecture for ME in H.264/AVC is proposed. The rest of this paper is composed as follows. In section II, the ME of H.264/AVC will be reviewed. Then, in section III, the proposed search order and the proposed architecture are introduced. In section IV, the experimental results are shown. The conclusion is given in section V.

II. MOTION ESTIMATION OF H.264/AVC

In H.264/AVC, several new technologies such as VBSME, multiple frame motion estimation, and sub-pixel motion estimation are introduced. By using these new features, H.264/AVC achieves high coding efficiency in comparison with previous coding standards.

1. Motion Search of H.264/AVC

As shown in Fig. 1, ME is a process to search for the motion vector (MV). First, the predictor motion vector (PMV) is calculated from the MV of the adjacent macroblocks (MB). Next, the search center is determined on the basis of the PMV and the ME is performed within a dedicated search range (SR). The position with smallest SAD (sum of absolute difference) is selected as the best point and the MV is obtained. SAD is obtained with the equation of

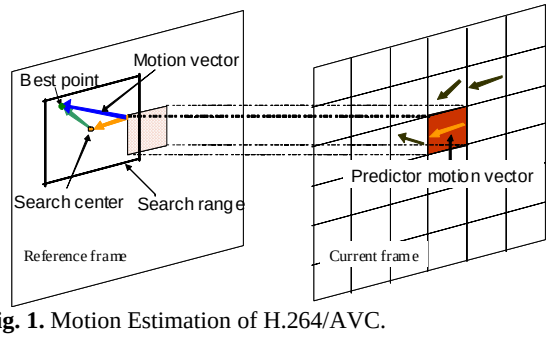


Fig. 1. Motion Estimation of H.264/AVC.

$$SAD(m,n)=\sum_{k=0}^{W-1}\sum_{l=0}^{H-1}Diff\,(m,n,k,l) \tag{1}$$

where

$$Diff\,(m,n,k,l)=\left|R(m+k,N+l)-C(k,l)\right| \tag{2}$$
$$m\in[0,M-1],n\in[0,N-1]$$

W and *H* indicate the width and height of current encoding block, respectively. *M* and *N* indicate the width and height of search range, respectively. *C(k,l)* is the value of the pixel of the current block, and *R(m+k,n+l)* is the value of the pixel in the reference block. In the process of ME in H.264/AVC, VBS-ME and MRF are adopted.

In H.264/AVC, VBS-ME achieves very high coding efficiency by introducing multiple block size motion estimation. VBS-ME uses 4 types bigger block of 16x16, 16x8, 8x16, 8x8 and other 3 types smaller block of 8x4, 4x8, and 4x4 which is shown in Fig. 2.

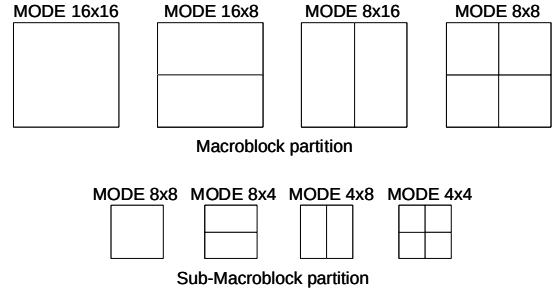


Fig. 2. Variable block size in H.264/AVC.

2. Spiral-Type Motion Estimation

From many fast ME algorithms and architectures proposed so far, it is well known that spiral-type motion estimation (STME) is efficient because almost best points concentrate on the search center. With the

increasing distance from the search center the probability to find the best point is decreasing.

The search order of STME is a spiral increasing range from the search center. Fig. 3(a) shows a STME sample.

As shown in Fig. 3(a), STME starts searching from the search center and gradually expands the search range. An efficient spiral search order is proposed in the reference software JM10 [16] for H.264/AVC which is shown in Fig.3(b). The number in the circle indicates the search order. Number 0 is the search center and the search process starts from this position. The search order proposed by [15] is only on the basis of the probability of selected point which makes it difficult to be implemented by hardware. However, in the case of hardware implementation, regular and simple search order is preferable. Furthermore, to realize continuous memory access a simple search order which can move to the next search point by one shift is desirable.

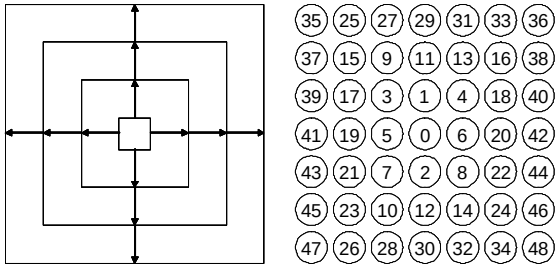


Fig. 3. STME and spiral search order.

III. PROPOSED ME ARCHITECTURE

1. Proposed search order

To design a hardware-friendly architecture the search order must be carefully considered. In this paper, search order that considering the hardware implementation of spiral-type motion estimation is proposed. Fig. 4 shows the proposed search order which can move to the next search point with only one shift to up, down, left, or right.

As shown in Fig. 4, the number in the circle indicates the search order. The search point starts from the search center and moves to the next neighboring search point with increasing numbers. Because the operation of this search order is simple, the hardware design is easier than that of the traditional spiral search order.

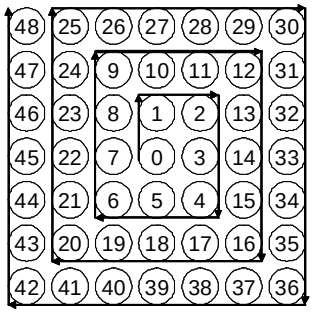


Fig. 4. Proposed search order.

2. Total architecture

We propose an efficient STME architecture to achieve the proposed search order. Fig. 5 shows the entire structure of the proposed architecture.

This structure can calculate the SAD for one search point in one clock cycle using 16 PE Array 4x4 modules as shown in Fig. 5. Moreover, this architecture can make it possible to shift reference pixel data to top, bottom, right or left by connecting PE Array of top, bottom, left or right. First, the pixel data of reference MB is loaded from an external memory to SRAM1 and SRAM2 and transferred from SRAM1 and SRAM2 to PE Array. Then, the pixel data of current MB is transferred from SRAM to PE Array and saved by a register array in PEs. The current MB is saved until the ME processing for one MB finished. Because the reference MB input to the SRAM1 and SRAM2 are completely duplicated data, the loaded reference data is transferred to SRAM1 and SRAM2 in the same cycle without additional transfer cycles. Therefore, comparing to traditional single SRAM architectures, the proposed architecture will not increase the external memory bandwidth. The pixel of reference MB can shift up, down, left or right in each cycle and the ME for one search point is performed in one clock. In

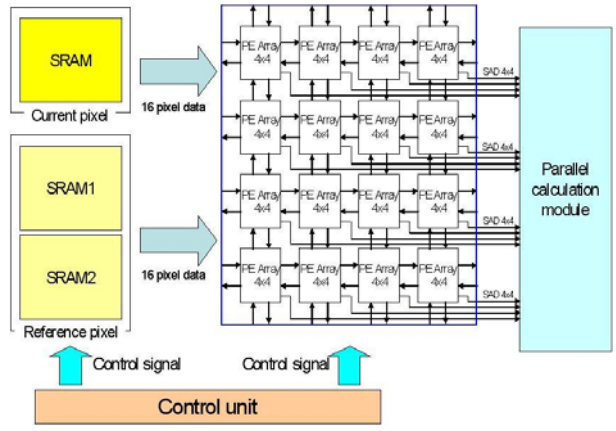


Fig. 5. Total architecture.

each cycle, 16 SAD 4x4 is calculated by 16 PE Array 4x4. Calculated SAD 4x4 is transferred to parallel calculation module. In parallel calculation module, SAD of all block size is calculated by using calculated SAD 4x4. These processing is controlled by control unit, and the ME processing is executed.

3. PE Array architecture

PE is a module that calculates the absolute difference between the pixel of the reference block and the pixel of the current block. Fig. 6 shows the architecture of PE Array 4x4.

To enable the reference data shifting to top, bottom, right or left in PE Array 4x4, each PE is connected to the PE of top, bottom, right or left one. This structure generates SAD 4x1 by accumulating the absolute difference of each PE. Furthermore, SAD 4x4 is generated by accumulating generated SAD 4x1. However, because long delay will be induced by the multiple arithmetic accumulating modules which generates SAD 4x1 and SAD 4x4, registers are inserted to improve maximum clock frequency.

The PE module is designed to be able to shift to top, bottom, right or left. The proposed PE and the traditional one are shown in Fig. 7 and Fig. 8, respectively.

As shown in Fig. 7, traditional PE can shift pixel data in one direction by connecting PEs of the same direction. However, the proposed PE can shift reference pixel data to top, bottom, right or left. The current pixel data is always shifting into the PE Array from top direction.

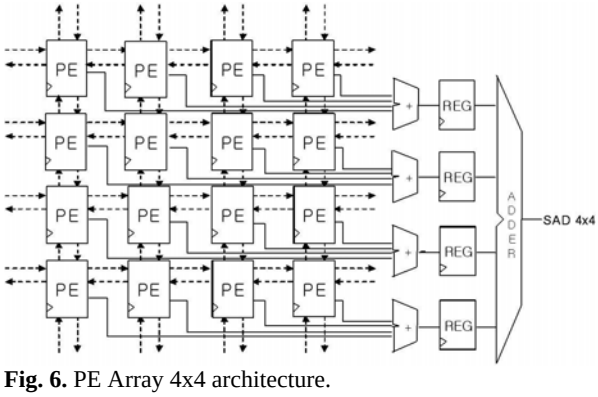


Fig. 6. PE Array 4x4 architecture.

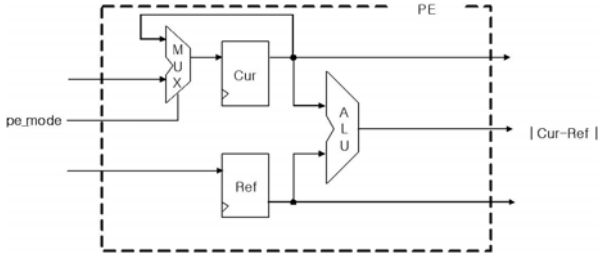


Fig. 7. Traditional PE architecture.

Reference pixel data need input from four directions as well as output to four directions. Therefore, four input ports and four output ports are prepared in each PE. The connection of PEs is shown in Fig. 9(a).

As shown in Fig. 9(a), each PE is connected with surrounding PEs: “from top” connects to “to bottom”, “from bottom” connects to “to top”, “from left” connects to “to right”, and “from right” connects to “to left”. An example when “from top” is selected by the multiplexer is shown in Fig. 9(b). The reference data from the output port “to bottom” of upper PEs is shifted to bottom PEs and the search position is shifted. In this way, each I/O port of PE enables the shift of the reference pixel data by selecting the input of reference pixel data using multiplexer.

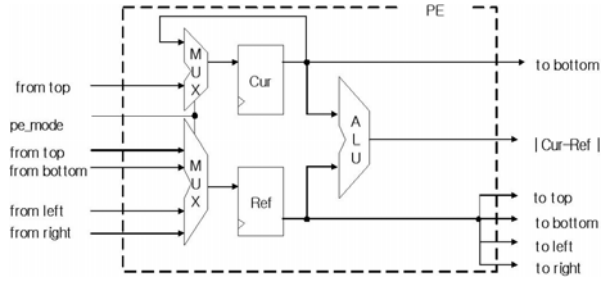


Fig. 8. Proposed PE architecture.

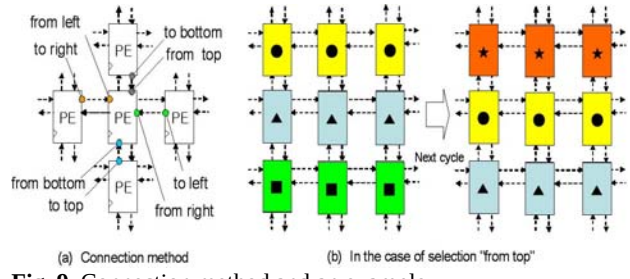


Fig. 9. Connection method and an example.

4. Double SRAM architecture

To achieve STME for one search point in one cycle, addition to 16x16 pixel data in the registers, complementally 16 pixels have to be loaded from memory in each cycle. However, due to the different shift directions in the corner of each range, the complementally 16 pixels have to be carefully arranged to be loaded in one cycle. Traditional single SRAM architecture is shown in Fig. 10.

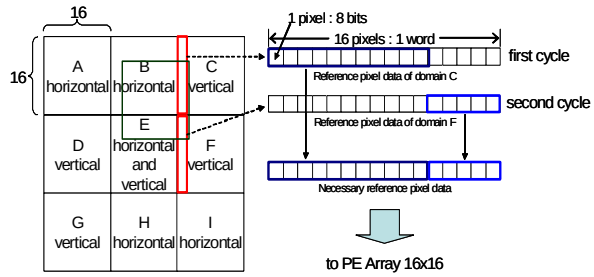


Fig. 10. Single SRAM architecture.

In Fig. 10 the search range is set to 48x48 and reference pixel data of MB A, B, C, D, E, F, G, H, I are saved separately. One word of SRAM consists of 16 pixel data with total bit length of 128 bits (8bits/pixel). Therefore, pixels of one row or one column with 16 pixels can be loaded in one cycle. However, because SRAM outputs the data of one address in one cycle, when the data of the area of C and F are necessary, 16 reference pixel data cannot be output in one cycle.

In the proposed memory solution, reference pixel data are double buffered into two separate SRAMs with pixels of vertical and horizontal directions saved respectively. Fig. 11 shows the double SRAM architecture.

As Fig. 11 shows, pixel data for MBs of A, B, C, D, F, G, H, I are saved in SRAM1, and the B, D, E ,F, H are saved in SRAM2. Pixel data for MBs of B, D, F, H, with different direction are double saved in two SRAMs. When the reference data for different directions are needed, the pixel data can be loaded from different SRAM in the same cycle. In the example shown in Fig. 11, the reference pixels for MBs C and F are loaded from SRAM1 and SRAM2 respectively when the search direction is from left position to right position.

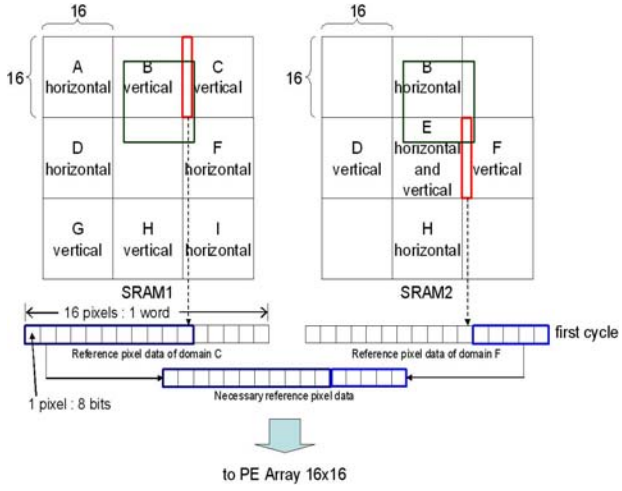


Fig. 11. An example for double SRAM architecture.

5. Parallel calculation of SAD

Previous work shows that in many cases the calculation of SAD for different block size is desirable to be performed in parallel [12]. In this paper, a parallel calculation method for the calculation of SAD is proposed. Fig. 12 shows the structure of the parallel calculation module.

As the input data, 16 SADs for 4x4 blocks are input in this module. Firstly, SADs for 8x4 and 4x8 blocks are calculated using SADs of 4x4. Next, SADs for 8x8 blocks are calculated using SADs of 8x4 or 4x8 block. Then, SADs of 8x16 and 16x8 blocks are calculated using calculated SADs of 8x8 blocks. Finally, the SAD of 16x16 blocks is calculated using SADs of 8x16 or 16x8 block.

However, the delay is generated because four adders are necessary to calculate the SAD of 16x16 blocks. The critical path can be shorten by inserting registers depend on the user target applications.

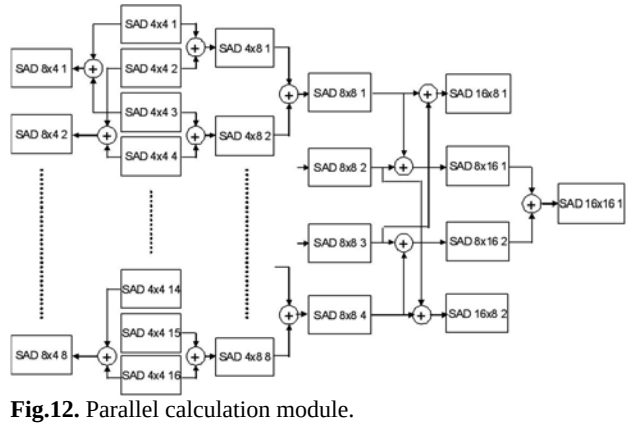


Fig.12. Parallel calculation module.

IV. Implementation Results

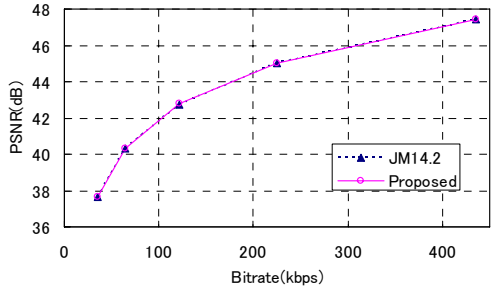
The proposed search order is different from that of the original order which is recommended by standard reference software. In our simulation, the R-D curve of the proposed search order is compared with the original one. Because the basic full search motion estimation algorithm has no relation with the search order, the fast-full search algorithm is used in our comparison [15]. The input resolution is CIF, and the test frame number is 60.

From the simulation results we can know that almost no change occurred in R-D curve. It is obvious that the proposed search order works well with the fast-full

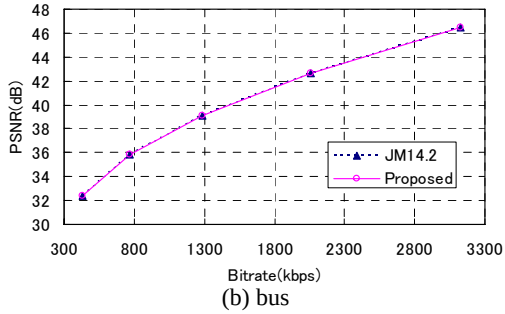
search algorithm. It is considered that the proposed algorithm can be widely used by other motion estimation algorithm. However, some other motion estimation dedicated evaluations are necessary.

The proposed architecture is described with Verilog-HDL and implemented on Xilinx Virtex5 FPGA technology. The implementation results are shown in Table 1.

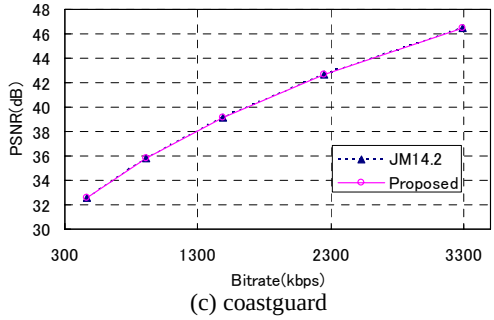
The gate count number shown in Table 1 does not include the control unit and SRAM. The implementation results show that the proposed architecture use more gates compare to previous implementations which perform simple line-order motion search. However, using the proposed architecture, more efficient spiral-type motion search is possible to be performed.



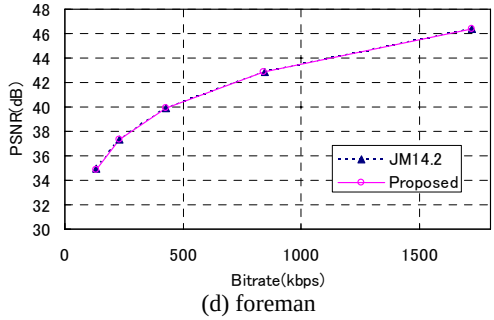
(a) akiyo



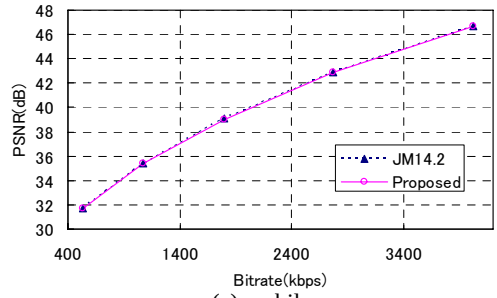
(b) bus



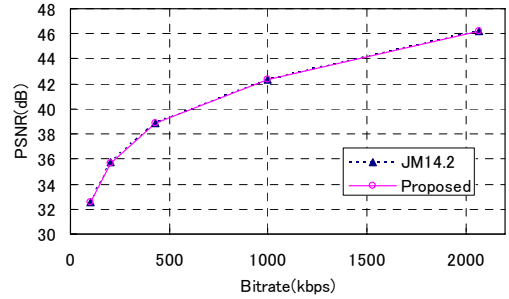
(c) coastguard



(d) foreman



(e) mobile



(f) waterfall

Fig. 13. RD curves comparison of CIF sequences.

Table 1. Implementation Results

	ours	[15]
Gates	145k	104.7k
Clock (MHz)	134	200
Memory (bit)	30,720	64,008

V. CONCLUSIONS

In this paper, a spiral-type ME architecture is proposed. The hardware design by the proposed search order can achieve the ME method to start a search from the search center because it is easier than the spiral search. As a result, the hardware cost is about 145k gate and maximum clock frequency of 134 MHz is achieved.

REFERENCES

[1] Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, “Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264-ISO/IEC 14496-10 AVC),” March 2003.

[2] D. Wu, F. Pan, K. P. Lim, S. Wu, Z. G. Li, X. Lin, S. Rahardja, and C. C. Ko, "Fast intermode decision in H.264/AVC video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 15, No.6, pp.953-958, 2005.

[3] B. Kim, S. Song, and C. Cho, "Efficient inter-mode decision based on contextual prediction for the P-slice in H.264/AVC video coding, *Proceedings of IEEE International Conference on Image Processing*, pp.1333-1336, Oct. 2006.

[4] T. Kuo, and C. 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, pp.1185-1195, 2006.

[5] R. Li, B. Zeng, and M. L. Liuo, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.4, No.4, pp.438.442, Aug. 1994.

[6] S. Zhu, and K.-K. Ma, "A new diamond search algorithm for fast block matching motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.92, No.2, pp. 287.293, Feb. 2000.

[7] C. Zhu, X. Lin, and L. P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.12, No.5, pp.349.355, Aug. 2002.

[8] J. H. Lee, and N. S. Lee, "Variable block size motion estimation algorithm and its hardware architecture for H.264/AVC," *Proceedings of IEEE International Symposium on Circuits and Systems*, Vol.3, pp.741-744, May 2004.

[9] T. C. Chen, S. Y. Chien, Y. W. Huang, C. H. Tsai, C. Y. Chen, and L. G. Chen, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 16, No.6, pp.673-688, 2006.

[10] Y. Song, Z. Liu, S. Goto and T. Ikenaga: "Scalable VLSI architecture for variable block size integer motion estimation in H.264/AVC," *IEICE Trans. on Fundamentals*, Vol.E89-A, No.4, pp.979-988, 2006.

[11] Y. Q. Hung, Z. Y. Liu, Y. Song, S. Goto, and T. Ikenaga: "Parallel improved HDTV720p targeted propagate partial SAD architecture for variable block size motion estimation in H.264/AVC," *IEICE Trans. Fundamentals*, Vol.E91-A, No.4 April 2008.

[12] L. Zhang, and W. Gao, "Improved FFSBM algorithm and its VLSI architecture for variable block size motion estimation of H.264," *Proceedings of IEEE International Symposium on Intelligent Signal Processing and Communication Systems*, pp.445-448, Dec. 2005.

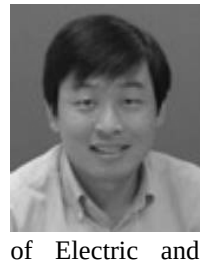
[13] K. Ogata, K. Saito, T. Song, and T. Shimamoto, "Variable search range motion estimation algorithm for H.264/AVC," *IEICE Society Conference*, No. A-6-8, Sep. 2006.

[14] T. Kato, T. Song, Y. Z. Liu and T. Shimamoto: "Motion vector predicted variable search range motion estimation algorithm for H.264/AVC," in *Proc. of International Workshop on Vision, Communications and Circuits* (IWVCC2008), pp.161-164, Nov. 2008.

[15] Y. Huang, Q. Liu, S. Goto, and T. Ikenaga: "Adaptive Sub-Sampling Based Reconfigurable SAD Tree Architectire for HDTV Applicaion," *IEICE Trans. Fundamentals*, Vol.E92-A, No.11 Nov. 2009

[16] JM10 (http://iphome.hhi.de/suehring/tml/)

Naoyuki Hirai received his B.E. degree from the Tokushima University in 2009. He is working towards his M. E. degree at the faculty of the department of Electric and Electronics at Tokushima University. He is a member of IEEE. His current research interests include video coding algorithms and VLSI architectures.



Tian Song received his B.E. degree from the Dalian University of Technology, China, in 1995, M.E., Dr.E from Osaka University in 2001, 2004 respectively. From 2004, he joined the faculty of the department of Electric and Electronics of the University of Tokushima. He is a member of IEICE and IEEE. His current research interests include video coding algorithm, VLSI architectures and system design methodology.

Yizhong Liu received his B.E. and M. E. degree from Xidian University, China in 2002 and 2005 respectively. From 2005 to 2008, he worked in Infineon Technologies Xi'an as an engineer. He is currently a Ph.D candidate at the Faculty of the Department of Electric and Electronics, Tokushima University, Japan. His current research interests include video coding algorithms, VLSI architectures and low-power system architectures.

Takashi Shimamoto was born in Tokushima, Japan, on November 22, 1959. He received the B.E. and M.E. degrees in electrical engineering from Tokushima University, and the Dr. E. degree from Osaka University, in 1982, 1984, and 1992, respectively. He joined Tokushima University in 1984 as an Assistant Professor. Presently he is an Associate Professor of the Department of Electrical and Electronic Engineering, Graduate School of Advanced Technology and Science, Tokushima University. His interests of research include heuristic algorithms for VLSI CAD.