2011 IEEE/IFIP 19th International Conference on VLSI and System-on-Chip

A Full-mode FME VLSI Architecture Based on 8x8/4x4 Adaptive Hadamard Transform For QFHD H.264/AVC Encoder

Jialiang Liu, Xinhua Chen
College of Information Science and Engineering
Shandong University of Science and Technology
Qingdao, China

Yibo Fan, Xiaoyang Zeng
State Key Lab of ASIC and System
Fudan University
Shanghai, China
Email: fanyibo@fudan.edu.cn

Abstract—**Adaptive Block-size Transform (ABT) has been added to H.264/AVC standard with the Fidelity Range Extension. In this paper, we apply this ABT concept to our FME design and propose a full-mode FME architecture based on 8x8/4x4 adaptive Hadamard Transform. This technique can avoid unifying all variable block-size blocks into 4x4-size blocks and improve the encoding performance. We also exploit the linearity of Hadamard Transform in quarter-pel refinement and decrease the cycles caused by the second long search process. In architecture level, we employ two interpolating engines that can support 8-pel and 4-pel input to time-share one SATD (Sum of Absolute Hadamard Transform) Generator. These strategies can increase parallelism and reduce the cycles efficiently. Besides, this design can support full modes, which guarantees the encoding performance. Experimental results show that our design can achieve real-time processing for QFHD@30fps at the operation frequency of 320MHz with 444.6K gates hardware.**

Keywords: H.264/AVC; fractional motion estimation; adaptive block-size hadamard transform; quad full high definition

I. INTRODUCTION

H.264/MPEG-4 AVC is the latest video-coding standard jointly developed by ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG). After the first version, H264/MPEG-4 AVC didn't stop its development. Afterwards Fidelity Range Extension (i.e. High Profile), Scalable Video Coding (SVC) and Multiview Video Coding (MVC) extensions were added to H.264/MPEG-4 AVC standard. Nowadays, H.264/MPEG-4 standard can be widely applied to all kinds of multimedia devices or occasions because of its more predominant performance.

Variable block size motion estimation (VBSME) is one of the most important techniques. And this technique exploits fully the temporal redundancy to provide much better compression performance. In general, most implementation works separate Motion Estimation (ME) into two stages: integer motion estimation (IME) and fractional motion estimation (FME). Firstly, IME performs motion search within the entire search window to find an integer motion vector (IMV) for each of 41 subblocks. In the next step, FME performs motion search around the refinement center pointed to by IMV and further refines 41 IMVs into quarter-pel precision [3]. In the whole ME process, FME plays an important role in improving the video quality (about 4+ dB) and reducing the bit rate (about 20%~40%). Almost all previous works about ME mentioned this point. However, FME will occupy almost over 45% of the computation complexity of encoding process [4].

Hadamard Transform (HT) is main part of Sum of Absolute Transformed Difference (SATD), which is widely used for estimating cost of rate distortion. The 8x8/4x4 adaptive transform (Adaptive Block-size Transform, ABT) technology with Fidelity Range Extension is added to the main profile of H.264/AVC standard. Our simulation shows that this technique can improve about 0.05dB video quality and save about 3% bit rate. In this paper, we propose a FME architecture that supports 8x8/4x4 adaptive HT (Adaptive Block-size HT, ABHT). ABHT is a subset of ABT technique. When the ABT technique is applied into the entire encoder, the ABHT will obtain its maximum performance.

The rest of this paper is organized as follows. In Section Ⅱ, we firstly point out the challenges of FME and introduce some others' efforts. Then we describe related algorithm which is the foundation of our design in Section Ⅲ. In Section Ⅳ, we introduce the details of our proposed architecture. And finally, we give out the implementation results and conclusion in Section Ⅴ and Section Ⅵ respectively.

II. CHALLENGES AND PREVIOUS WORKS

A. The Challenges of FME

As one part of entire ME, FME isn't also normative. However, there is a commonly used method for FME in JM reference software. Most of FME VLSI architectures proposed later adopt this method [1][2][3]. The search process of this method is also separated into two steps: in the first step, we perform half-pel refinement around the best integer-pel candidates, and then find the best quarter-pel candidate around the best half-pel candidate in the second step. The second step doesn't work until the accomplishment of the first step. As a result, this two-step method critically limits the searching speed.

The second challenge of FME is the multiple modes caused by VBSME. VBSME splits one MacroBlock (MB) into one 16x16, two 16x8, two 8x16, or four 8x8 partitions, and that every 8x8 partition can be further split into one 8x8, two 8x4, two 4x8 or four 4x4 sub-partitions. In the two-step search method, all partitions are organized as shown in Fig.1. There are total 7 modes, *i.e.* 41 partitions with 41 IMVs. If one FME architecture supports all modes, it will need very high performance to meet the requirement of real-time high-definition video encoding.

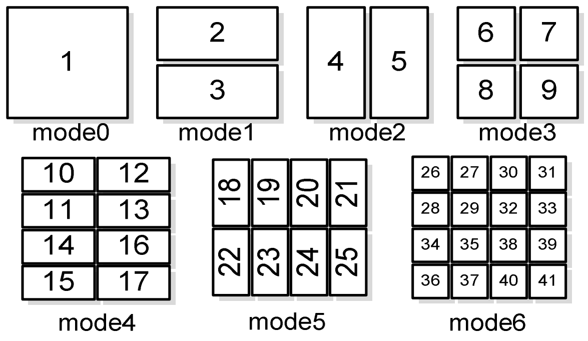

Figure 1 7 modes of all partitions

B. Previous Works

Nowadays, there are some VLSI architectures proposed, some of them support two-step full modes search, but most of them employ the low complexity algorithm for high performance application at the cost of visual quality drop.

Chen *et al.*[1] proposes a architecture based on the two-step full search algorithm. This architecture unifies different size blocks into 4x4 size block, which obtains 100% hardware utilization. This architecture can support full modes, however, which also limits its performance. Finally, they adopt one Advanced Mode Pre-Decision Algorithm (AMPA) to reduce 7 modes to 3 modes only for SDTV specification. Wu's architecture [2] also supports all modes. It employs three interpolator engines to share two SATD generators, which obtains the high performance by increasing the data throughput. It still takes more than 600 cycles @154MHz to support the 1080p specification, but also the control of this architecture would be more complex.

The full-mode and two-step search algorithm limits the high specification application of FME. So most architectures based on the low complexity algorithm are proposed. Wang *et al.*[4] exploit the unimodal error surface feature of FME and only explore the neighborhood position around the minimum one to skip other unlikely ones. Though this method reduces the hardware cost by reducing search points, it doesn't resolve the problem caused by the two-iteration search. Lin *et al.*[5], Kuo *et al.*[6] and Ta *et al.*[7] propose their single-iteration architectures based on the prediction-based directional FME algorithm. This algorithm only searches six candidates, so it is hard to find the best matching candidates, especially when encoding the complex motion video sequences. Huang *et al.*[8] propose a high parallel architecture meeting SHV application. They remove the modes below 8x8 and then exploit the edge detection techniques to further reduce modes. Tsung *et al.*[8] adopt a completely different solution. They abandon the search method ruled by JM reference software, and then utilize the linearity of HT to obtain the HT coefficients of sub-pels. This method reduced the high bandwidth caused by 6-tap FIR and only 0.02dB PSNR drop.

TABLE I. THE NUMBER OF MODES SUPPORTED BY SOME IMPLEMENTATIONS

References	Techniques	Modes
[1]	IME mode decision	3
[4]	IME mode decision reduce candidates	2
[6]	reduce modes	5-7
[7]	remove lower8x8 modes edge detection	2
[10]	adaptive mode decision	1-7
[2][3]	full search	7
our work	single-iteration	7

Table I gives the number of modes supported by some implementations, and most of previous works achieved their required performance by reducing modes. In this paper, we propose a FME architecture that can support full modes on the basis of previous studies. In our architecture, we employ two interpolators to share one SATD generator in turn. Moreover, we use ABT technology in the HT for SATD Generating which can improve the encoding performance. In order to avoid the long latency caused by the second step, we also exploit the linearity of HT to obtain HT coefficients of quarter-pels. This idea has been brought up in [9], however, there are some differences. We make use of the arrival delay of the SATD of quarter-pels and half-pels to avoid adopting 25-input comparator, which reduces hardware cost and long data path.

III. RELATED ALGORITHM

A. Reusable relationship between 8x8 and 4x4 Hadamard Transform

Firstly, we give out the 8x8 and 4x4 HT matrices (1) and (2). Here we ignore the scaling factor of transform matrix.

$$\mathbf{H}_{4x4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{1}$$

$$\mathbf{H}_{8x8} = \left[\begin{array}{cccc:cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hdashline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right] \tag{2}$$

By analyzing the relationship between $\mathbf{H}_{4x4}$ matrix and $\mathbf{H}_{8x8}$ matrix, we can easily express 8x8 HT matrix as (3)

$$\mathbf{H}_{8x8} = \begin{bmatrix} \mathbf{H}_{4x4} & \mathbf{H}_{4x4} \\ \mathbf{H}_{4x4} & -\mathbf{H}_{4x4} \end{bmatrix} \tag{3}$$

Assume we have a 8x1 vector, $\vec{x}$, which can be expressed as two 4x1 vectors, $\vec{x}_0$ and $\vec{x}_1$, as shown in (4).

$$\vec{x} = \begin{bmatrix} \vec{x}_0 \\ \vec{x}_1 \end{bmatrix} \tag{4}$$

Now, we operate one dimension (1-d) HT to vector $\vec{x}$, which is expressed as (5).

$$\vec{X} = \mathbf{H}_{8x8} \cdot \vec{x} = \begin{bmatrix} \mathbf{H}_{4x4} \cdot \vec{x}_0 + \mathbf{H}_{4x4} \cdot \vec{x}_1 \\ \mathbf{H}_{4x4} \cdot \vec{x}_0 - \mathbf{H}_{4x4} \cdot \vec{x}_1 \end{bmatrix} \tag{5}$$

From (2), we can see that one 1-d 8x8 HT unit contains two 1-d 4x4 HT units. So we can design a reusable circuit that supports one 8x8 HT and two 4x4 HT adaptively. Fig.2 shows this 1-d 8x8/4x4 HT circuit. The shaded part indicates the 1-d 4x4 HT. In our design, we utilize this 1-d 8x8/4x4 adaptive HT circuit to implement the 2-d 8x8/4x4 HT circuit used in the SATD calculation.

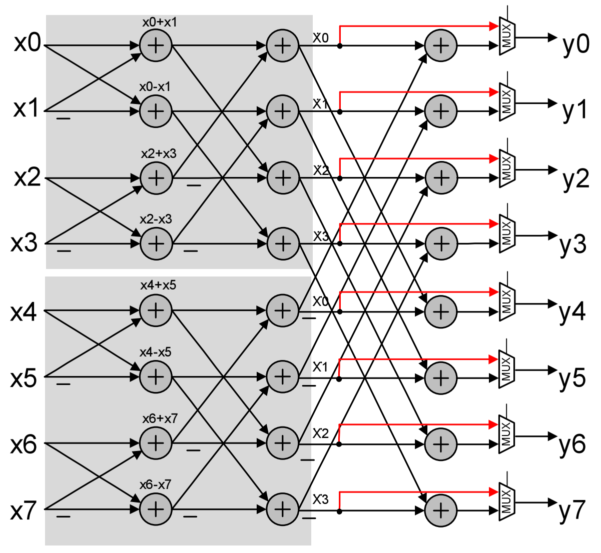

Figure 2 Reusable 8x8/4x4 1-d Hadamard Transform

B. *The linearity of Hadamard Transform*

The linearity of HT is defined as (6):

$$HT(a\vec{x} + b\vec{y}) = aHT(\vec{x}) + bHT(\vec{y}) \tag{6}$$

Here, HT (.) presents HT operation. "a" and "b" are constants. "$\vec{x}$" and "$\vec{y}$" are transformed data.

Let "**Q**" and "**C**" denote the quarter pixels vector and current MB (CMB) pixels vector respectively. "$\mathbf{H}'$" and "$\mathbf{H}''$" represent the half pixels (or integer pixels) vectors used for generating quarter pixels vector by bilinear filter that is defined in standard, as shown in (7).

$$\mathbf{Q} = (\mathbf{H}' + \mathbf{H}'' + \mathbf{1})/2 \tag{7}$$

The HT of differences between quarter-pel candidates and CMB pixels can be expressed as (8).

$$\begin{aligned} HT(\mathbf{Q} - \mathbf{C}) &= HT(\frac{\mathbf{H}' + \mathbf{H}'' + \mathbf{1}}{2} - \mathbf{C}) \\ &= HT(\frac{\mathbf{H}' - \mathbf{C}}{2} + \frac{\mathbf{H}'' - \mathbf{C}}{2} + \frac{\mathbf{1}}{2}) \\ &= (HT(\mathbf{H}' - \mathbf{C}) + HT(\mathbf{H}'' - \mathbf{C}) + HT(\mathbf{1}))/2 \end{aligned} \tag{8}$$

Here, $HT(\mathbf{H}' - \mathbf{C})$ and $HT(\mathbf{H}'' - \mathbf{C})$ are the HT of differences between half pixels (or integer pixels) and CMB pixels, the values of which have been yielded in the first step search. So, we can use this feature to generate the HT coefficients of quarter-pel candidates in the first step, which can remove the long latency caused by the second step search.

IV. THE PROPOSED ARCHITECTURE FOR FME

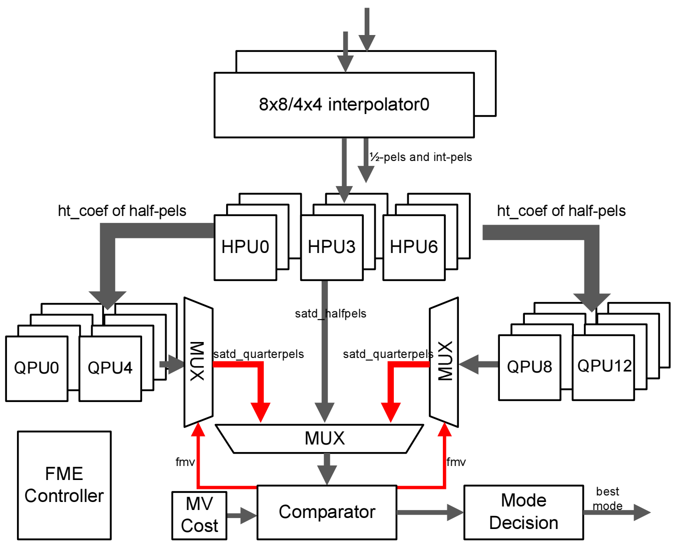

Figure 3 The proposed FME architecture

Aiming at high performance application, we propose a high parallelism and high throughput architecture, as shown in Fig.3. In this architecture, we employ two interpolators. These two interpolators can continuously feed the half candidates to SATD Generator in turn, which can increase the utilization rate of SATD Generator in unit time. In Fig.3, SATD Generator is divided into two parts: one part is half-pel SATD calculation (HPU, Processing Unit of half pixels) and the other is quarter-pel SATD calculation (QPU, Processing Unit of quarter pixels). In our design, we employ 9 HPUs and 16 QPUs for high parallelism. The HT circuit in HPU can support 8x8/4x4 adaptive transform. The upper 8x8 blocks (including 8x8 block) are arranged into 8x8 HT, while the lower 8x8 blocks run at 4x4 HT. Compared with those architectures in which there is only 4x4 HT, this design can increase the data throughput and

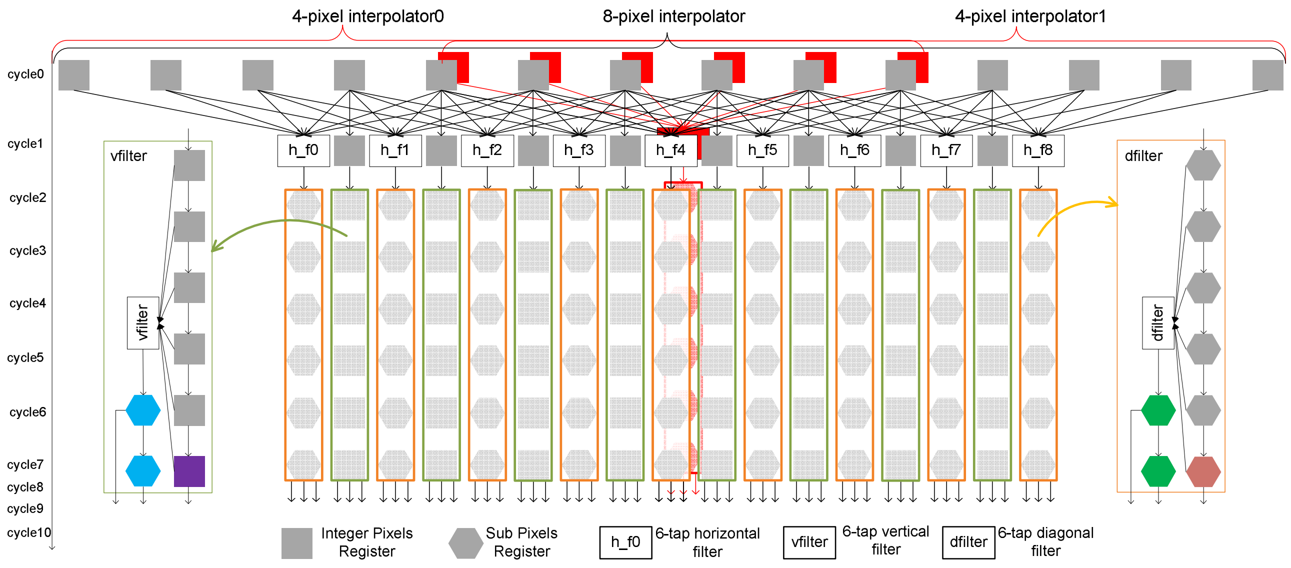

Figure 4 8x8/4x4 interpolator

encoding performance. In addition, we also exploit the linearity of HT to obtain the quarter-pel HT for quarter-pel search. The half-pel HT can be obtained directly by bilinear filtering of half-pel HT. The QPUs in Fig.3 include the bilinear filters and add-tree for generating SATD of quarter-pel candidates.

A. Two Interpolators Time-sharing One SATD Generator

The proposed interpolator is shown as Fig.4. This interpolator can support 8-pixel interpolation and two 4-pixel interpolations, which can adapt to all variable block sizes. One 16x16 block is split into two 8x16 blocks, and 16x8 block is split into two 8x8 size blocks. Since our interpolator can support two 4-pixel interpolations, it allows loading concurrently two 4xn (n=4, 8) blocks.

One problem exists in the conventional architectures based on single interpolator. The available data thrown out by the interpolator are not continuous, while SATD Generator has the capacity of accepting the input data continuously. So if we only adopt single interpolator, the utilization rate of SATD Generator in unit time is very low. To solve this problem, we employ two interpolators which can continuously send the candidates generated by these two interpolators in turn to SATD Generator. These two interpolators keep SATD Generator working. Fig.5 depicts the schedule of loading reference pixels for all partitions in every mode. If we only use one interpolator applied to our design, from Fig.5 (a) can we see that it will take 419 cycles in loading reference data. However, according to our two-interpolator scheme, the cycles can be reduced to 292 cycles. This is because we adopt a pre-loading strategy. Now, let us take 8x8 mode as an example. An 8x8 mode has four 8x8 blocks, and every block needs a 14x14 size reference block. When the reference pixels of the first 8x8 block are continuously fed into one interpolator by row, according to our design, only the outputs from 10th to 17th after 11 cycles latency are available. If there is only one interpolator and we continuously feed data to it, there will be 6 cycles latency during switching blocks, and in this interval SATD Generator is ideal state. By employing another interpolator, we

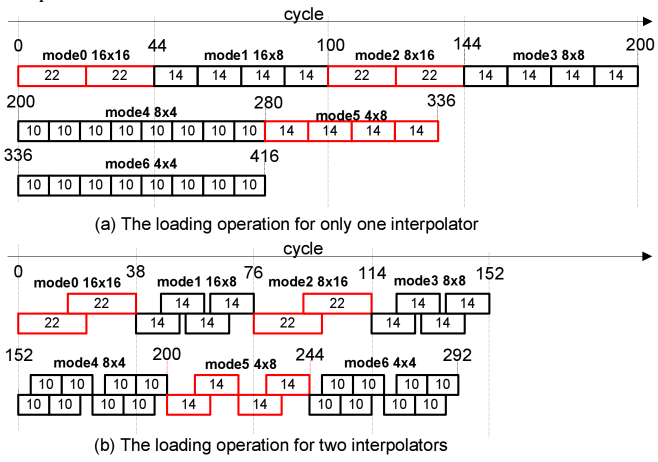

Figure 5 the schedule of loading reference pixels for all the partitions in every mode

can load the reference data in advance of 6 cycles. So, when one interpolator ends outputting available data, another one is just in time to yield the available data of the next block. Compared with conventional single-interpolator scheme, this method can cut down cycles efficiently.

B. SATD Generator Based 8x8/4x4 Adaptive Hadamard Transform

Most of previous solutions only adopted 4x4 HT, because the minimum block size is 4x4. However, the variable block-size transform can better present the natural feature of the real-world image. In this design, we introduce the 8x8 HT into our FME architecture, as shown in Fig.2. The reusable relationship between 8x8 and 4x4 HT matrices determines that the circuit used for 8x8 HT can also be used for 4x4 HT. The upper 8x8-size blocks perform 8x8 HT, while the rest blocks run at 4x4 HT. This SATD Generator can support two-way 4x4 HT calculations, so we can transform two 4x4 size blocks simultaneously. According to our experimental results (JM 16.0, IPPP, RDO off, QP=25/30/35/40/45), ABT technology cause about 0.05dB PSNR increase and about %3 bitrate saving, as shown in Fig.6.

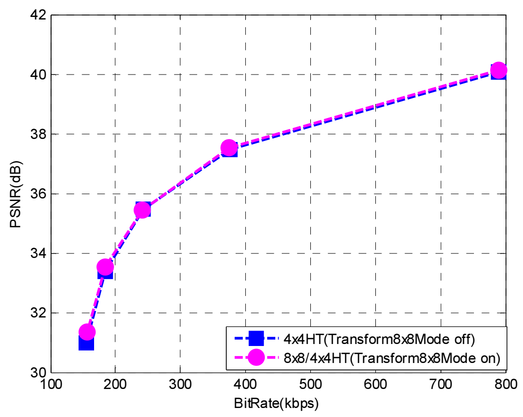

Figure 6 Rate-distortion curve of adopting 8x8/4x4 adaptive HT, compared with only 4x4 HT

C. *Improved Two-iteration Process*

The original second iterative process includes bilinear filtering for quarter-pel candidates, SATD Generating and best candidates selecting. Now, we apply the linearity of HT to quarter-pel refinement process and reduce the second iterative process to one process that only includes Best Candidate Selecting. Fig.7 shows comparison of the original iteration and the improved two-iteration process.

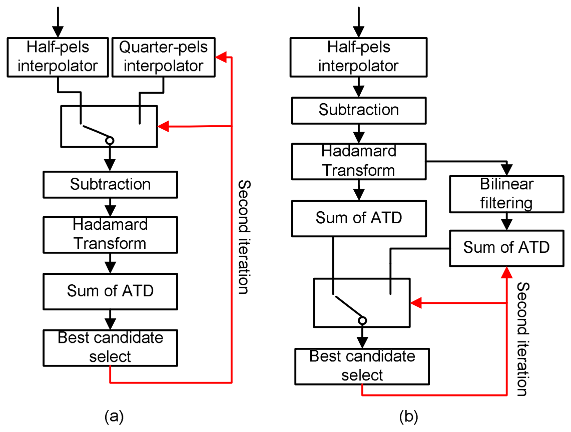

Figure. 7 The two-iteration search process.(a) the original two-iteration search process.(b) the improved two-iteration search process

After adopting this method, we have to face one problem. According to our interpolator, there will be 9 candidates arriving contemporarily. Using the HT coefficients corresponding to these nine candidates, we only get HT coefficients of 16 quarter-pel candidates in the red square, as shown in Fig.8. If we want get HT coefficients of all quarter-pel candidates, we have to increase 12 HT circuits to serve other 12 positions (yellow circles in Fig.8) that are useless for FME. Fortunately, our experimental results show that about 77.4% of the best matching candidates are located at the central 25 candidates, as shown in Fig.9. Based on this feature, we decide to abandon the marginal 21 quarter-pel candidates for saving hardware cost. Besides, in the second iteration process, we still select the quarter-pel candidates according to the best half candidate as original two-iteration process. The quarter-pel candidates are around the best half-pel candidate but limited in the red square. This can avoid searching simultaneously among 16 quarter-pel candidates, and sequentially we can avoid the long comparing data path. According to our simulation, there is about 0.03 dB quality drop in average compared with the situation in which there is only 4x4 HT. If we didn't exploit 8x8/4x4 adaptive HT techniques, the quality loss would be more. Fig.10 shows the rate-distortion performance after introducing our scheme.

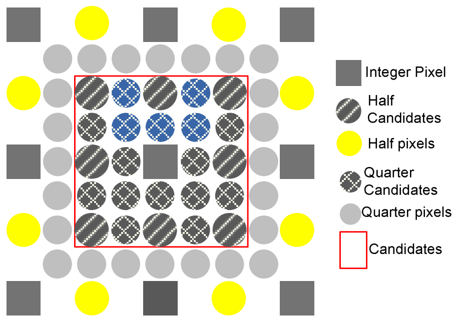

Figure 8 Sub-pel candidates in red square

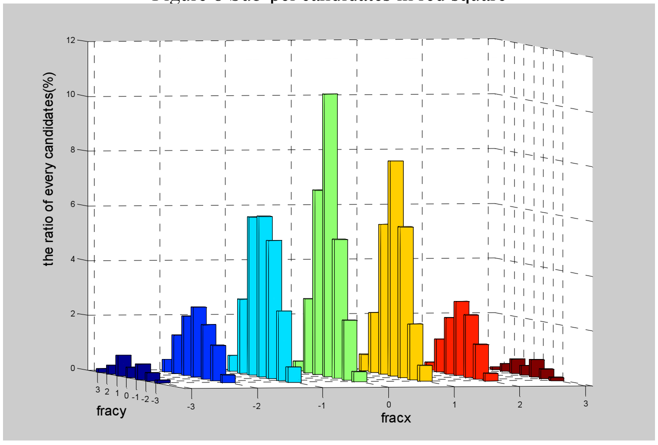

Figure. 9 The statistic of the best candidate distribution among 49 sub-pel candidates (city, soccer,football,mobile)

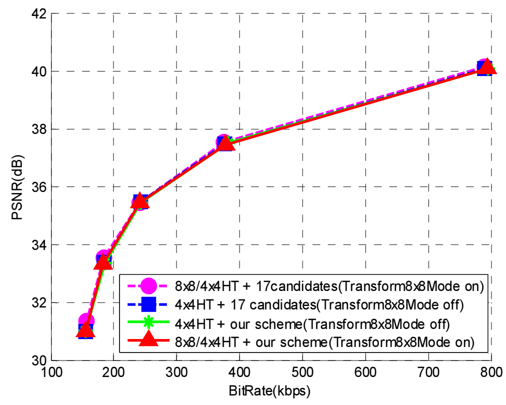

Figure 10 Rate-distortion comparison between our proposed method and original method.

TABLE II COMPARISON BETWEEN OUR PROPOSED WORK AND SOME OTHER'S WORKS

	Tsung's[8]	Yang's[10]	Huang's[7]	Wu's[2]	Our Proposed
Methods	Using linearity of HT to 1/2 and 1/4	AMS[(1)](1-7mode)	2 modes	full-mode (7modes)	full-mode (7modes)
Max. Spec.	4Kx2K@24fps	4Kx2K@30fps	4Kx4K@60fps	1920x1080@30fps	4Kx2K@30fps
Lat./MB(cycles)	--	300cycles	--	631cycles	330cyles
Operating Freq.	280MHz	300MHz	145MHz	154MHz	340MHz
Gate Count(K)	448	125.4	976.5/3	321	444.6
CMOS Tech.	TSMC 90nm	UMC 90nm	TSMC 180nm	TSMC 180nm	SMIC 130nm

(1) ASM: Adaptive Selected Modes

V. IMPLEMENTATION RESULTS AND COMPRASION

A. *Implementation Results*

We have implemented the proposed FME in Verilog HDL and synthesized it with the SMIC 130nm CMOS library under the constraint of 350MHz (The Max. Frequency).The detailed implementation results are listed in Table III.

TABLE III IMPLEMENTATION RESULTS OF OUR DESIGN

Modules	**Gate Count**
2xInterpolators	75746
9xHalf-pel SATD(HPU)	217425
16xQuarter-pel SATD(QPU)	132409
Best Candidates Selecting	14158
Mode Decision	4505
Controller	418
Total	444648

B. *Comparison*

Table II gives out the comparison between some other's works and our work. Since we employ some parallel schemes, such as 8x8/4x4 adaptive HT, two parallel interpolators and improved two- iteration process, our implementation efficiently decreases the cycles needed by one MB. So, we can support higher performance compared with [2]. Though Huang's implementation could reach SHV specification, the number of modes is reduced to 2 modes. Reducing modes is a very efficient method to decrease the latency at the cost of higher video quality drop. Our performance is close to Yang's, but Yang's also adopts the method of reducing modes for saving hardware and decreasing cycles. Tsung's[8] changes the JM algorithm and apply the linearity of HT to half-pel and quarter-pel refinement. Their design still needed high parallelism, so the cost of hardware is still very high under the 90nm technology. Our implementation under the 130 nm technology can support QFHD 3840×2160 @30fps at the frequency of 320MHz and 4Kx2K@30fps at the frequency of 340MHz.

VI. CONCLUSION

In this paper, we present a full-mode FME architecture. The architecture can support 8x8/4x4 adaptive Hadamard Transform (ABHT). Compared with conventional architectures based on 4x4 HT, our architecture can reduce computational complexity and increase the parallelism. In addition, we also exploit the linearity of HT to the quarter-pel refinement. This method can cut down the long latency caused by the second iteration efficiently. Using the SMIC 130nm technology, the proposed FME design cost 444.6K gates with the maximum working frequency of 350MHz. Our design can meet the requirement of encoding QFHD 3840×2160 @30fps at the frequency of 320MHz.

REFERENCES

[1] T. C. Chen, Y. W. Huang, and L. G. Chen, "Fully utilized and reusable architecture for fractional moiton estimation of h.264/avc," Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing, 2004, pp.V9-V12

[2] C. Y. Wu, C. L. Wu, Y. L. Lin, "A high-performance three-engine architecture for h.264/avc fractional motion estimation," IEEE Trans. Very Large Scale Integration(VLSI) Systems, vol.18, 2010, pp.662-666.

[3] C. Q. Yang, S. S. Goto, T. S. Ikenaga, "High performance VLSI architecture of fractional motion estimation in h.264 for hdtv," in Proc. IEEE Int. Symp. Circuits and Systems, 2006, pp.2605-2608.

[4] Y. K. Lin, C. C. Lin, T. Y. Kuo, and T. S. Chang, "A hardware-efficient h.264/avc motion-estimation design for high-definition video," IEEE Trans. Circuits and Systems, vol. 55, 2008, pp.1526-1533.

[5] T. Y. Kuo, Y. K. Lin, T. S. Chang, "SIFME A single iteration fractional-pel motion estimation algorithm and architecture for hdtv sized h.264 video coding," IEEE Int. Conf. Acoustics, Speech and Signal Processing, 2007, pp.I-1185-I1188.

[6] N. Th. Ta, J. R. Ch, "High performance fractional motion estimation in h.264/avc based on one-step algorithm and 8x4 element block processing," in Journal of Image Communication, vol.26,2011, pp.85-92

[7] Y. Q. Huang, Q. liu, T. S. Ikenaga, "Highly parallel fractional motion estimation engine for super hi-vision 4kx4k@60fps," IEEE Int. Works. Multimedia Signal Processing, 2009.

[8] P. K. Tsung, W. Y. Chen, L. F. Ding, C. Y. Tsai, T. D. Chuang, L. G. Chen, "Single-iteration full-search fractional motion estimation for quad full hd h.264/avc encoding," IEEE Int. Conf. Multimedia and Expo, 2009, pp. 9-12.

[9] Y. H. Chen, T. Ch. Chen, Ch. Y. Tsai, S. F. Tsai, L. G. Chen, "Algorithm and architecture design of power-oriented h.264/avc baseline profile encoder for portable devices," IEEE Trans. Circuits and Systems, 2009, vol. 19, pp.1118-1127.

[10] C. C. Yang, K. J. Tan, Y. Ch. Yang, J. I. Guo, "Low complexity fractional motion estimation with adaptive mode selection for h.264/avc," IEEE Int. Conf. Multimedia and Expo, 2010, pp. 673-678.

[11] Z. Y. Liu, D. S. Wang, T. S. Ikenaga, "Hardware optimizations of variable block size hadamard transform for h.264/avc frext," IEEE Int. Conf. Image Processing, 2009, pp. 2701-2704