Fast Mode Decision Based on RDO for AVS High Definition Video Encoder

Xiaohan Wang*, Chuang Zhu, Haibing Yin,Wen Gao, Xiaodong Xie, and Huizhu Jia

School of Electronics Engineering and Computer Science, Peking University,
100871 Beijing, P.R. China
{xhwang,czhu,hbyin,wgao,xdxie,hzjia}@jdl.ac.cn

Abstract. In this paper, we propose a fast and effective mode decision (MD) algorithm based on rate distortion optimization (RDO) for AVS high definition video encoder. The fast algorithm is composed of two parts. Firstly, mode pre-selection based on sum of absolute difference (SAD) is employed to reduce modes for candidate so as to alleviate the dramatic throughout burden. Secondly, we adopt 4-way parallel scanning technique to reduce the cycles of each mode decision based on RDO. Theoretical analysis and experimental results show that the proposed fast algorithm can meet the needs of 720P and 1080P real-time high definition AVS video encoding. Besides, the mode pre-selection algorithm provides a similar performance to the all modes enabled algorithm. And the 4-way parallel technique using negligible extra resources increases the speed of coding bits estimation by 3.4 times than traditional techniques.

Keywords: AVS, mode decision, RDO, zigzag scan, VLC.

1 Introduction

In order to reduce transmission bandwidth and storage space, video coding standard and its application is becoming a hot spot in research. Chinese audio and video coding standard (AVS) is a new national standard for digital media applications. Its video part (AVS-P2) was formally accepted as the Chinese national standard in 2006. AVS achieves equivalent coding performance compared with H.264 with lower complexity. The AVS industry alliance is initiating the industrialization for the AVS standard. Although the implementation complexity of AVS is relatively lower than that of H.264, real-time high definition AVS video coding is still a huge challenge. Currently exclusive AVS video encoder chip is still vacant. So, hardware implementation for AVS encoder is highly desired for the industrialization process.

AVS can achieve superior rate distortion performance mainly due to adopting multiple coding modes decision based on RDO. Although RDO based mode decision in AVS improve the performance greatly, the rate and distortion cost (RDcost) function calculations for all modes are very computationally intensive. In addition, the RDcost function calculations itself takes a lot of time, especially zigzag scanning and variable

* This work was supported in part by National Basic Research Program of China (973 Program, 2009CB320904), and National Science Foundation of China (60802025, 60803013).

G. Qiu et al. (Eds.): PCM 2010, Part II, LNCS 6298, pp. 62–72, 2010.
© Springer-Verlag Berlin Heidelberg 2010

length coding (VLC) table switch. It is challenging to implement RDO based mode decision for all coding modes with moderate resources. Feature and rate distortion modes based quick mode decision algorithms has attracted intensive research recently. However, these algorithms suffer from coding performance degradation, or are not well suited for hardware implementation due to algorithm irregularity.

In this paper, we propose a fast and effective RDO based mode decision algorithm, composed of two parts, which can meet the needs of 720P and 1080P real-time high definition AVS video encoding. One is mode pre-selection based on SAD so as to alleviate the dramatic throughout burden. The other is the 4-way parallel scanning technique which can increase the speed of coding bits estimation by 3.4 times than traditional techniques so as to solve the clock bottleneck in mode decision. This technique can complete zigzag scanning and switching VLC table for an 8X8 block in 19 cycles while traditional signal-way scanning method takes more than 64 cycles.

The rest of this paper is organized as follows. Architecture of AVS Video Encoder and mode decision are shown in section 2. Mode pre-selection algorithm and its performance are issued in section 3. The 4-way parallel scanning technique and its improving performance are proposed in section 4. Finally, experimental results and conclusion are given in section 5 and section 6 respectively.

2 Architecture of AVS Video Encoder and Mode Decision

Top level architecture of AVS video encoder is shown in Fig.1.

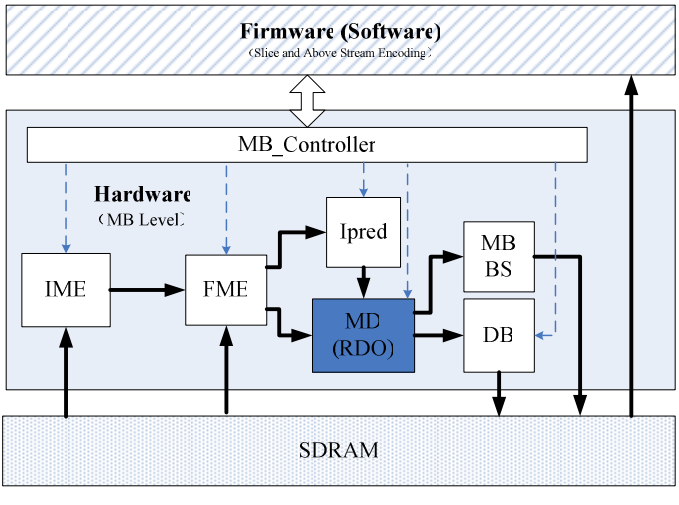

Fig. 1. Top level architecture of video encoder

Macro Block (MB) Controller module receives commands from CPU and controls the MB-level pipeline. Integer Motion Estimation (IME) and Fraction Motion Estimation (FME) modules complete motion estimation of original pixels when the frame is P/B frame. Intra frame Prediction (IPred) module predicts pixels for original

pixels when the frame is I frame or the mode is intra in P/B frame. Mode Decision module selects the best mode on the basis of RDcost. De-Block (DB) module processes reconstruction pixels from MD. Finally, Bit Stream (BS) module generates bit stream for video according to code-number from MD.

To achieve mode decision based on RDO, MD module should includes DPCM loop (DCT, Quantization, IQ, IDCT); calculates distortion sum of squared difference (SSD) on the basis of reconstructed pixels, original pixels and predicted pixels. Simultaneously, MD includes EC loop (run-length coding, coding table switch, code number finding, coding bits estimation). Finally, MD calculates

RDcost = SSD + λ*BITS and selects the best mode based on the minimum RDcost.

Architecture of RDO MD module is shown in Fig.2.

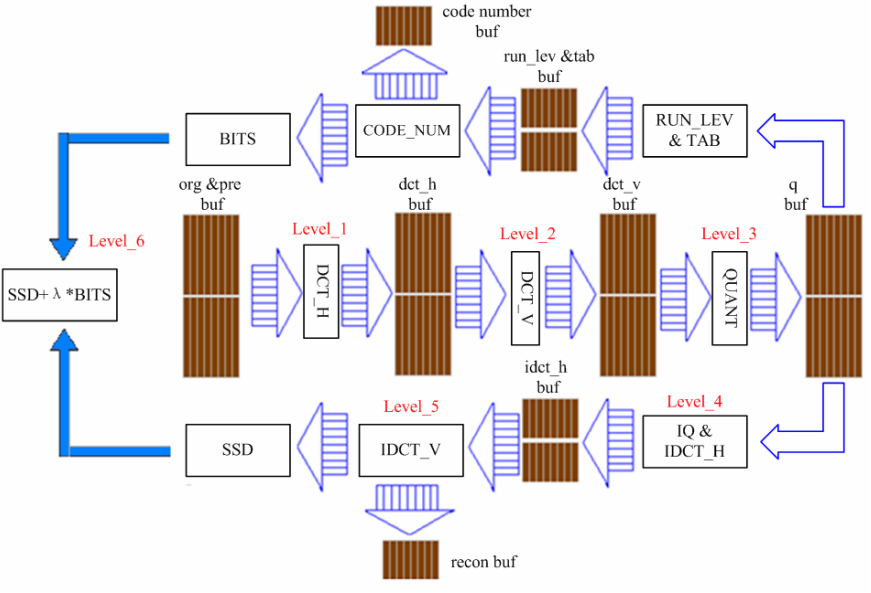

Fig. 2. Architecture of RDO MD

Table 1. Block Pipeline in MD

Level_1	horizontal-DCT of an 8X8 block
Level_2	vertical-DCT of an 8X8 block
Level_3	4-way quantization of an 8X8 block
Level_4	iquantization& horizontal-IDCT of an 8X8 block
	4-way zigzag scanning& table switch
Level_5	tical-IDCT meanwhile save the reconstructed pixels in recon buf
	code number search& coding bits estimate meanwhile save the code number in code number buf for the following BS module.
Level_6	calculation of RDcost= SSD+λ*BITS

Every block pipelines in MD are introduced in Table 1. Original and predicted data passed level_1: dct_h, level_2: dct_v, level_3: quantization forks into 2 paths: one is level_4: iquantization & idch_h, level_5: idct_v, this path generates reconstructed pixels and SSD; the other is level_4: zigzag scanning & coding table switch, level_5: code number search & bits estimation, this path saves code number and generates coding bits. Following each block pipeline there is a ping-pong buffer for block data cache.

3 Proposed Mode Pre-selection Algorithm

3.1 Original Algorithm: All Modes Enabled Algorithm

Firstly, we analyze the complexity of all modes enabled algorithm based AVS in MD module.

For I frame: We should select the best mode for every 8X8 blocks in each macro block which has 4 luma 8X8 blocks and 2 chroma 8X8 blocks. In AVS, each luma block has 5 modes which are ① vertical, ② horizontal, ③ DC, ④ downleft, ⑤ downright; each chroma block has 4 modes which are ①DC, ②horizontal, ③vertical, ④plane. So we get the computational complexity of I frame:

$$\text{I}: \ 5 \times 4 + 4 \times 2 = 28 \text{ RDcost} . \tag{1}$$

For P/B frame: Direction of motion (Forward, Backward, and Symmetric) is predicted in IME and FME. The best intra mode in P/B frame is selected on the basis of SAD. Then MD selects the best mode from ①intra, ②skip, ③16X16, ④16X8, ⑤8X16, ⑥ 8X8(direct), ⑦ 8X8(F/B/S) for each macro block. So we get the computational complexity of P/B frame:

$$\text{P/B}: \ 6 \times 7 = 42 \text{ RDcost} . \tag{2}$$

Considering the calculations, consumption of resources and cycles overall in our AVS encoder, we adopt 4-way quantization for a block. So the block-level cycle is determined by quantization which takes the maximum cycles. Quantization takes 10 cycles on account of 2 multiplying units and several adders in it. Consequently it takes $T \approx 64/4 + 10 = 25$ cycles to quantize a block. Then we should spend

$(42 + 6) * T \approx 1200$ cycles completing mode decision for a macro block before any simplification. From the analysis above we can conclude that the system clock frequency of 720P@30fps is about:

$$1280 \times 720 \times 30/256 \times 1200 = 129.6 \text{ MHz} . \tag{3}$$

The system clock frequency of 1080P@30fps is about:

$$1920 \times 1088 \times 30/256 \times 1200 = 293.7 \text{ MHz} . \tag{4}$$

As a result, the throughput burden is too high and unacceptable for the all modes enabled algorithm in P/B frame. Necessary simplification is highly desired.

3.2 Mode Statistics Experiment

In mode decision we should select the best mode of a MB in P/B frame from intra, skip, 16X16, 16X8, 8X16, 8X8. But the probability of each mode selected as the best mode is different. We test the probability using 30 frames of five 720P sequences: *City, Sailormen, Crew, Raven, ShuttleStart*, and one 1080P sequence: *BlueSky*.

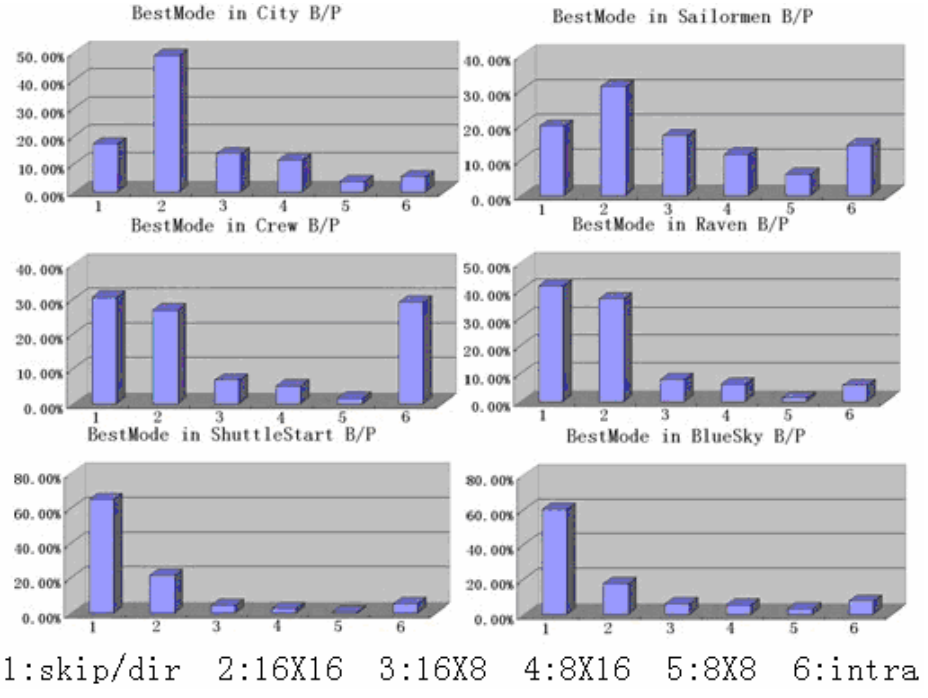

Fig. 3. Best mode statistics in P/B frame

As Fig.3 shows, skip/direct, 16X16 and intra mode selected as the best mode is more possible. Moreover the predicted pixels of these 3 modes are quite different. So we can simplify the rest 16X8, 8X16, 8X8 modes based on this experiment.

3.3 Optimal Algorithm: Mode Pre-selection Algorithm

All modes enabled mode decision based on RDO is still adopted for intra mode selection in I frames to sustain the fidelity of anchor frame of the whole GOP. And SAD based pre-selection algorithm is used for intra mode selection in P/B frames considered that there are too many candidate modes to be checked. The simplified algorithm and its complexity are analyzed below:

We select one candidate mode on the basis of SAD from ④16X8, ⑤8X16, ⑥ 8X8(direct), ⑦8X8(F/B/S).Then we select the best mode on the basis of RDO for a

macro block in P/B frame from ①intra, ②skip, ③16X16, ④candidate. After the simplification, we should spend

(24 + 6) * T ≈ 750cycles completing mode decision for a macro block in P/B frame. Therefore we can conclude that the system clock frequency of 720P@30fps is about:

$$1280 \times 720 \times 30/256 \times 750 = 81.0 \text{ MHz} . \quad (5)$$

the system clock frequency of 1080P@30fps is about:

$$1920 \times 1088 \times 30/256 \times 750 = 183.6 \text{ MHz} . \quad (6)$$

From the comprehensive analysis of all the above we can reduce the AVS encoder's data throughput significantly using the mode pre-selection algorithm while only spend extra SAD expense. The system clock frequency comparison is shown on Table 2.

Table 2. The System Clock Frequency Comparison

frequency / algorithm	720P@30fps	1080P@30fps
all modes enabled	129.6 MHz	293.7 MHz
mode pre-selection	81.0 MHz	183.6 MHz

4 Proposed Fast 4-Way Parallel Scanning Algorithm

In order to achieve 720P and 1080P real-time AVS encoder mentioned above, we should control block-level pipeline in 25 cycles. Traditional zigzag scanning spends 64 cycles completing scanning an 8X8 block owing to scanning data by data in zigzag order. It generating clock bottleneck. Therefore we put forward 4-way parallel scanning technique in this paper to remove this bottleneck, which can complete zigzag scanning and switching VLC table for an 8X8 block in 19 cycles.

4.1 First Step: 4-Way Parallel Scanning for Run, Level and VLC Table

Architecture of 4-way fast algorithm is shown in Fig.4.

The quantized coefficient is stored in a ping-pong buffer with 64 units as Fig.5 shows and each coefficient is represented by 0~63. Evenly divide the zigzag-ordered coefficients into 4 parts, in each part 16 coefficients seen as a way. In the 4-way scanning module which is shown in Fig.6, we do 4-way parallel (run, level) pair detection in reverse zigzag order, each way scans 16 coefficients, simultaneously we calculate all possible switch of VLC table number and store them in the subsequent ping-pong buffer.

Each scanning goes according to the order in the Fig.6, we put it down whatever the first coefficient is 0 or not, see it as the first (level), then continuously add the number of consecutive 0 until we encounter a non-zero, and then we can output the first (run,level) pair. If the first coefficient is 0, then we see this (run,0) as a special

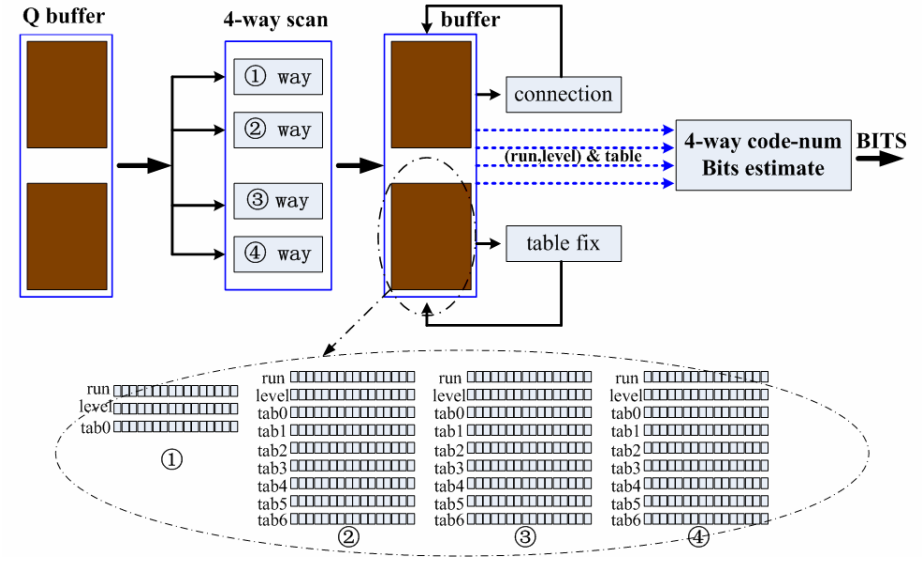

Fig. 4. Architecture of 4-way parallel scanning

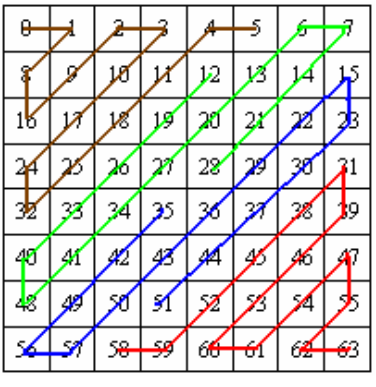

Fig. 5. Ping-pong buffer stored quantized coefficient

(run,level), its level is 0; if the first coefficient is non-zero, then this (run,level) is a correct pair. Then wherever we encounter a non-zero coefficient in scanning, we input the previous (run, level), and save this non-zero figure until we find the next non-zero coefficient. When scanning the last coefficient of each way, we output the pair. It dose not matter that the pair is not always correct, for we can rectify it in "connection" module in second step. After 16 cycles, the scanning of the quant coefficients in an 8X8 block is over, and all (run,level) pairs were stored in the subsequent buffer. When detecting the (run,level), we compute the number of VLC table simultaneously.

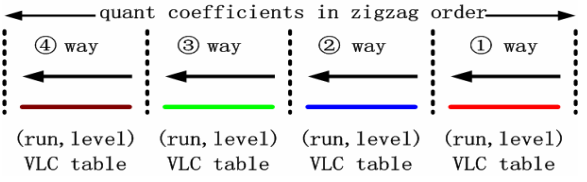

Fig. 6. 4-way scanning in reverse zigzag order

For the first way: the starting table number of the first (run,level) is always 0, then the table number of the (run,level) in the first way is the unique answer. As for the rest 3 ways: the table number of the first (run,level) is not confirmed, 0~6 is possible. We save all possible table number in a buffer. The construction of this buffer is shown in Fig.4.

4.2 Second Step: Connection and VLC Table Fix

After the scanning work is done, we begin to amend certain "run" of the buffer, which is called as "connection", meanwhile we fix the correct starting table number of each way in "table fix" module. The "connection" module can be divided into 3 steps:

Step 1: examine the interface of the fourth and the third way, that is to examine the first (run,level) of the fourth way and the last (run,level) of the third way: If level4 = 0 in the first (run,level) of the fourth way, we modify run3 in the last (run3,level3) of third way to run3+run4; if level4 ≠0, there is no need for modification. If there is no non-zero coefficient in the third way, we modify the special pair (16,0) to (16+run4,0).

Step2: examine the interface of the third way and the second way with a method similar to step 1.

Step3: examine the interface of the second way and the first way with a method similar to step 1.

When operating the connection, we simultaneously calculate the starting table number of each way. It also has 3 steps:

Step 1: examine the corresponding table number of the last (run,level) of the first way . This table number is the starting table number of the second way. Supposing it's *X*, then we can choose the storing unit named tab*X* in the buffers tab0~tab6 of the second way. It is related with the first way.

Step2: examine the corresponding table number of the last (run,level) in the buffer tab*X* of the second way. This is the starting table number of the third way, which is related with both the first way and the second way.

Step 3: fix the starting table number of the fourth way, with a method similar to the above.

Even in the worst case: 65 (run,level) pairs including EOB, 4-way method only takes about 20 cycles to finish (run,level) detection and Exp-Golomb coding table search for an 8X8 block while traditional scanning method triples the cycles. And in resources consumption, it has only 3x5 extra buffers. Each unit in extra buffer only takes 3 bits given the table number. Therefore, we can constrain the cycle of block-level pipeline in T = 25cycles using negligible extra resource so as to fulfill the needs of real-time AVS encoder.

5 Experimental Results

5.1 Coding Performance Comparison

The fast mode decision algorithm are tested using 10 sequences *Raven, Crew, Sailormen, Sheriff, Night, Spincalendar, Cyclists, Optis, City* and *Harbour* of 720P format in 30Hz. IPBBPBB format with GOP length 15 are used. As Fig.7, Fig.8 and Fig.9 show, the blue line marked with "X" stands for RDO mode decision algorithm with all modes enabled. The fast RDO mode decision algorithm based on pre-selection and 4-way method is represented by pink line marked with "■". And the pink dotted line marked with "★" stands for mode decision algorithm not using RDO but SAD. These 3 algorithms are compared with luma PSNR in the same bit rate.

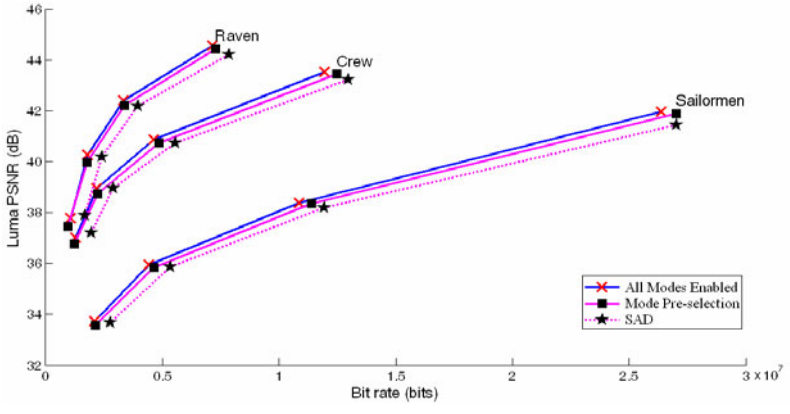

Fig. 7. Luma PSNR of 3 sequences *Raven, Crew, Sailormen*

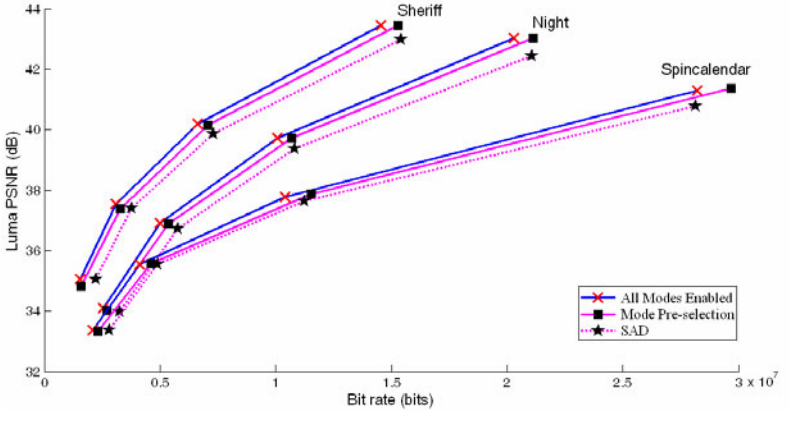

Fig. 8. Luma PSNR of 3 sequences *Sheriff, Night, Spincalendar*

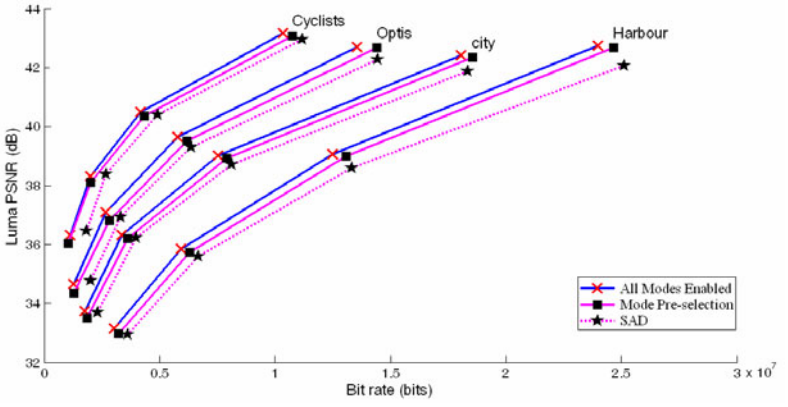

Fig .9. Luma PSNR of 4 sequences *Cyclists, Optis, City, Harbour*

We can see clearly that the performance of our fast algorithm is very close to that of all modes enabled algorithm while the performance of SAD based algorithm declines significantly. Therefore, we can conclude that the fast algorithm alleviates the throughout burden, whereas the performance degradation is negligible.

5.2 Simulation and Resources Evaluation

The proposed 4-way parallel scanning algorithm is implemented using Verilog language. It is simulated on large amount of data using Modelsim, and the simulating results are absolutely consistent with the AVS standard. Meanwhile, it is synthesized with ISE 10.1.03 using Xilinx FPGA -Vertex5 XC5VLX330, and the synthesis results is illustrated in Fig.10, where the frequency of synthesis is 246.28MHz which is enough to satisfy the real-time high definition video coding requirement. The simulation results show that our Verilog code is functionally identical with the C-code model on AVS standard.

```
Selected Device: 5vlx330ff1760-2
----------------------------------------
Slice Logic Utilization:
Number of Slice Registers:          1074  out of  207360   0%
Number of Slice LUTs:               1756  out of  207360   0%

Specific Feature Utilization:
Number of Block RAM/FIFO:      7  out of   288   2%
Number of BUFG/BUFGCTRLs:      1  out of   32    3%

Timing Summary:
Minimum period: 4.060ns (Maximum Frequency:  246.275MHz)
Minimum input arrival time before clock:      4.229ns
Maximum output required time after clock:     0.396ns
Maximum combinational path delay:             1.772ns
```

Fig. 10. Synthesis Results of The 4-way parallel algorithm

6 Conclusions

In this paper, a fast and effective mode decision algorithm is proposed for AVS high definition video encoder. We adopt 2 methods to accelerate mode decision and meet the needs of 720P and 1080P real-time video encoding. One method is reducing the candidate modes based on RDO, the other is reducing the cycles of each mode decision based on RDO. Experimental results have shown that this fast algorithm alleviates the throughout burden, meanwhile the performance degradation and extra resources are negligible. This fast algorithm is well suited for both AVS and H.264 high definition video encoder. Currently, AVS high definition video encoder based the fast mode decision algorithm was verified successfully on Xilinx FPGA -Vertex5 XC5VLX330. In the future work, we will optimize the mode decision algorithm and AVS encoder by increasing system frequency and reducing cycles of one MB to meet the needs of real-time high definition video encoding better.

References

1. Advanced coding of audio and video Part 2, Video: AVS Standard (2006)
2. Liu, Z., Song, Y., Goto, S., et al.: HDTV 1080P H.264/AVC Encoder Chip Design and Performance Analysis. IEEE Journal of Solid-state Circuits 44(2) (2009)
3. Xu, L., Deng, L., Ji, X., Peng, X., Gao, W.: Hardware Architecture of AVS Entropy Encoder. IEEE Transactions on Consumer Electronics (2008)
4. Su, Y., Sun, M.T.: Encoder Optimization for H.264/AVC Fidelity Range Extensions, pp. 12–15. VCIP (2005)
5. Chang, H.C., Chen, L.G., Chang, Y.C., Huang, S.C.: A VLSI architecture design of VLC encoder for high data rate video/image coding. In: Proc. IEEE Int. Symp. Circuits and Systems, vol. 4, pp. 398–401 (1999)
6. Novell, M., Molloy, S.: VLSI implementation of a reversible variable length encoder/decoder. In: Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing, vol. 4, pp. 1969–1972 (1999)
7. Chen, T.C., Chien, S.Y., Huang, Y.W., Tsai, C.H., Chen, C.Y., Chen, T.W., Chen, L.G.: Analysis and Architecture Design of an HDTV720p 30 Frames/s H.264/AVC Encoder. IEEE Trans. Circuit and syst. Video technol. 16(5) (2006)
8. Dui, Q., Zhu, D., Ding, R.: Fast Mode Decision For Inter Prediction in H.264. In: Proc. of IEEE Int. Conf. Img. Proc. (ICIP 2004), pp. 119–122 (2004)
9. Yang, E.H., Wang, L.: Joint optimization of run-length coding, Huffman coding and quantization table with complete baseline JPEG decoder compatibility. U.S. Patent Application (2004)
10. Pan, F., Lin, X., Susanto, R., Lim, K.P., Li, Z.G., Feng, G.N., Wu, D.J., Wu, S.: Fast mode decision for intra prediction. In: Joint Video Team (JVT) JVT-G013 (2003)