A 16-pixel Parallel Architecture with Block-level/Mode-level Co-reordering Approach for Intra Prediction in 4kx2k H.264/AVC Video Encoder

Huailu Ren[2], Yibo Fan[1*], Xinhua Chen[2], Xiaoyang Zeng[1*]

[1] State Key Lab of ASIC and System, Fudan University, Shanghai, China

[2] College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao, China

[*] {fanyibo, xyzeng}@fudan.edu.cn

***Abstract*—Intra prediction is the most important technology in H.264/AVC intra frame encoder. But there is extremely complicated data dependency and an immense amount of computation in intra prediction process. In order to meet the requirements of real-time coding and avoid hardware waste, this paper presents a parallel and high efficient H.264/AVC intra prediction architecture which targets high-resolution (e.g. 4kx2k) video encoding applications. In this architecture, the optimized intra 4x4 prediction engine can process sixteen pixels in parallel at a slightly higher hardware cost (compared to the previous four-pixel parallel architecture). The intra 16x16 prediction engine works in parallel with intra 4x4 prediction engine. It reuses the adder-tree of Sum of Absolute Transformed Difference (SATD) generator. Moreover, in order to reduce the data-dependency in intra 4x4 reconstruction loop, a block-level and mode-level co-reordering strategy is proposed. Therefore, the performance bottleneck of H.264/AVC intra encoding can be alleviated to a great extent. The proposed architecture supports full-mode intra prediction for H.264/AVC baseline, main and extended profiles. It takes only 163 cycles to complete the intra prediction process of one macroblock (MB). This design is synthesized with a SMIC 0.13µm CMOS cell library. The result shows that it takes 61k gates and can run at 215MHz, supporting real-time encoding of 4kx2k@40fps video sequences.**

I. INTRODUCTION

H.264/AVC is the latest video coding standard developed by the Joint Video Team(JVT) of ISO/IEC MPEG and ITU-T VCEG. Its new features mainly include variable block-size motion compensation, arithmetic entropy coding, in-loop deblocking filter and directional spatial prediction for intra coding, etc. These features can help H.264/AVC standard get over 50% bit rate savings while achieving the similar quality compared to prior standards [2].

Performance improvement of H.264/AVC mainly comes from the enhanced prediction part. Unlike previous standards, intra prediction is allowed in all types of slices of H.264/AVC, so it should be performed not only for intra macroblocks (I-MB) but also for inter macroblocks (P-MB). When motion estimation for P-MB fails to find a good match, intra prediction may still have a chance to get a good result. Analysis in [6] shows that the number of I-MBs is significantly larger than that of P-MBs in the case of rapid motion scenes. In the field of still image compression, H.264/AVC intra encoding has better quality and lower complexity than JPEG2000 [3]. So intra prediction is a very important part of H.264/AVC encoder. But it has heavy computational burden and high data dependency due to its supporting variable-block size and tens of prediction modes. Intra prediction unit and intra mode decision unit together can occupy 80% of the computation time of the entire intra compression process [3].

Several VLSI architectures are proposed for H.264 intra prediction to achieve the goal of supporting real-time encoding high-resolution videos. To speed up the intra predict

tion, several previous designs adopt hardware parallelism strategies. Architecture proposed in [9] can predict 8 pixels per cycle, and another architecture presented in [4] has the ability to process 48 pixels per cycle by using two prediction engines. But the data-dependency among luma 4x4 blocks decreases the hardware utilization and increases the latency time.

There are mainly three methods to resolve the data-depend-ency problem. The most common one is to optimize the 4x4 block processing order [10][11]. It can resolve this problem to some extent, but it is still not efficient enough. Another method is to perform prediction with original boundary pixels. But it will bring obvious image quality loss. The third method is to ignore several prediction modes [8]. It can reduce image quality too. Besides these methods, interlacing intra 4x4 prediction with intra 16x16 and/or chroma 8x8 prediction can efficiently improve hardware utilization [3][8][9], but actually it does not resolve the data-dependency problem. It just reduces bubble cycles in the intra prediction process, and the bottleneck (intra 4x4 prediction) is not changed.

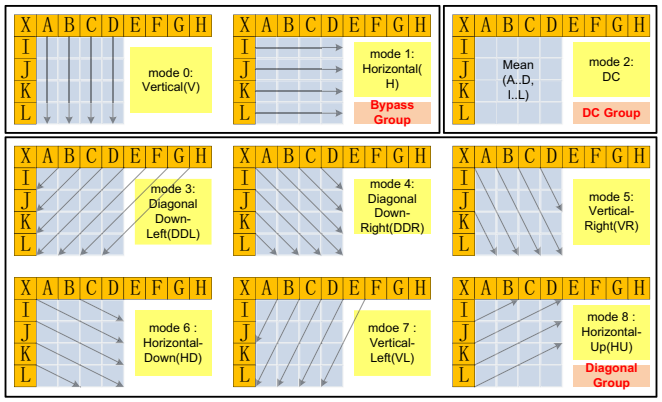

Fig. 1. Intra 4x4 prediction modes

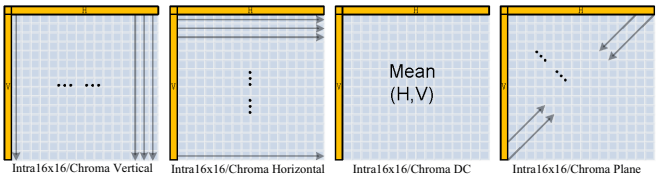

Fig. 2. Intra 16x16/Chroma 8x8 prediction modes

978-1-4673-0772-7/12/$31.00 ©2012 IEEE

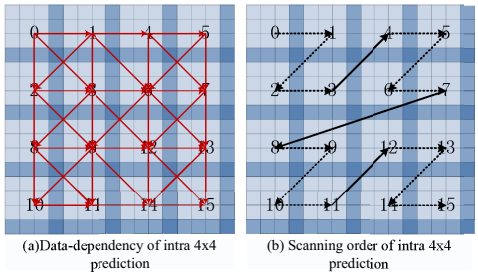

Fig. 3. Data-dependency and Scanning order of H.264/AVC intra 4x4 prediction

In this paper, a high efficient intra prediction hardware architecture is proposed. It supports 16-pixel parallel processing without increasing heavy hardware cost. We analyze the data-dependency among luma 4x4 blocks and propose a block-level/mode-level co-reordering method to alleviate this problem.

The remainder of this paper is organized as follows. In Section II, we briefly analyze H.264/AVC intra prediction encoding process. Then we propose our intra prediction hardware architecture and scheduling method in Section III. In section IV, experimental results are presented. Finally, conclusions are drawn in Section V.

II. Overview Of Intra Prediction

A. Intra Prediction

In a 4:2:0 image format, there are one 16x16 luma component and two 8x8 chroma components in one MB of H.264/AVC. Luma component can be predicted as either sixteen 4x4 sub- blocks (i.e. intra 4x4 prediction) or one single 16x16 block (i.e. intra 16x16 prediction), while both chroma components are processed as one 8x8 block. H.264/AVC intra prediction has 17 prediction modes: nine for intra 4x4 blocks, four for intra 16x16 block and four for chroma blocks. Both chroma blocks (Cb and Cr) of one MB use the same prediction modes [1].

All processing blocks in intra prediction are produced based on the boundary reconstructed pixels of neighboring blocks (i.e. reference samples, marked with deeper color in Fig. 3).

Fig. 1 illustrates the nine prediction modes of intra 4x4 prediction. All except intra 4x4 DC mode each has a prediction direction, and we can call them “prediction modes”. The intra 4x4 prediction algorithm will be analyzed in Section III. Intra 16x16 prediction has four modes as shown in Fig. 2. All except intra 16x16 Plane mode are similar to that of intra 4x4 prediction. Plane mode works well in areas of smoothly varying image and it is the most complicated prediction mode. Both 8x8 chroma components are predicted by one same mode. The four chroma prediction modes are the same with that of intra 16x16 prediction except that the block size and the order of mode numbers are different.

B. Data Dependency

The data-dependency among luma 4x4 blocks is illustrated in Fig. 3(a). Fig. 3(b) illustrates the zig-zag scanning order for luma 4x4 blocks in the H.264/AVC intra encoding process. The dashed line in Fig. 3(b) indicates that the prediction engine needs a period of waiting time here when it jumps to process next block. For example, the dashed line arrow pointing to block 6 means that we cannot perform the prediction of block 6, untill the reconstructions of block 1, 4, 5, 3(as shown in Fig. 3(a)) are done. As can be seen from the Fig., 12 out of 16 blocks have this kind of data-dependency (For block 4, 8, 12, intra prediction can be pipelined).

To achieve higher throughput and hardware utilization, a pipelined and parallel architecture is needed. But in the conventional scanning order, the block to be predicted has a high chance to use the reconstructed pixels of its available neighboring blocks. It takes a long latency to wait for the completion of the reconstruction loop. This makes the use of pipeline technology a big challenge. The proposed method to alleviate this problem will be discussed in Section III. In the case of 4kx2k@30 fps videos, one MB have to be processed in 203 cycles. And as there are tens of prediction modes for H.264/AVC prediction, it is hard to encode high resolution videos in real-time even the intra prediction architecture is pipelined. So besides the pipeline technology a parallel strategy has to be adopted. The proposed optimized 16-pixel parallel architecture is proposed in section III. Note: the reconstruction process mainly includes DCT transform (T), Quantification (Q), Inverse Quantification (IQ), and Inverse DCT transform (IT).

III. Proposed Hardware Architecture

Fig. 4 shows the overall hardware architecture of this design. The major processing modules are intra 4x4 prediction engine, intra 16x16/chroma 8x8 prediction engine and SATD

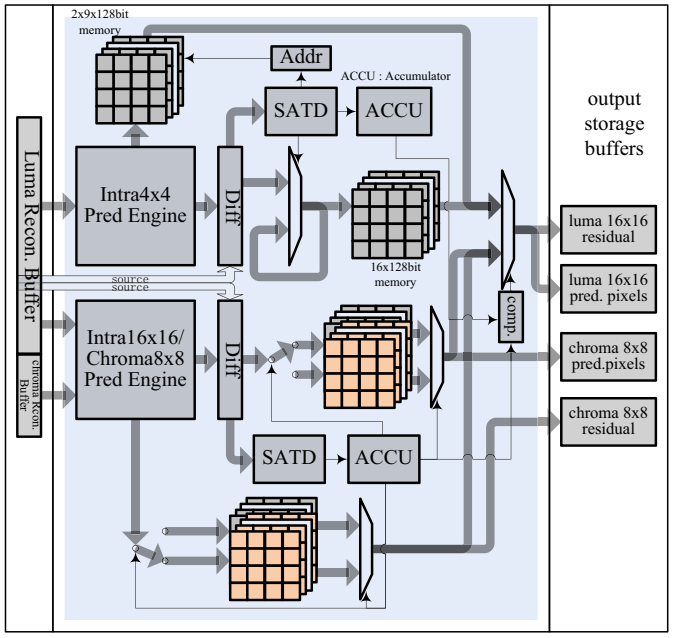

Fig. 4. Proposed architecture of intra prediction

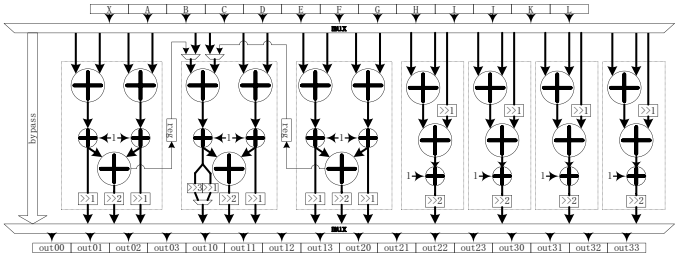

Fig. 5. Proposed architecture of intra 4x4 prediction

TABLE I. ANALYSIS OF INTRA 4X4 PREDICTION ALGORITHM

mo	mode3:DDL			mode4:DDR			mode5:VR			mode6:HD			mode7:VL			mode8:HU			mode2:DC		
de	temp value	equation	pred(yx)	temp value	equation	pred(yx)	temp value	equation	pred(yx	temp value	equation	pred(yx)	temp value	equation	pixel(yx	temp value	equation	pred(yx)	temp value	equation	pred(yx)
Stage0	t_0=A+B+1	-	-	t_0=L+K+1	-	-	t_0=K+J+1	S1(t_3)	00,21	t_0=L+K+1	S1(t_0)	30	t_0=A+B+1	S1(t_0)	00	t_0=L+L+1	S1(t_1)	12,20	t_0=L+K+1	S2(t_4+t_5)	00,01,02,03 10,11,12,13 20,21,22,23 30,31,32,33
	t_1=B+C+1	-	-	t_1=K+J+1	-	-	t_1=J+I+1	S1(t_4)	01,22	t_1=K+J+1	S1(t_1)	20,32	t_1=B+C+1	S1(t_1)	01,20	t_1=L+K+1	S1(t_2)	02,10	t_1=J+I+1		
	t_2=C+D+1	-	-	t_2=J+I+1	-	-	t_2=I+X+1	S1(t_5)	02,23	t_2=J+I+1	S1(t_2)	10,22	t_2=C+D+1	S1(t_2)	02,21	t_2=K+J+1	S1(t_3)	00	t_2=A+B+1		
	t_3=D+E+1	-	-	t_3=I+X+1	-	-	t_3=X+A+1	S1(t_6)	03	t_3=I+X+1	S1(t_3)	00,12	t_3=D+E+1	S1(t_3)	03,22	t_3=J+I+1			t_3=C+D+1		
	t_4=E+F+1	-	-	t_4=X+A+1	-	-	t_4=A+B+1	-	-	t_4=X+A+1	-	-	t_4=E+F+1	S1(t_4)	23	-	-	-	t_4=t_0+t_1		
	t_5=F+G+1	-	-	t_5=A+B+1	-	-	t_5=B+C+1	-	-	t_5=A+B+1	-	-	t_5=F+G+1	-	-	-	-	-	t_5=t_2+t_3		
	t_6=G+H+1	-	-	t_6=B+C+1	-	-	t_6=C+D+1	-	-	t_6=B+C+1	-	-	-	-	-	-	-	-	-		
	t_7=H+H+1	-	-	t_7=C+D+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-		
Stage1	-	S2(t_0+t_1)	00	-	S2(t_0+t_1)	30	-	S2(t_0+t_1)	30	-	S2(t_0+t_1)	31	-	S2(t_0+t_1)	10	-	S2(t_0+t_1)	13,21	-	S2(t_0+t_1)	00,01,02,03 10,11,12,13 20,21,22,23 30,31,32,33
		S2(t_1+t_2)	01,10		S2(t_1+t_2)	20,31		S2(t_1+t_2)	20		S2(t_1+t_2)	21,33		S2(t_1+t_2)	11,30		S2(t_1+t_2)	03,11		S2(t_2+t_3)	
		S2(t_2+t_3)	02,11,20		S2(t_2+t_3)	10,21,32		S2(t_2+t_3)	10,31		S2(t_2+t_3)	11,23		S2(t_2+t_3)	12,31		S2(t_3+t_3)	01		-	
		S2(t_3+t_4)	03,12,21,30		S2(t_3+t_4)	00,11,22,33		S2(t_3+t_4)	11,32		S2(t_3+t_4)	01,13		S2(t_3+t_4)	13,32		-	-	-	-	
		S2(t_4+t_5)	13,22,31		S2(t_4+t_5)	01,12,23		S2(t_4+t_5)	12,33		S2(t_4+t_5)	02		S2(t_4+t_5)	33		-	-		-	
		S2(t_5+t_6)	23,32		S2(t_5+t_6)	02,13		S2(t_5+t_6)	13		S2(t_5+t_6)	23		-	-		-	-		-	
		S2(t_6+t_7)	33		S2(t_6+t_7)	03		-	-		-	-		-	-		-	-		-	

Note: Sn(x) stands for (x)>>n

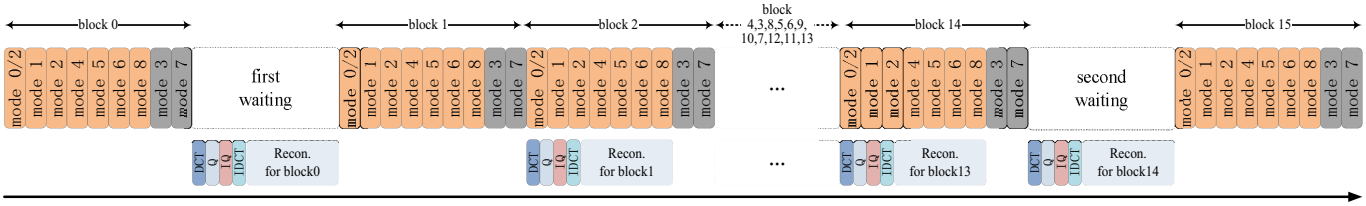

Fig. 6. Proposed block-level/mode-level co-reordering method

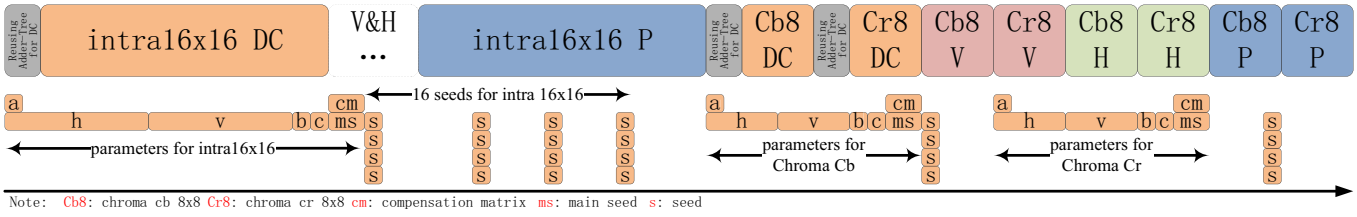

Fig. 7. Timing diagram of intra 16x16 and chroma 8x8 prediction

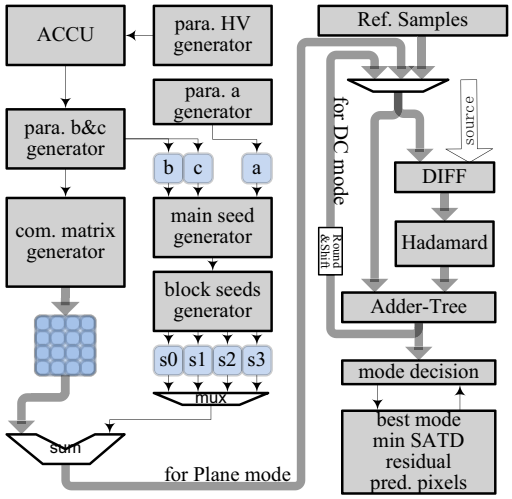

Fig. 8. Proposed architecture of intra 16x16/chroma 8x8 prediction

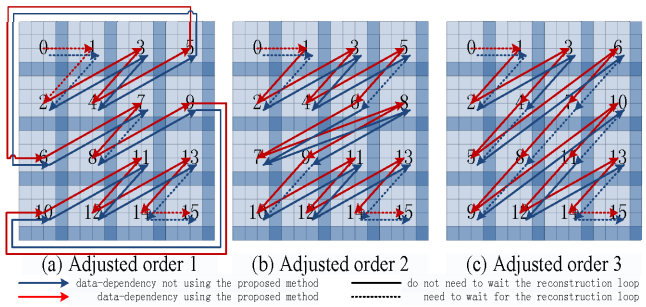

Fig. 9. Adjusted scanning orders of intra 4x4 prediction

generator. Intra 4x4 prediction engine computes nine intra 4x4 prediction modes for luma, while intra 16x16/chroma 8x8 prediction engine computes four intra 16x16 prediction modes for luma and four chroma 8x8 prediction modes for chroma. Intra 4x4 prediction is the bottleneck of the whole design. We adopt two data paths taking same cycles to perform intra prediction: one is for intra 4x4 prediction; the other is for intra 16x16 and chroma 8x8 prediction.

The two prediction engines load reference samples from luma/chroma reconstruction buffer and generate prediction pixels. Then the residuals of prediction pixels and original pixels are sent to SATD generator to calculate a SATD cost. The best prediction pixels and residual values are chosen by the minimum SATD cost. Their intermediate values are stored in several independent memories and when the final mode decision is done they are sent to the output storage buffers. For intra 4x4 prediction, the prediction pixels for nine modes are all stored in a two port memory. After intra 4x4 mode decision is done, only the best prediction pixels are read from the memory and used as inputs of reconstruction module. For intra 16x16 prediction, several memories are used in pairs to store the values (prediction pixels and residuals) generated by current prediction mode and intermediate best prediction mode. When SATD cost of current prediction mode is smaller than that of intermediate best mode, switch operation between two memories in one pair is performed.

A. Intra 4x4 Prediction Architecture

In this paper, we design a reconfigurable intra 4x4 prediction architecture, which can predict 16 pixels in parallel for all intra 4x4 prediction modes at a low hardware cost. According

to the equations defined in intra 4x4 prediction part of H.264/AVC standard, we divide intra 4x4 prediction modes into 3 groups: bypass group, dc group and diagonal group, as shown in Fig. 1. The bypass group includes intra 4x4 Vertical and Horizontal modes. In this group, prediction pixels are generated by copying the reconstructed pixels from upper or left blocks and the process does not require any computation. The second group only covers DC mode. In this case, prediction pixels are the mean of the upper and/or left reference samples.

The other 6 modes fall into the diagonal group. Equations in this group are composed of 2-tap and 3-tap filters. These equations have two features. First, most of their final prediction results are reusable. For example, for intra 4x4 DDL mode, pred(03)(prediction pixel in the 1st column, 4th row of one 4x4 block), pred(12), pred(21), pred(30) of a block are the same value. And we find there are no more than 10 different values in these 6 modes (half of the 6 modes only have 6 different values). Second, intermediate values of these 2-tap and 3-tap filters are reusable. For example, for intra 4x4 VL mode, make t0=A+B+1, t1=B+C+1, (X,I~K,A~H are reference samples of current block, as shown in Fig. 1), then pred(00)=(t0)>>1, pred(01)=(t1)>>1, pred(10)=(t0+t1)>>2, the intermediate value t0 and t1 can be reused. Based on the above two features, we summarize all the reusable intermediate and final values (as shown in TABLE I) in intra 4x4 prediction equations. Then we design a high efficient hardware architecture of intra 4x4 prediction, as shown in Fig. 5. In this architecture, we assemble two 2-tap filters and one 3-tap filter into one 4-tap filter that can output 3 prediction values simultaneously by reusing the intermediate values. The whole intra 4x4 prediction architecture is composed of three 4-tap filters and four 3-tap filters. It can fully reuse all the reusable values in intra 4x4 prediction algorithm and can process all prediction modes of diagonal group in one cycle.

Moreover, this architecture also supports intra 4x4 DC mode. For example, in the case that only the upper block is available, make t0=(A+B+1), t1=(C+D+1), then the dc prediction value is equal to (t0+t1)>>1 (It is similar in the case when only the left block is available). When both the upper and left blocks are available, the prediction is completed in two cycles. In the first cycle (Vertical mode is calculated in the same cycle), intermediate values are generated using two 4-tap filters. They are sent to another 4-tap filter to calculate the final result (the process is shown in TABLE I and Fig. 5). So, all the intra 4x4 prediction modes can be processed by using only three 4-tap filters and four 3-tap filters.

In [5], similar optimization of intra 4x4 prediction is done. But they do not separate 3-tap filters from 4-tap filters, causing the architecture is not optimized completely. Finally they used 33 adders for the 6 diagonal modes and do not take DC mode in consideration while our design can process all intra 4x4 prediction modes requiring only 27 adders. Moreover, compared with 4-pixel parallelism architecture in [3], we only use extra 5 adders to achieve four times increasing in throughout (They do not regard “round” as an add operation).

B. Intra 16x16 Hardware Architecture

Since intra 16x16 prediction is not the critical part of the whole architecture, this module is designed principally to save area cost. For example, the sum of reference samples is obtained by adder-tree which can also generate SATD costs, therefore, little extra circuit is introduced for intra 16x16 DC mode (as show in Fig. 6).

Intra 16x16 Vertical and Horizontal modes are similar to that of intra 4x4 prediction without needing any computation. There are two methods to implement intra 16x16 plane pixel generator. The first one is to compute the prediction pixel values fully referring to the equations in H.264/AVC [5]. In the second method, several seed pixels are calculated first, then they are added by parameter b or c (the meanings of these para-meters can be found in [1]) to generate the other prediction pixels [3]. This paper adopts the second method. What is different in our design is that we produce 4 seed pixels at a time and generate sixteen pixels in parallel. The intra 16x16 prediction architecture can also perform chroma 8x8 prediction. The schedule diagram of intra 16x16 and chroma 8x8 prediction is given in Fig. 7.

C. Block-level/Mode-level Co-reordering

Data dependency among luma 4x4 blocks makes pipeline execution of intra prediction and reconstruction a big challenge and causes performance bottleneck of the whole system. Rearranging the scanning order of intra 4x4 blocks can improve the situation to some extent. Reference [10] compares ten alternative scanning orders based on rearranging order of intra 4x4 blocks, and finds that even in the best order, intra 4x4 prediction still requires waiting 5 times (as shown in Fig. 9(a)).

One intra 4x4 block is predicted mainly referring its upper, left and upper-right neighboring blocks. Rearranging scanning order in block level can avoid one block’s dependency from its upper and left neighboring blocks. For example, in the two scanning orders shown in Fig. 9(a) and Fig. 9(b), all except block 1 and block 15 do not need to wait for the reconstruction of its upper or left neighboring blocks. Note: because mode decision between intra 4x4 and intra 16x16 modes should be made after the completion of intra 4x4 and intra 16x16 prediction, luma 4x4 blocks between different MBs cannot be processed in pipeline and block 1 and block 15 in a MB have to wait the reconstruction of its left blocks. So we draw a conclusion, rearranging scanning order in block level cannot completely resolve the problem of data dependency among intra 4x4 blocks because it cannot avoid one block’s dependency caused by its upper-right block.

We find that only mode 3 (intra 4x4 DDL) and mode 7 (intra 4x4 VL) of intra 4x4 prediction need upper-right reconstruction pixels. In conventional mode order (as shown in Fig. 1), when the fourth mode is processed, the upper-right reconstruction pixels are needed. But if we rearrange the scanning order of intra 4x4 prediction modes by processing mode 3 and mode 7 at last then we can perform intra 4x4 prediction early for the other 7 modes without waiting for the reconstruction of upper-right block (as shown in Fig. 6).

This block-level/mode-level co-reordering method can decrease the data dependency among luma 4x4 blocks more efficiently. This method is most appropriate for four-pixel parallelism system for there is enough time left to complete the reconstruction loop. It takes 4 cycles to finish prediction of one mode, so there remains 28 cycles before upper-right reconstruction pixels are needed. Assuming T/IT takes 8 cycles and Q/IQ takes 4 cycles, there are still 4 slack cycles.

TABLE II. GATE COUNTS FOR EACH COMPONENT

Function Module	Gate Count
Intra 4x4 Prediction	5441
Intra 16x16 Prediction	28058
Hadamard	8928
Adder-tree	6238
Others(ctrl, mux, diff …)	12280
Total	60945

Fig. 9 illustrates the optimized results by using the schedule method proposed in this paper. In adjusted scanning order 1, 2 and 3, the previous reordering method requires to wait 5, 6 and 9 times, while our method only needs waiting 2 times.

In this design, one intra 4x4 mode is processed in one cycle, and there are 3 cycles latency for the insertion of registers in the data path. There are 4 cycles left to early perform intra 4x4 prediction. The work [12] presents a dedicated architecture for the transform and quantization loop of H.264/AVC intra prediction. This work shows that the reconstruction loop can be completed in 4 cycles when the architecture running at 129MHz. In this paper, we assume the reconstruction loop needs 7 cycles to finish. In this case, we can still save 21 cycles, compared to the best block-level reordering method. There are 134 modes at most for intra 4x4 prediction, because not all the "prediction modes" are available. After the optimization of timing schedule, it takes only 163 cycles to complete the prediction of one MB.

IV. EXPERIMENTAL RESULTS

The proposed architecture is designed with Verilog HDL and synthesized with a SMIC 0.13μm CMOS cell library. The gate count for each component is listed in TABLE II. This design takes total 61k gates and can run at 215MHz. TABLE III shows the comparison with previous full-mode related works. In [3] and [4], only the prediction parts of their designs are taken in consideration.

The results show that our design can achieve performance for real-time processing of different video resolutions (4kx2k, 1080p, 720p) at a lower operation frequency and relatively smaller area size. [3]'s work performs intra 16x16/chroma prediction when the intra 4x4 prediction engine is in idle time, but the bottleneck of the whole architecture (i.e. intra 4x4 prediction) is not changed. So it can only support real-time encoding for low resolution videos. Architecture in [4], adopting two prediction engines, can support the encoding of 1080HD (1920x1088) video sequences at 30 frames per second (fps) , but the bubble cycles in intra 4x4 reconstruction loop is not reduced and the hardware utilization is low. Even [5] adopts mode-parallel approach, but the data dependency problem in intra 4x4 reconstruction loop is also not resolved.

TABLE III. COMPARISONS WITH RELATED WORKS

		This work	[3]	[4]	[5]	[6]	[7]
Gate count		61k	26k	-	42k	-	22k
CMOS Technology		SMIC 130nm	TSMC 250nm	TSMC 130nm	TSMC 180nm	-	TSMC 180nm
Parallel(pixel)		16	4	-	4	4	4
Memory bits		12544	5120	60000	44160	-	-
Max. Freq.		215M	130M	142M	156M	300M	75M
Cycles/MB		163	1184	562	451	1758	678
Freq. for (Hz)	4kx2k	158M	-	-	-	-	-
	1080p	40M	-	138M	110M	-	-
	720p	18M	54M	61M	48M	300M	75M

Although the proposed 16-pixel parallel architecture with block-level/mode-level co-reordering approach consumes a larger area cost, the hardware utilization or our work is relatively higher than that of previous works. As can be seen in TABLE III, our architecture is 7.3x, 2.8x and 4.2x faster than [3], [5] and [7]'s, while the gate count of our design is only 2.3x, 1.4x and 2.8x larger than that of [3], [5] and [7], respectively. The area of intra 4x4 prediction architecture of our work, which is fully optimized, is only about 1/5th, 1/8th and 1/4th of that of [3], [5] and [7]. And there is still some room for the optimization of our architecture for intra 16x16/chroma prediction. Besides, only our architecture supports the real-time encoding of 4kx2k videos.

V. CONCLUSION

This paper proposes a low latency and high throughout hardware architecture for intra prediction, supporting real-time encoding of 4kx2k@40fps. A 16-pixel parallel high utilization intra prediction architecture is proposed in this paper. The proposed block-level/mode-level co-reordering method efficiently decreases the latency caused by intra 4x4 reconstruction loop. This presented architecture is described in Verilog HDL and synthesized into a SMIC 0.13μm CMOS cell library. It can process one MB within 163 cycles and achieve a maximum operation frequency of 215MHz.

REFERENCES

[1] ITU-T Rec. H.264 and ISO/IEC 14496-10:2005 (E) (MPEG-4 AVC), "Advanced video coding for generic audiovisual services", 2005

[2] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard", IEEE TCSVT, Jul. 2003, pp.560-576.

[3] Y.-W. Huang, B.-Y. H, T.-C. Chen, and L.-G. Chen, "Analysis, Fast Algorithm, and VLSI Architecture Design for H.264/AVC Intra Frame Coder", IEEE TCSVT, Mar. 2005, pp.378-401.

[4] H.-C. Kuo, Y.-L. Lin. "AN H.264/AVC FULL-MODE INTRA-FRAME ENCODER FOR 1080HD VIDEO", IEEE ICME, Apr. 2008, pp. 1037-1040.

[5] C.M. Diniz, B. Zatt, L. Agostini, A. Susin, and S. Bampi, "A Real Time H.264/AVC INTRA FRAME PREDICTION HARDWARE ARCHITECTURE FOR HDTV 1080P VIDEO", IEEE ICME, Aug. 2009, pp.1138-1141.

[6] M. Shafique, L. Bauer, and J. Henkel, "A parallel Approach for High Performance Hardware Design of Intra Prediction in H.264/AVC Video Codec", IEEE DATE, Apr. 2009, pp.1434-1439.

[7] Y.-C. Kao, H.-C. Kuo, Y.-T. Lin, C.-W. Hou, Y.-H. Li, H.-T. Huang, and Y.-L Lin, "A High-Performance VLSI Architecture for Intra

Prediction and Mode Decision in H.264/AVC Video Encoding", IEEE APCCAS, April 2007, pp.562-565.

[8] C.-W. Ku, C.-C. Cheng, G.-S. Yu, M.-C. T, and T.-S. Chang, "A High-Definition H.264/AVC Intra-Frame Codec IP for Digital Video and Still Camera Applications", IEEE TCSVT, Aug. 2006, pp.917-928.

[9] Y.-K. Lin, C.-W. Ku, D.-W. Li, and T.-S. Chang, "A 140-MHz 94 K Gates HD1080p 30-Frames/s Intra-Only Profile H.264/AVC Encoder", IEEE TCSVT, Mar. 2009, pp.432-436.

[10] S. Smaoui, H. Loukil, A. Ben Atitallah, and N. Masmoudi, "An Efficient Pipeline Execution of H.264/AVC Intra 4x4 Frame Design", IEEE SSD, Jun. 2010, pp.1-5.

[11] GenHua Jin, Jin-Su Jung, and Hyuk-Jae Lee, "An Efficient Pipeline Architecture for H.264/AVC Intra Frame Process" IEEE ISCAS, May 2007, pp.1605-1608.

[12] R. Dornelles, F. Sampaio, D. Palomino, and L. Agostini, "Transforms and Quatization Design Targeting the H.264/AVC Intra Prediction Constraints", 22nd Annual Symposium on Integrated Circuits and System Design, ACM, 2009.