Waseda University Doctoral Dissertation

Research on Architecture Design for Ultra High Definition Video Encoding

Gang HE

Graduate School of Information, Production and Systems

Waseda University

January 2014

Abstract

Compared with high definition (HD, 1920x1080) video, ultra high definition (UHD) could provide much better visual experience. It contains two resolutions: 4K (3840x2160) and 8K (7680x4320, also known as super Hi-vision), which are 4 and 16 times of pixels per frame of HD video. UHD will be used as the future standard in the digital television.

Due to the huge data of UHD video, the storage and transmission will be great challenges and therefore the real-time efficient video compression is necessary. H.264/AVC and H.265/HEVC are the latest video coding standards. Although they could offer efficient compression performance, while the implementations of them takes very high computational complexity, due to many involved complex coding tools, for example, the intra prediction and fractional motion estimation (FME). Moreover, due to data dependency and the algorithm process, it is limited to apply the pipelining and parallel processing techniques in hardware design. The existing works proposed many architecture designs of intra prediction and FME, mainly targeting for 1080p HD or lower resolution videos. We cannot realize the UHD-throughput design by simply increasing times of previous architectures, since it leads to incredibly large hardware cost and high power for a chip.

Therefore, the target of this dissertation is to develop the efficient architectures of intra prediction and FME in H.264 or HEVC, targeting for UHD videos. Firstly, for intra prediction, an MB/Block co-reordering method is proposed. By doing so, we can parallelize the mode decision and reconstruction processes, and hardware utilization is improved by almost 50%. An H.264 VLSI architecture is designed for 4kx2k UHD. Secondly, to achieve higher throughput and reduce more complexity, an interlaced block reordering strategy, together with a preliminary mode decision are proposed. The hardware reusing methods are also used to optimize the architecture. As a result, the hardware complexity (the product of hardware area and required operating frequency) is reduced by 77%. The design achieves not only the

throughput of 8kx4k UHD, but also high hardware utilization. Thirdly, a high-performance and low-power HEVC FME architecture is designed, with the co-optimization in algorithm and hardware architecture. An approximation interpolation, together with a directional search pattern reduces the complexity. The exhaustive-size hadamard transform is adopted to improve coding quality. The design realizes the real-time encoding for 8kx4k UHD and achieves much improvement on power efficiency.

The dissertation consists of 5 chapters as follows:

Chapter 1 [Introduction] gives a brief introduction to ultra-high definition video and video compression system. The basic knowledge of intra prediction and fractional motion estimation are introduced, followed by the contributions and an overview of this dissertation.

Chapter 2 [An H.264 Intra Prediction Architecture for 4kx2k UHD] first introduces the design challenges of intra prediction, and then presents the proposed architecture based on H.264 for 4kx2k UHD.

Due to the high data dependency of intra prediction in H.264, both pipelining and parallel processing techniques are limited to be applied. Moreover, it is difficult to get high hardware utilization and throughput because of the long MB-level and block-level reconstruction loops.

Many researchers has proposed the hardware architectures of intra prediction. Lin (TVLSI'2009) proposed a interlaced schedule for prediction modes of different sizes to improve the pipeline efficiency, but it is only limited to the mode decision component. In order to eliminate the data dependency, Mochizuki (JSSC'2008) a

(PCS'2007) used original pixels instead of reconstructed one to do the prediction, but brings large quality loss. Moreover, the above designs targets at HD or lower resolutions.

In this chapter, an H.264/AVC intra prediction architecture for 4kx2k UHD is presented. The proposed macroblock (MB) and block co-reordering can avoid data dependency and improve pipeline utilization by almost 50%. The timing constraint of real-time 4096x2160 encoding can be achieved with negligible quality loss.

16x16 and 8x8 prediction engines work parallel for generating coefficients. A reordering interlaced reconstruction is also designed for fully pipelined architecture. It takes only 160 cycles to process one MB. Hardware utilization of the prediction and reconstruction modules achieve 90-100%. Furthermore, PE-reusable 8x8 intra predictor and hybrid SAD & SATD mode decision are proposed to save hardware cost. The design is implemented by 90nm CMOS technology with 113.2k gates and can encode 4096x2160 video sequences at 60 fps with operation frequency of 332MHz. Compared with previous designs of 1080p@30fps (Lin,TCVST'2009, Chuang,PCS'2007), throughput of the design increases for 8 times with the overhead of less than 30% hardware cost.

Chapter 3 [An H.264 Intra Prediction Architecture for 8kx4k UHD] discusses the design problems for very high throughput and presents an H.264 8kx4k UHD intra-prediction architecture.

Firstly, due to this huge throughput requirement, design challenges such as complexity and data dependency, which currently exist for lower resolutions (e.g. 2160p and 1080p), become even more critical. Moreover, Pipeline latency influences the efficiency of pipelines with serious data dependency.

In this work, we first propose an interlaced block reordering scheme together with a preliminary mode decision (PMD) strategy to resolve the data dependency between intra mode decision and reconstruction. In the meantime, hardware cost is reduced by PMD. We also propose a probability-based reconstruction scheme to solve the problem of long pipeline latency. In addition, hardware reuse strategies including a shared fine decision module and processing element-reusable prediction generator, are applied to further optimize the design. As a result, the hardware complexity (the product of hardware area and required operating frequency) is reduced by 77%, and it takes an average of 33 cycles to process an MB. The implementation result demonstrates that our design can support up to the specification of 7680 × 4320p 60 f/s when running at 273 MHz. The 1080p 30 f/s encoding requires less than 9 MHz operating frequency, which is much lower than those (114MHz, 140MHz) used in previous works (Kuo:TVLSI'2011,

Lin:TCSVT'2009).

Chapter 4 [An HEVC Fractional Motion Estimation Architecture for 8kx4k UHD] presents a high-performance low-power HEVC FME architecture design.

FME improves the rate-distortion performance significantly about 3-6dB, but results in high computation complexity (40% encoding time) due to the complex interpolation and fractional search process. Previous works have proposed efficient designs in H.264 (T I'2010). However, since HEVC involves many FME-related new features, such as 8-tap interpolation and quad-tree coding structure, they cannot be applied directly.

In this work, the design is co-optimized in algorithm and hardware architecture to reduce the complexity and achieve high throughput. Bilinear quarter pixel approximation, together with a directional 5T12S (5 transform points among 12 search points) search pattern is proposed to reduce the complexity of the interpolation and search process. Furthermore, an exhaustive size-hadamard transform (ES-HAD) is introduced to improve coding quality, and determine the best transform size rather than using complex transform coding. Besides, a data reuse method of ES-HAD is applied to reduce the hardware overhead. This design is implemented in 65nm CMOS chip and verified by FPGA based evaluation system. It achieves 995Mpixels/s for 7680x4320 30fps encoding, at least 4.7 times faster than previous designs. Its power dissipation is 198.6mW at 188MHz, with 0.2nJ/pixel power efficiency. Despite high complexity in HEVC, the chip achieves 56% and 90% improvement on power efficiency than previous works in H.264 (Tsung,ICME'2009, Kao,TVLSI'2010).

Chapter 5 [Conclusion] concludes the proposed works and present the contribution of this dissertation.

Acknowledgement

First of all, I would like to thanks my supervisor, Professor Satoshi Goto, for his guidance, encouragement and supported me in the whole process of my research. His patient and understanding personality makes me enjoy the time to work with him. He also taught me a lot in my life.

I would also like to thank Professor Shinji Kimura, Professor Takeshi Yoshimura and Professor Jiro Katto of Waseda University, for their guidance and suggestions through my research. Thanks a lot for the advice and comments for improving my dissertation and all the guidance through my study.

I would like to thank Assistant Professor Dajiang Zhou of Waseda University. He gives me accurate suggestions and precious comments to help me improving my research.

Thanks to the excellent research collaborators I have, Mr. Tianruo Zhang, Ms. Jinjia Zhou, and Mr. Zhixiang Chen. Discussing with them gave me great inspiration in my research. I also thanks to all the members in Goto lab. They gave me a lot help and advices in my research and life in the past three years. Also thanks to my friends in Hibikino who enriched my study life in the past ten years in Japan.

Finally, I would extend my deep thanks to my parents for their understanding and supporting.

Contents

Index of Figures

Index of Tables

1. Introduction

1.1 Background

Compared with high definition (HD, 1920x1080) video, ultra high definition (UHD) could provide much better visual experience. It contains two resolutions: 4K (3840x2160) and 8K (7680x4320, also known as super Hi-vision), which are 4 and 16 times of pixels per frame of HD video. UHD will be used as the future standard in the digital television.

Due to the huge data of UHD video, the storage and transmission will be great challenges and therefore the real-time efficient video compression is necessary. H.264/AVC and H.265/HEVC are the latest video coding standards. Although they could offer efficient compression performance, while the implementations of them takes very high computational complexity, due to many involved complex coding tools, for example, the intra prediction and fractional motion estimation (FME). Moreover, due to data dependency and the algorithm process, it is limited to apply the pipelining and parallel processing techniques in hardware design. The existing works proposed many architecture designs of intra prediction and FME, mainly targeting for 1080p HD or lower resolution videos. We cannot realize the UHD-throughput design by simply increasing times of previous architectures, since it leads to incredibly large hardware cost and high power for a chip.

1.1.1 Introduction to Intra Prediction

Intra prediction, which uses neighboring pixel values to predict the currently coding block, explores spatial redundancy of the video. Fig. 1 shows the H.264/AVC intra-frame encoding flow. Firstly, a prediction generator (PG) unit refers to reconstructed pixels of neighboring block to generate predictive pixels for each mode. The residues are then generated and transformed into coefficients. Based on a given

cost function, the costs are calculated to perform mode decision. The quantization unit processes the coefficients of the best mode and outputs the results to both the inverse quantization unit and the entropy coding unit. The inverse quantization and inverse transform units translate quantized values back to residuals, which are used to reconstruct pixels for PG operation of the next block. The entropy coding unit encodes quantized values and mode information for bit-stream output.

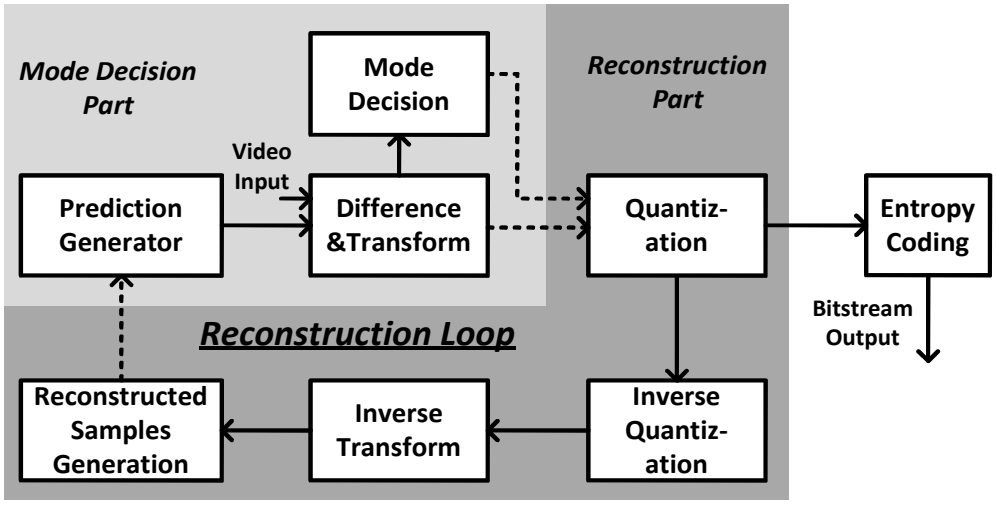
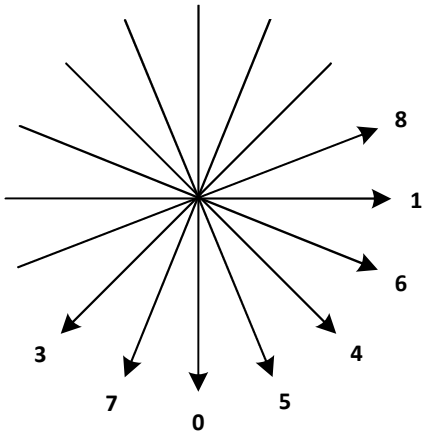


Fig.1. H.264/AVC intra-frame encoding flow



MODE	DIRECTION	4x4/ 8x8	16x16
0	Vertical	y	y
1	Horizontal	y	y
2(DC)	N/A	y	y
3	Diagonal down-left	y	
4	Diagonal down-right	y	
5	Vertical-right	y	
6	Horizontal-down	y	
7	Vertical-left	y	
8	Horizontal-up	y	
Plane	N/A		y

Fig.2. The H.264 intra prediction modes

a) Prediction Mode

The prediction modes for luma component are categorized into three sizes. As shown in Fig. 2, both 4x4 and 8x8 predictions contain eight directional and a DC

modes, while 16x16 prediction has two directional (vertical and horizontal), a DC and a plane modes. The 16x16 pixels in an MB can be encoded as 16 predicted blocks with the 4x4 mode, or four predicted blocks with the 8x8 mode, or just a predicted block with the 16x16 mode. In addition, four 8x8 chroma modes, which are similar to 16x16 luma modes, are adopted to predict the two Cb and Cr 8x8 blocks within an MB.

b) Cost Function

The optimal cost function for coding efficiency is the rate distortion optimization (RDO), which brings high computational complexity. In the hardware design that targets at real-time coding for high resolution videos, RDO leads to large hardware cost and power dissipation. For lower complexity, the sum of absolute transform difference (SATD) is commonly used in conventional designs. The cost of each mode is estimated with transformed coefficients. The computational formula is defined as follows:

$$Cost = SATD + \lambda(QP) \bullet R \quad (1)$$

$$SATD = \sum\sum |T(Cur - Pre)| \quad (2)$$

In (1), R is set to 0 for the most probable mode and to 4 for the other modes. The value of λ is a function of the quantization parameter (QP). In (2), T(x) can be either a Hadamard transform (HT) or an integer discrete cosine transform (DCT). Cur and Pre denote the original and predictive pixels. In addition, the sum of absolute difference (SAD) cost function is also suitable for hardware design. It does not need to perform the transform operation, thus introduces the least complexity.

1.1.2 Introduction to Fractional Motion Estimation (FME)

As an important component of video encoding, fractional motion estimation (FME) provides the sub-pixel refinement. It improves the rate-distortion performance significantly about 3-6dB, but results in high computation complexity

(40% encoding time) due to complex interpolation and fractional search process.

In the common reference software of video coding JM [1] and HM [2], FME is carried out as follows: the fractional pixels (pels) with quarter accuracy are first interpolated with specific filter. Then, two refinements from half pixel to quarter pixel are performed sequentially, as shown in Fig. 3. In each refinement, nine neighboring points around the former best result are searched. The cost function adopted in FME is the hadamard transform absolute difference (HAD). It includes the operations of difference generation (DG), 4x4/8x8 hadamard transform (HT), absolute value and cost accumulation.

```
Loop (CU)
  Loop (PU mode)
  {
    integer motion estimation();
    fraction motion estimation
    {
      sub-pixel interpolation;
      half-pixel refinement;
      quarter-pixel refinement;
    };
  };
  Residual quad-tree coding;
};

sub-pixel refinement
{
  Loop(9 neighboring points)
  {
  difference generation;
  hadamard transform;
  absolute value;
  cost accumulation;
  }
};
```

Fig.3. FME process in reference software (HM)

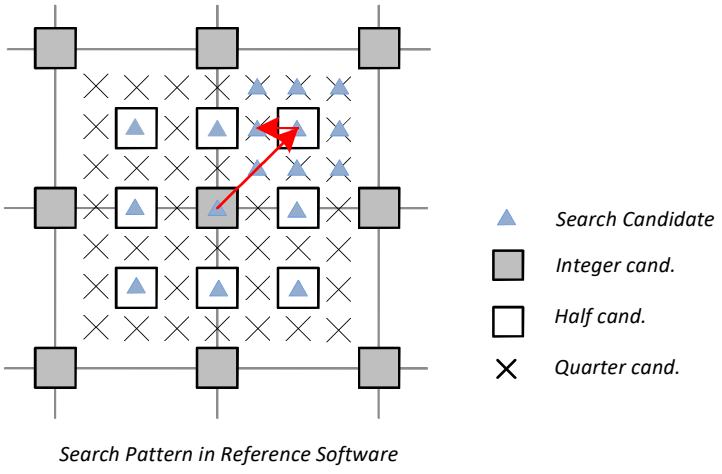


Fig.4. FME search pattern in reference software

1.2 Contributions of the Dissertation

1.2.1 Architecture Design for an H.264/AVC 4kx2k UHD Intra Prediction

In this dissertation, we first analyzed the design challenges of intra prediction, and proposed an efficient architecture based on H.264 for 4kx2k UHD. Due to the high data dependency of intra prediction, both pipelining and parallel processing techniques are limited to be applied. Moreover, it is difficult to get high hardware utilization and throughput because of the long MB-level and block-level reconstruction loops.

The MB and block co-reordering is proposed to solve data dependency problem and improve pipeline utilization by almost 50%. The timing constraint of real-time 4096x2160 encoding can be achieved with negligible quality loss. 16x16 and 8x8 prediction engines work parallel for generating coefficients. A reordering interlaced reconstruction is also designed for fully pipelined architecture. Furthermore, PE-reusable 8x8 intra predictor and hybrid SAD & SATD mode decision are proposed to save hardware cost. It takes only 160 cycles to process one MB. Hardware utilization of the prediction and reconstruction modules achieve 90-100%. The design is implemented by 90nm CMOS technology with 113.2k gates and can encode 4096x2160 video sequences at 60 fps with operation frequency of 332MHz.

Compared with previous designs of 1080p@30fps (Lin,TCVST'2009, Chuang,PCS'2007), the throughput of proposed design increases for 8 times with less 30% area overhead. For HD1080p 30 fps encoding requirement, this design can reduce 72% of operating frequency compared with Lin's work (TCVST'2009) because of high parallelism and pipeline efficiency.

1.2.2 Architecture Design for an H.264/AVC 8kx4k UHD Intra Prediction

In this dissertation, an H.264 8kx4k UHD intra-prediction architecture is also presented. Due to this huge throughput requirement, design challenges such as complexity and data dependency, which currently exist for lower resolutions (e.g. 2160p and 1080p), become even more critical. Moreover, Pipeline latency influences the efficiency of pipelines with serious data dependency.

In this work, we first propose an interlaced block reordering scheme together with a preliminary mode decision (PMD) strategy to resolve the data dependency between intra mode decision and reconstruction. In the meantime, hardware cost is reduced by PMD. We also propose a probability-based reconstruction scheme to solve the problem of long pipeline latency. In addition, hardware reuse strategies including a shared fine decision module and processing element-reusable prediction generator, are applied to further optimize the design. As a result, the hardware complexity (the product of hardware area and required operating frequency) is reduced by 77%, and it takes an average of 33 cycles to process an MB. The implementation result demonstrates that our design can support up to the specification of 7680 × 4320p 60 f/s when running at 273 MHz. The 1080p 30 f/s encoding requires less than 9 MHz operating frequency, which is much lower than that used in previous works(Kuo:TVLSI'2011, Lin:TCSVT'2009).

1.2.3 Architecture Design for An HEVC 8kx4k UHD Fractional Motion Estimation

In this dissertation, a 995Mpixels/s 0.2nJ/pixel fractional motion estimation architecture in HEVC for 8kx4k UHD is also presented.

In this work, the design is co-optimized in algorithm and hardware architecture

to reduce the complexity and achieve high throughput. They are characterized as follows. (1) By using bilinear quarter pixel approximation, we reduce 76% interpolation complexity and save transform operation for quarter candidates. (2) A 5T12S search pattern is proposed to achieve a tradeoff between hardware cost and coding quality. 48% hardware cost is reduced with negligible quality loss, compared with conventional 9T25S. (3) Exhaustive size-hadamard transform (ES-HAD) is adopted in FME. It avoids unifying all blocks into small transform ones. Furthermore, it determines the best transform size, rather than using the complex RQT. Besides, data reusing in ES-HAD is exploited and 58% hardware cost is reduced, compared with the straightforward implementation.

This design is implemented in 65nm CMOS chip and verified by FPGA based evaluation system. It achieves 995Mpixels/s for 7680x4320 30fps encoding, at least 4.7 times faster than previous designs. Its power dissipation is 198.6mW at 188MHz, with 0.2nJ/pixel power efficiency. Despite high complexity in HEVC, the chip achieves 56% improvement on power efficiency than previous works in H.264 (Tsung,ISSCC'2009, Kao,TVLSI'2010).

1.3 Organization of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 and 3 presents the architecture design of intra prediction in H.264. Chapter 2 proposed an MB/block co-reordering strategy and target for 4kx2k UHD applications. Chapter 3 adopted a block interlaced reordering method and designed an 8kx4k UHD intra prediction architecture. Chapter 4 introduces the FME process in HEVC and presented a high-performance low-power FME architecture for 8kx4k UHD. Finally, chapter 5 concludes the whole dissertation.

2. An H.264 Intra Prediction Architecture for 4kx2k UHD

2.1 Introduction

2.1.1 Background

Due to the excellent video coding efficiency, the H.264/AVC coding standard which is developed jointly by ITU-T and ISO/IEC [1] is now widely adopted in industry. A lot of coding techniques are involved to improve the coding efficiency, such as intra prediction, multiple reference frame, variable block size, and context adaptive binary entropy coding. For the intra frame coding, H.264/AVC outperforms the famous picture coding standard JPEG 2000.

In order to improve coding efficiency at high-end application, H.264 Fidelity Range Extensions (FREext) project brings up a suite of some new profiles collectively called High profiles. The high profile supports all features of the prior Main profile, and introduces two main coding tools, 8x8 integer transform and 8x8 intra prediction additionally. Similar to intra 4x4, intra 8x8 has eight different direction prediction modes and one DC prediction mode. The luma values of each sample in 8x8 blocks are predicted from neighboring reconstructed reference pixels based on prediction modes. Meanwhile, the boundary reference pixels are pre-filtered before the next prediction. Furthermore, the 8x8 integer Discrete Cosine Transform (DCT) is adopted for the 8x8 intra prediction. The coding efficiency of the intra frame coding is further improved with these coding tools [4][22].

2.1.2 Intra Prediction Encoding Analysis

Figure 1 illustrates the processing flow of the H.264/AVC intra frame encoding. Firstly, referring the neighboring reconstructed pixels, the intra prediction unit generates the predicted pixels of the current block for all modes. The best mode is chosen by the mode decision unit. Then residuals are generated and transformed into coefficients. After that, the quantization unit processes coefficients and sends them to both inverse quantization unit and entropy coding unit. The inverse quantization unit and the inverse transform unit translate quantized values back to residuals. The reconstructed pixels are generated by the adding operation for the next intra prediction. Besides, the entropy coding unit encodes the quantized values and mode information for final bit-stream output.

High throughput and hardware utilization are always challenged by the reconstruction loop shown in Fig. 5. Fig. 6 illustrates the standard zig-zag scanning order for 8x8 block in the H.264/AVC intra-frame encoding process. In the intra encoding, the neighboring left, upper and upper-right reconstructed pixels are used for one block prediction. Serious data dependency decreases prediction and reconstruction pipeline efficiency. For example, block 1 prediction process does not start until block 0 reconstruction is finished. Block 2 and block 3 prediction process need the reconstruction pixels of their former block respectively. Therefore, prediction pipeline has to be idle during these reconstruction cycles. Meanwhile, reconstruction part is also idle on prediction working time. Besides, block 0 prediction process which does not need reconstruction pixels of former block 3 will wait longer time if the former MB chooses 16x16 as best mode. In that case, 16x16 mode reconstruction of the former MB has to be done before the next block/MB prediction operation. So it is difficult for the pipeline to achieve high utilization due to data dependency. Too many cycles are used to process one MB in this way and high throughput like Quad Full High Definition (QFHD, considered as 4096x2160 resolution in this paper) cannot be realized. For high throughput application, since the design takes more hardware cost and the efficiency should be paid more attention, this problem becomes more critical.

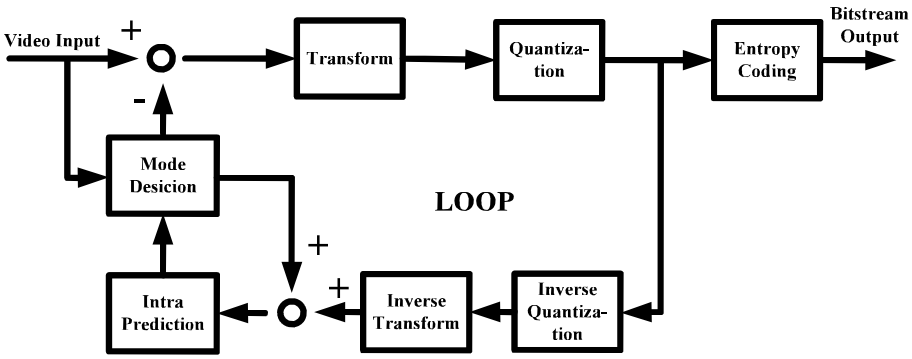


Fig.5. H.264/AVC intra-frame encoding flow.

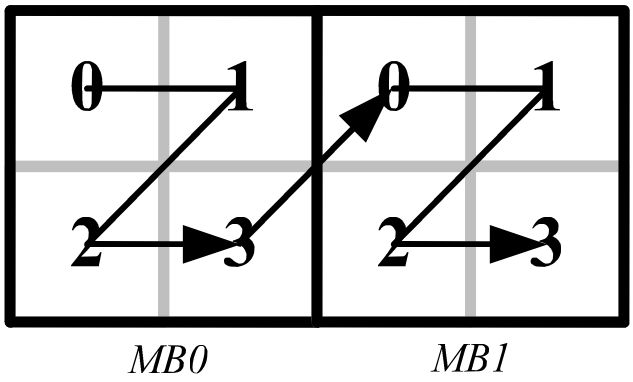


Fig.6. Standard zig-zag scanning order for 8x8 block

Several researches have been published on H.264 intra hardware architecture. Basically, there are two methods to deal with the data dependency problem in previous works. A common strategy is interlacing different size mode prediction, which usually mixes 4x4 prediction, 16x16 prediction and 8x8 prediction to reduce some pipeline bubbles. Reference [3][9][11] inserted luma 16x16 into the idle waiting cycles of luma 4x4 intra prediction. It could improve the hardware utilization to some extent, but it is still not efficient enough due to the following two reasons. Firstly, hardware is designed to be configurable while the hardware efficiency and utilization are reduced. Secondly, it still introduces bubbles in the pipeline. The other method is processing mode prediction with original boundary pixels instead of reconstruction pixels [2][5][8]. At the cost of some quality loss, it could alleviate the data dependency of prediction part. Table 1 shows

performance comparison between original pixels and reconstructed pixels predicting. The degradation of coding efficiency is about 5% bit rate increasing when encoding videos with bigger quantization parameters (QP), such as 32 and 36. This is because in that case original pixels are far from reconstructed pixels due to the quantization operation. In addition, original pixels predicting causes 1.43% BD-bit rate [10] increasing averagely. Due to the low pipeline efficiency and parallelism, previous works are focused on High Definition (HD) resolution such as 720p, 1080p. Only [2] could work for 4096x2160@24fps, while it used original pixels for intra mode decision with quality loss. In this paper, we focus on developing high-throughput intra encoder with high hardware utilization and good quality.

Table 1 Performance comparison between original pixels predicting & JM17.0 (supporting 8x8 and 16x16 mode)

Seq. / QP	InToTree(4096x2160)		RushHour(1080p)	
	△Bit rate (%)	△PSNR (dB)	△Bit rate (%)	△PSNR (dB)
36	6.93	-0.008	5.59	-0.114
32	3.89	0	4.26	-0.079
28	2.07	-0.005	3.84	-0.011
24	0.84	-0.002	2.84	-0.006
20	0.29	-0.002	1.24	-0.001
16	0.12	-0.002	0.64	0.001
12	0.07	0.001	0.35	-0.001

2.1.3 Overview and organization

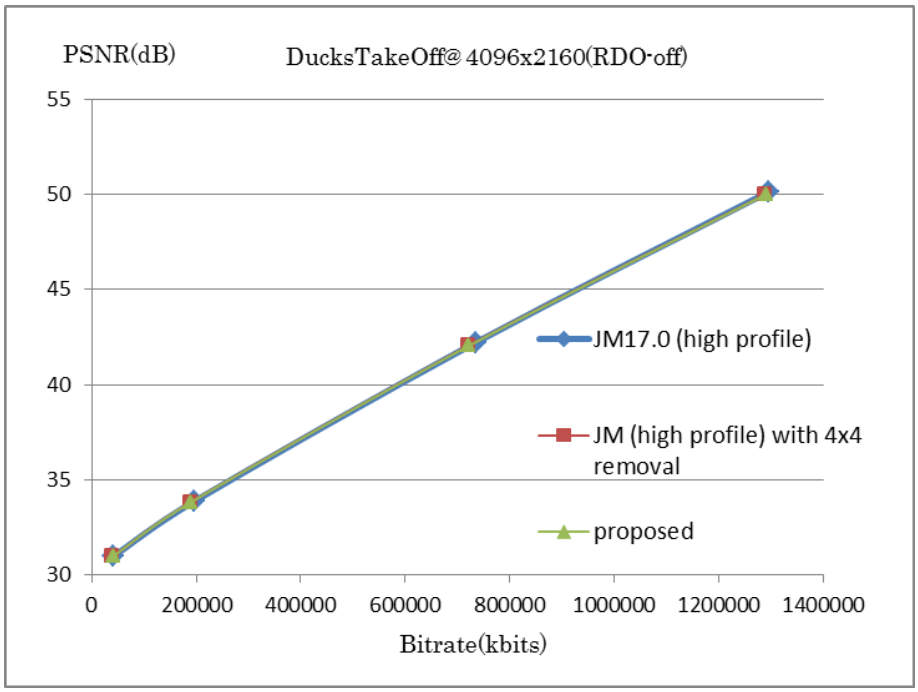
The proposed architecture is designed for 4096x2160@60fps real-time application with reconstruction pixels predicting. Near full-mode intra 8x8 mode and 16x16 mode are chosen and good performance is achieved. In summary, this paper proposes the following techniques: (1) MB/block co-reordering to solve data dependency problem, (2) two prediction engines processing parallel and reordering interlaced reconstruction to enhance the throughput with high hardware utilization, and (3) PE-reusable 8x8 intra predictor and hybrid SAD & SATD mode decision to save hardware cost.

The rest of this chapter is organized as follows. In Section 2, the details and design features of the proposed intra prediction architecture are described. Then the implementation results and design comparison are shown in Section 3. The final section summarizes the chapter and presents our conclusion.

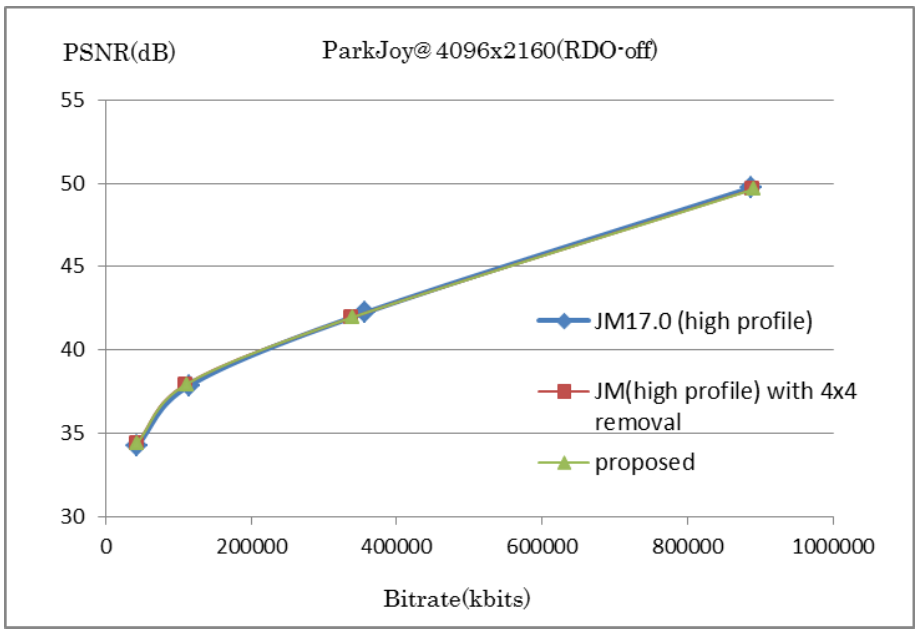
2.2 Proposed Architecture

Our design target is very high throughput application such as 4096x2160 and 1080p. Experiment result shows that in this kind of video sequence, if 4x4 mode is removed we can get almost the same performance with reserving 4x4 mode prediction. In the experiment, intra prediction is processed with sum of absolute transformed difference (SATD) mode decision which is more suitable for hardware implementation. Table 2 illustrates the comparison results with BD-bit rate and Figure 3 shows the rate-distortion (RD) curves of the two sequences. By comparing these results, we can observe that 4x4 mode usually introduces large bit rate with unobvious PSNR increasing. 8x8 mode prediction in high profile may brings a little more residual in the bit rate scream, but generates much less head information than 4x4 for smooth MBs. Besides that, SATD mode decision cannot choose best mode accurately like rate-distortion optimization (RDO) and influences the comparison results. Without 4x4 prediction, quality loss could be caused for small resolution. While, the main target of this design is the high throughput application, 4x4 mode is removed in this

architecture.



(a)



(b)

Fig.7. R InToTree" and (b) "ParkJoy" in QP range from 12 to 36.

Table 2 Performance Comparison with JM17.0 [13] (High Profile supporting full 4x4, 8x8 and 16x16 modes)

Sequence	QP	IP 8x8&16x16		Proposed.	
		△PSNR (db)	△Bit rate(%)	△PSNR (db)	△Bit rate(%)
DucksTakeOff	12	-0.118	-0.410	-0.120	-0.338
	20	-0.157	-1.765	-0.167	-1.733
	28	-0.055	-3.683	-0.058	-3.556
	36	0.008	-0.510	0.013	0.239
BD-Bit rate		-0.98%		-0.85%	
ParkJoy	12	-0.099	0.208	-0.097	0.334
	20	-0.315	-5.567	-0.292	-5.110
	28	0.087	-4.141	0.087	-3.646
	36	0.116	-2.285	0.118	-1.836
BD-Bit rate		0.30%		0.31%	

In H.264/AVC reference software, Hadamard Transform (HT) is used to calculate SATD for mode decision since it has less computation complexity and no scaling effect. However, in the real encoding process, the DCT is adopted for reconstruction loop. In order to estimate bit rate more accurately, 4x4 and 8x8 integer DCT in cost function is used to approximate the effect of transform and quantization in H.264 encoding process. In this architecture which is designed for high throughput application, 8x8 mode and

16x16 mode with DCT-based SATD are implemented to simplify the architecture with good performance.

Table 3 The increase in BD-bit rate(%) by comparison with JM17.0 (supporting 8x8 and 16x16 mode) in QP range from 12 to 36

Sequences	Discarding mode 3 & 7 for 8x8 down-left block
DucksTakeOff(4096x2160)	0.03
ParkJoy(4096x2160)	0.32
InToTree(4096x2160)	0.14
CrowdRun(4096x2160)	0.34
Pedestrian(1080p)	0.42
BlueSky(1080p)	0.36
Station(1080p)	0.62
RushHour(1080p)	0.31
Tractor(1080p)	0.37
SunFlower(1080p)	0.63
Average	0.35

2.2.1 MB/Block Co-reordering

Intra encoding process can be divided into prediction and reconstruction phases. Prediction phase includes the value predicting, mode cost calculation and mode decision. There are two prediction engines (8x8 and 16x16 engines) to do it, as shown in Fig. 12. Reconstruction operation is composed of quantization, inverse-quantization, inverse-transform and addition.

In this design for QFHD application, 8x8 prediction part takes 40 cycles to process one block and 30 cycles are needed for one 8x8 block reconstruction. Fig. 9 (a) shows the timing schedule by standard zig-zag direct implementation. In this original design,

370 cycles are required for one MB encoding. The idle time takes about one half of total cycles and the hardware efficiency is very low. To alleviate this serious data dependency and improve hardware efficiency, a novel hardware-orient scanning order is proposed

As illustrated in [1], for 8x8 intra prediction, mode 3 and mode 7 need upper-right reconstruction pixels. And this causes data dependency between upper-right block and down-left block of one MB. For removing the idle cycles in pipeline caused by this data dependency, mode 3 & mode 7 of 8x8 block intra prediction are discarded only for down-left block in each MB. By breaking this data dependency, one block reconstruction idle time can be avoided. Moreover, it also could be used for compacting MB/block co-reordering pipeline (explained in the fifth paragraph of section 2.1). The probability of that mode 3 & 7 are chosen as best mode is quite low and the removal of these two modes is only conducted for the down-left 8x8 block, so the quality loss caused by the removing is quite small. Table 3 shows the performance compared with original one, it causes about 0.35% BD-bit rate increasing averagely. Moreover, the degradation does not change a lot according to QP. For example, it changes from 0.03% to 0.29% in QP range from 12 to 36 for InToTree(4096x2160). Therefore, considering the improvement for pipeline architecture and the degradation of coding efficiency, mode 3 & mode 7 are discarded for each down-left block in one MB with small quality loss. Fig. 9(b) shows the changed schedule. There is no idle time between block 1and block 2 for 8x8 intra prediction.

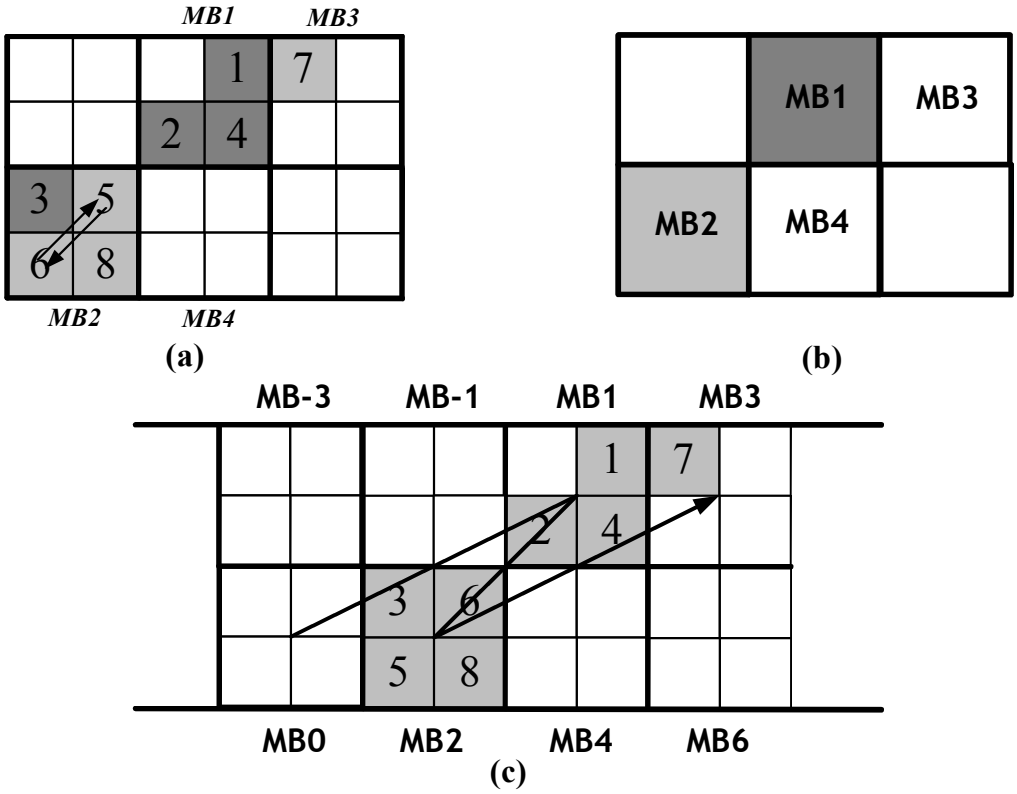


Fig.8. Intra prediction scanning order (a) block level (b) MB level (c) MB/block co-reordering.

However, the main data dependency is not eliminated yet and there are still lots of idle cycles. Due to the strong left and upper prediction reference relationship, data dependency problem cannot be solved with discarding modes as above. However, by processing some other interlaced blocks in these idle cycles, we can improve the pipeline efficiency a lot and reduce the cycles to process one MB. Thus, a scan reordering is proposed for this purpose. The interlaced blocks are from the other neighboring upper and down MBs. 8x8 block intra prediction is processed following the block number in Fig. 7 (a). When block 2 prediction in MB1 is finished, block 3 which actually belongs to MB2 in the down row is under predicting operation. The next one is block 4 and MB1 operation is finished. The same order is processed to finish MB2. The consequence is that no idle time is needed to wait the boundary reconstruction pixels and no quality loss is introduced by this interlacing and

reordering blocks. This interlacing block method is illustrated in Fig. 8. As modified order, block 0 of MB M+1 is processed after block 2 of MB M. Before block 3 prediction of MB M is started, block 2 reconstruction of MB M is already done. Thus, when mode prediction is performed for one block, reconstruction is performed for its former block. Reconstruction waiting idle is fully avoided in 8x8 intra prediction encoding. This makes the hardware utilization reach almost 100%.

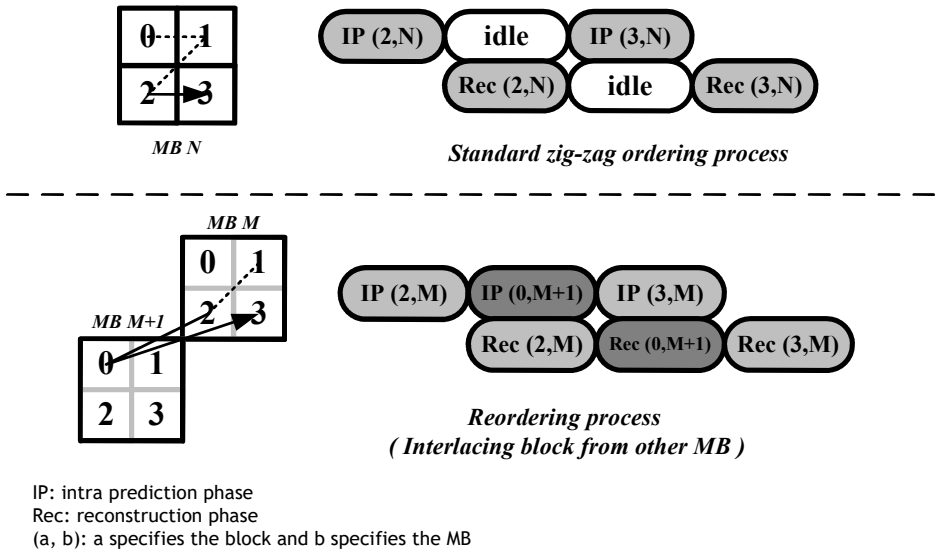


Fig.9. Block level pipeline by interlacing other block.

The new reordering changes the scan order not only in block-level, but also in MB-level for avoiding reconstruction waiting time. The 16x16 mode prediction processes following the MB number in Fig. 8 (b). The 16x16 MB prediction takes the same cycles with four 8x8 blocks prediction. 8x8 engine and 16x16 engine finish MB1 at the same time and choose the final best mode from best 8x8 mode and best 16x16 mode to do the reconstruction. It causes pipeline bubble problem that the MB-level reordering and the block-level reordering are combined directly. For example, in Fig. 8 (a) if MB1 chooses 16x16 prediction mode as best mode, the MB1 reconstruction is started at the time when the 8x8 upper-right block prediction of MB2 (block 5 in Fig. 8 (a)) is supposed to be done. We can find out that block 5 prediction of MB2 needs the reconstruction pixels of block 2 in MB1. So 8x8 prediction engine has to be idle and

waits for the down-left 16x16 reconstruction pixels in MB1. To solve this problem, both 8x8 prediction order and MB internal order of 16x16 mode reconstruction are adjusted. Firstly, another 8x8 block prediction is inserted before upper-right block prediction. Because upper-right block (block 5 in Fig. 8 (a)) and down-left block (block 6 in Fig. 8 (a)) have no data dependency, their processing sequence can be exchanged, as arrows shown in Fig. 8 (a). In this way, one 8x8 block prediction time is obtained before upper-right block prediction of MB2. The MB1 internal order of 16x16 mode reconstruction can be adjusted. In Fig. 8 (a), the down-left 8x8 segment of MB1 can be firstly reconstructed when predictingdown-left block of MB2 (block 6). By this way, the prediction for upper-right block of MB2 (block 5) can be done without waiting and all idle waiting time is eliminated. Fig. 8 (c) shows our final MB/block co-reordering.

Furthermore, to set up a fully pipelined architecture, an interlaced reordering reconstruction is also implemented. Reconstruction for each 8x8 block processes following the same order with 8x8 block prediction. 16x16 mode reconstruction of each MB can be divided into four 8x8-size segments. Every segment of 16x16 mode reconstruction is interlaced between two 8x8 block reconstruction when the best mode is 16x16 prediction mode. Besides, four segments reconstruction should be done in the specified order. In Fig 4(c), for reconstructing 16x16 mode of MB1, down-left segment must be reconstructed at the same time with block 5 prediction, prior to other segments of MB1. The next one is upper-right segment which contains the reference pixels of block 7. Then the down-right and upper-left segments are interlaced sequentially. Thus both 8x8 mode & 16x16 mode prediction can be processed without reference reconstruction waiting even if 16x16 mode is chosen as best mode. Figure 6 shows the cycle reduction by adopted techniques and 16x16 mode is supposed as best mode in the figure. Operation time of four consecutive 8x8 blocks is compared, which is also the processing time of one MB. As illustrated in (d), 8x8 prediction for 3 blocks of current MB and one block of next MB are processing sequentially. Operation time of these 4 blocks which belong to two MBs is 160 cycles, which is considered as the processing time of one MB. Interlaced reconstruction including 8x8 mode and 16x16 mode is also shown in it. For example, “blk 2 M+1 & seg. 1 M” includes 8x8 mode reconstruction for block 2 of MB M+1 and 16x16 mode reconstruction for segment 1 of MB M.

Figure 7 shows timing schedule of the design which only takes 160 cycles to process one MB. Referring to Fig. 8 (c), this figure illustrates interlaced reordering reconstruction at the time when MB2 is under 16x16 mode predicting and block 5, 6, 7 and 8 are under 8x8 mode predicting. In prediction phase, 8x8 mode prediction engine adopts 16 pixels per cycle to achieve high throughput for 4096x2160 application. It takes 36 cycles to finish 9 modes prediction and 4 cycles to recompute the best mode for reconstruction. As shown in "Intra prediction 8x8 mode" line of Fig. 10, it takes 40 cycles to process one block. The working flow of 16x16 mode prediction engine is explained in part 2.3. The 160 cycles for each MB reconstruction can be divided into 4 periods equably. In each period (40 cycles), former 16 cycles are taken by 8x8 block reconstruction. There is only one exception that reconstruction for down-right 8x8 block in each MB is unnecessary if the MB chooses 16x16 mode as the final best mode. In that case, the former 16 cycles are used to process direct current (DC) coefficients of both chroma and 16x16 mode luma, which are generated by DCT and processed by HT in intra prediction phase. It is illustrated as the first dark rectangle of "Reconstruction choose 16x16" line in Fig. 10. "Seg." in that line means one 8x8-size segment of MB reconstruction for 16x16 mode and its index specifies the location of segment referring to Fig. 8 (c). 16 cycles in one period are taken for one segment. Segments are interlaced with blocks and processed as specified order. To process chroma component, 8 cycles in one period are taken for one half of Cr or Cb, as shown in Fig. 10. Thus, this reordering interlaced reconstruction schedule, which guarantees no waiting time for intra prediction phase, improves hardware utilization in the design.

In intra prediction encoding, quantization and inverse-quantization which contain multiplications always take a lot of area in the whole design. Most of previous researches [3][4][5][12] use 8 or 4 samples per cycle in Q and IQ for 1080p application. Actually, due to pipeline bubbles caused by data dependency, the usage efficiency of Q and IQ is very low. In our design, we remove the pipeline bubbles and make Q and IQ utilization nearly 100%. Thus only 4 samples per cycle are adopted in Q and IQ for 2160p application, which saves a lot of hardware cost.

Considering the proposed reordering is performed on not only the block level but also the MB level, to apply this method, the processing sequence of some other

components of a complete video encoder, such as the motion estimation (ME), should also be reordered. While the processing order of the entropy coding (CABAC) part cannot be modified due to the dependency of the context table, a DRAM buffer system similar to that in [7] can be utilized to reorder the syntax elements prior to CABAC. As shown in Fig. 12, in this work, the buffer is designed to be located between the quantization and entropy coding. The DRAM bandwidth overhead incurred by this buffer can be negligible, compared to the huge DRAM bandwidth consumption of the other parts of the encoder including ME and the line buffer for the deblocking filter.

It should be noted that the MB-level reordering can also be an efficient optimization method for other parts of the encoding/decoding system. It has been reported in [7] that the reordering saves DRAM bandwidth for MC and deblocking by 20% and 75% respectively, while even more reduction can be expected for ME [6] in an encoder. Therefore, it is reasonable to apply the proposed MB/block co-reordering as a system-level optimization technique, instead of only considering the intra part.

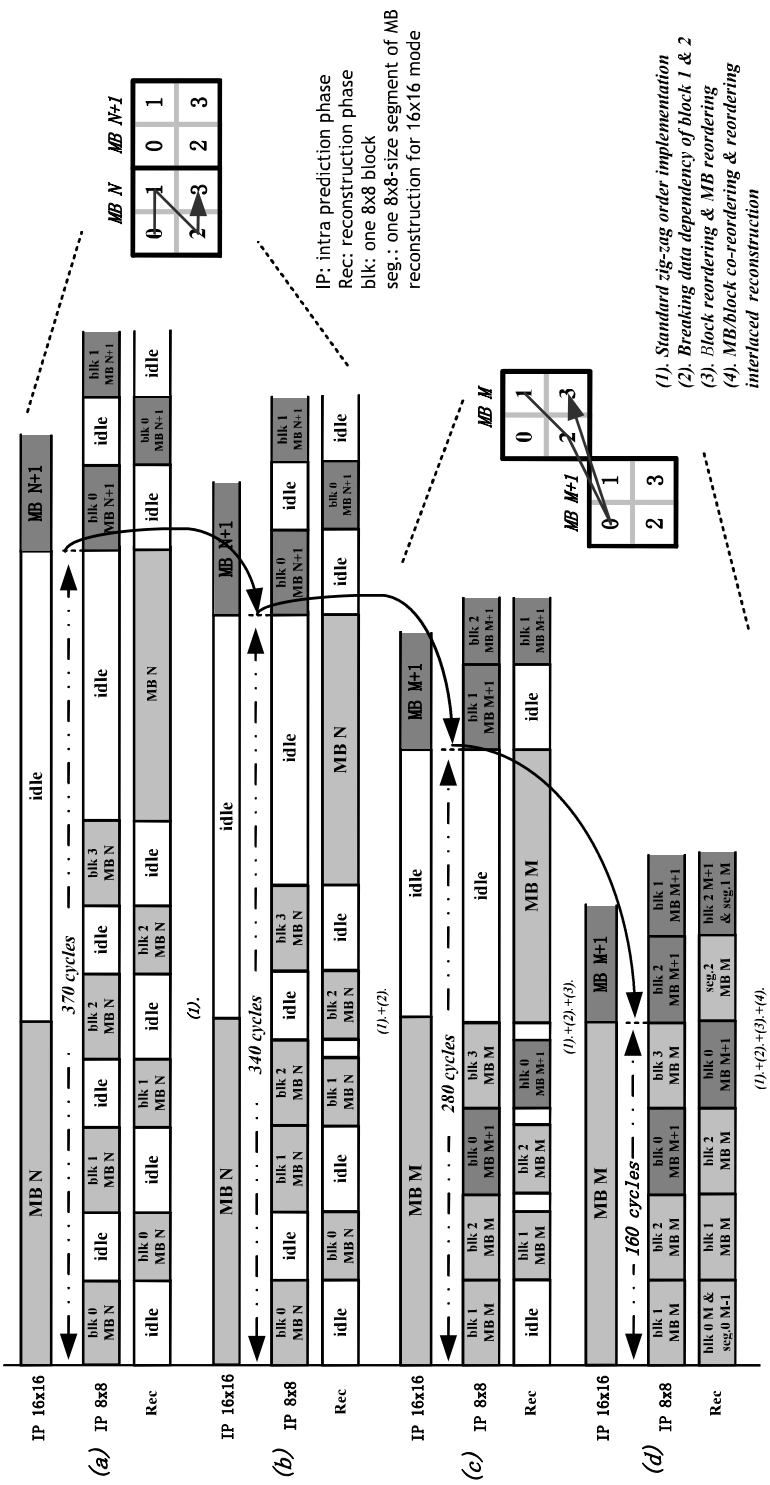


Fig.10. Cycle reduction by adopted techniques

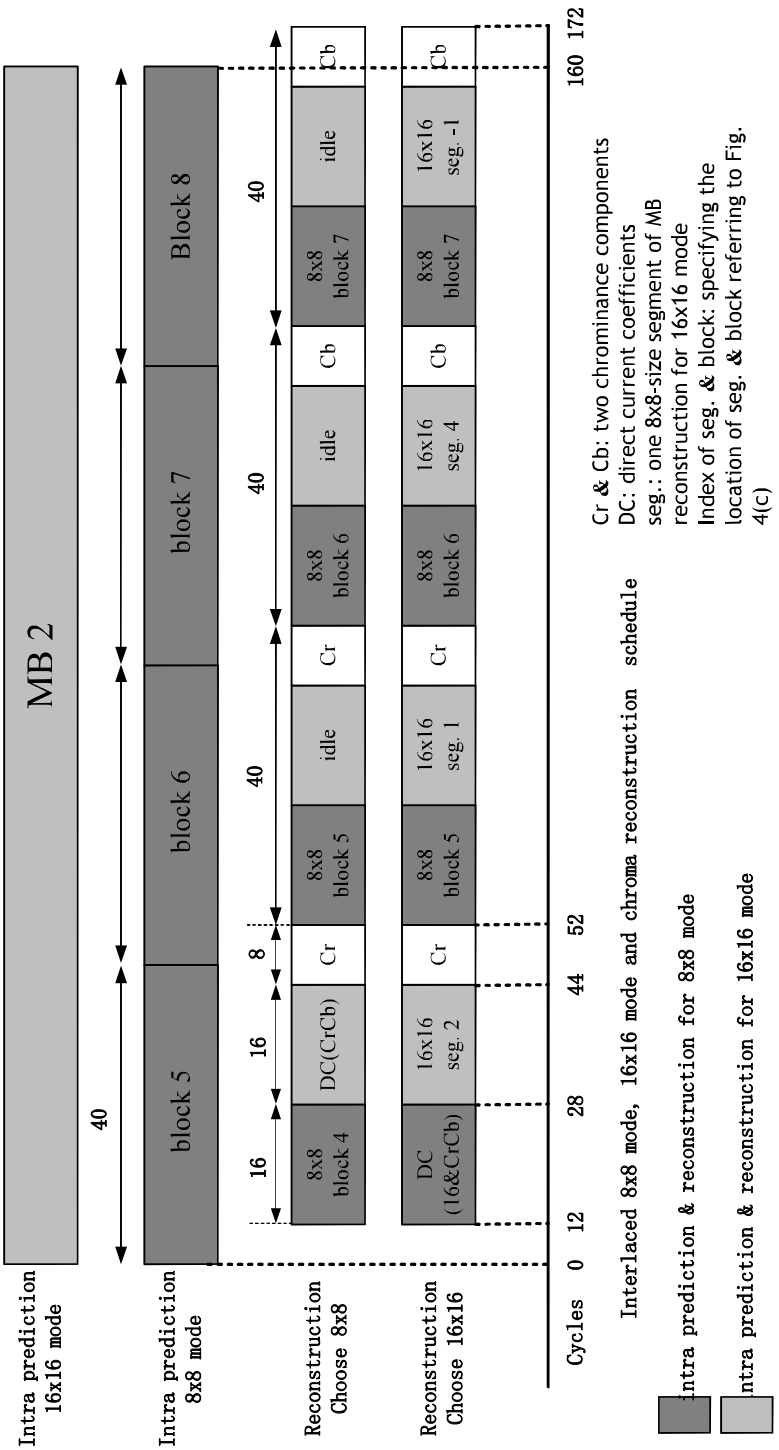


Fig.11. Proposed intra prediction 160 cycles/MB schedule.

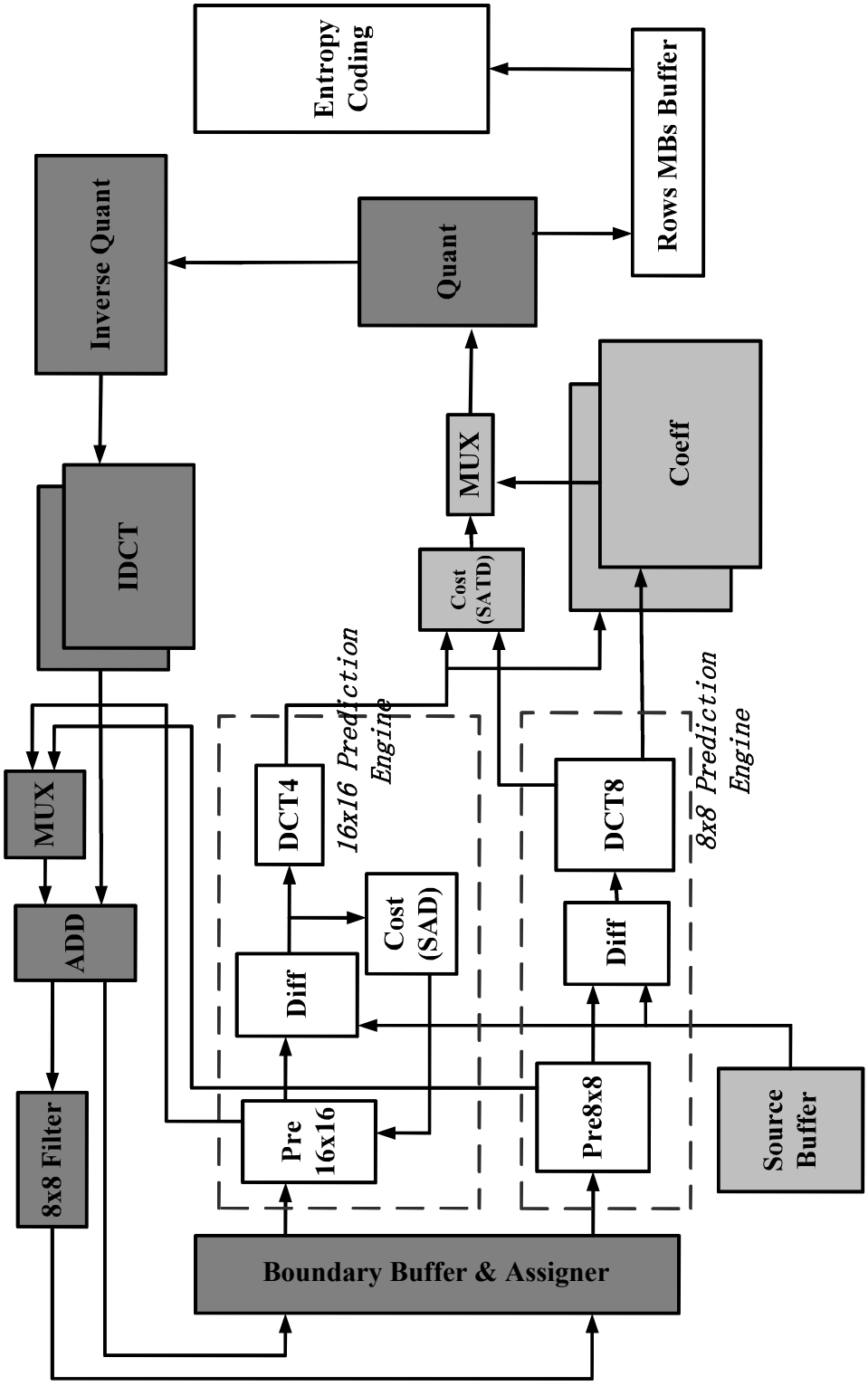


Fig.12. Proposed intra encoding architecture.

2.2.2 PE-Reusable 8x8 Intra Predictor

In the 8x8 intra prediction, two rows of one block are processed by engine at the same time. For each 8x8 prediction mode, there are certain equivalent prediction samples between rows. All 9 prediction modes are divided into 3 classes. The first class contains vertical, horizontal and DC mode. DC prediction value is computed in mode 0 & mode 1 prediction cycles. All prediction values are bypassed in class 1. Figure 8 showsthe prediction direction of each mode in class 2 and class 3. In class 2, six or seven pixels in one row can be obtained directly from its upper row prediction values and only one or two pixels are needed to compute. In class 3, each odd or even row has seven equivalent prediction pixels with its upper odd or even row. Hence, ten pixel prediction elements are used to compute sixteen (two rows) pixels. Taking advantage of prediction mode data characteristic, processing element (PE)-reusable intra predictor is proposed and about 3/8 predicting hardware cost is reduced.

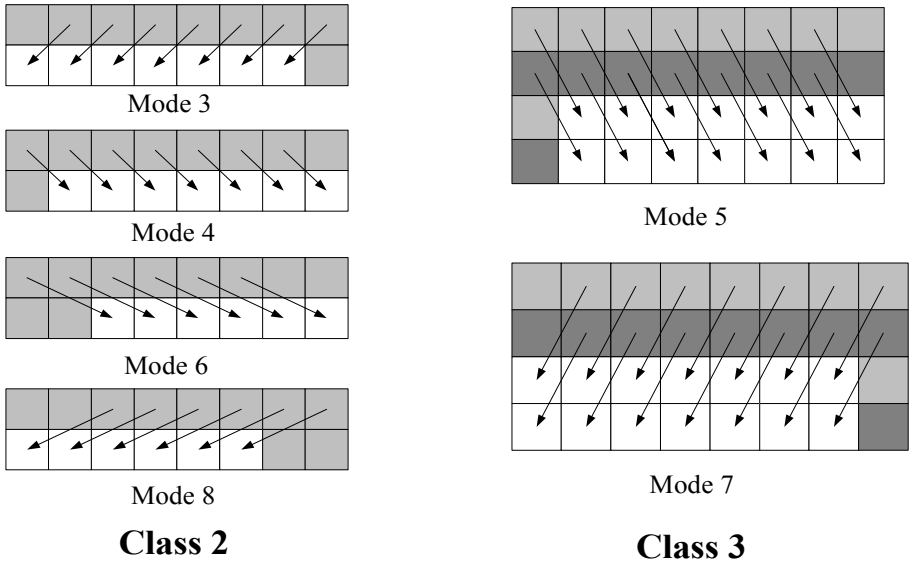


Fig.13. 8x8 prediction pixels relationship.

2.2.3 Hybrid SAD &SATD Mode Decision

In previous work, most of designs [2][3][4][5] choose SATD as mode decision for good performance such as high PSNR and low bit rate. However, SATD includes transform (Hadamard or integer DCT) operation which requires more computation and takes more hardware cost than sum of difference (SAD) mode decision. The hybrid SAD & SATD mode decision is proposed to reduce hardware cost while keeping high performance. In this design, 16x16 prediction has 3 modes and there is nearly no quality loss if SAD is used as internal 16x16 best mode decision. When the best 16x16 mode is selected, its SATD cost is computed and compared with the best 8x8 mode which is decided by SATD to find the final best mode. Figure 10 summarizes this intra prediction mode decision process. In proposed architecture, the 16x16 prediction engine is composed of 4 pixels/cycle SATD sub-engine and 8 pixels/cycle SAD sub-engine. SATD sub-engine consumes most of the hardware cost and its hardware utilization is important. The 16x16 prediction engine performs for both 16x16 luma prediction and 8x8 chroma prediction. While 16x16 intra prediction is being processed by SAD sub-engine in three 32 cycles as shown in Fig. 10, chroma prediction is being processed by SATD sub-engine at the same time. After that, the SATD computation of best 16x16 mode takes 64 cycles. In proposed architecture, the SATD sub-engine reaches 100% utilization.

Table 4 illustrates the comparison results for encoding the 4096x2160 and 1080p sequences. Our proposal makes not only the hardware efficient but also encoding quality good.

Table 4 The increase in BD-bit rate(%) by comparison with JM17.0 (High Profile supporting 4x4, 8x8 and 16x16 mode)

Sequences	IP.8x8 & 16x16	Proposed
DucksTakeOff(4096x2160)	-0.98	-0.85
ParkJoy(4096x2160)	0.30	0.31
InToTree(4096x2160)	-1.04	-0.93
CrowdRun(4096x2160)	-0.94	-0.61
Pedestrian(1080p)	0.84	1.20
BlueSky(1080p)	-0.87	-0.44

Table 5 Logic Gate count distribution of proposed intra prediction Architecture.

Module	Gate count
SAD (16) sub-engine	4670
SATD(16) sub-engine	13662
8x8 Pre. Engine	41730
Quantization	11143
Inverse Quantization	5741
IDCT(8x8&4x4)	11800
Reconstruction	5515
Boundary buffers & assigner	18929
Total	113.2k

2.3 Experimental Results

The proposed architecture is designed with Verilog HDL and synthesized using SMIC 90nm CMOS technology. The total logic gate count is 113.2k and the gate count for each component is listed in Table 5. Most of area is spent on boundary buffer, 8x8 prediction engine, quantization and IDCT. Because we improve the hardware efficiency and only use four-pixel parallelism in Q and IQ for 4096x2160@60fps applications, these units no longer occupy very large area in the whole design. The on-chip memory including source buffer and coefficient buffer between transform and quantization is about 22.5k.

The comparison with previous designs is in Table 6. Compared with previous works, our design is the first hardware architecture that can support 4096x2160 and 3840x2160 real-time encoding at 60 fps when running at 332MHz and 310MHz respectively. Moreover, for HD1080p 30 fps encoding requirement, this design can reduce 72% of operating frequency compared with [3] because of high parallelism and pipeline efficiency.

2.4 Conclusion

In this chapter, we propose a novel intra prediction hardware architecture that can support H.264/AVC 4096x2160@60fps real-time encoding. MB/block co-reordering process is proposed to alleviate data dependency and make the pipeline more efficient with negligible quality loss. Moreover, PE-reusable 8x8 intra predictor, hybrid SAD & SATD mode decision, interlaced reconstruction and other scheduling techniques are applied to improve hardware utilization and save hardware cost. Only 160 cycles are used to process one MB. Compared with previous designs of 1080p@30fps, throughput of proposed design increases for 8 times with small area overhead.

3. An H.264 Intra Prediction Architecture for 8kx4k UHD

3.1 Introduction

Whereas 1080p high definition (HD) is the current main stream of video services such as digital TV broadcasting and home entertainment, even higher resolutions of Ultra-HD such as 3840x2160p and 7680x4320p (also known as Super Hi-Vision) with faster frame rates (e.g. 60fps) have been targeted by next-generation applications. These can provide not only an enhanced sensation of reality, but also rich monocular cues for 3D impression such as visual field angle, linear perspective, and texture gradient [14]. Such massive data have to be compressed by the efficient video coding standard such as H.264/AVC. Intra prediction in H.264/AVC, which uses neighboring pixel values to predict the currently coding block, explores spatial redundancy of the video. This paper focuses on the intra prediction architecture. Firstly, the architecture, together with an entropy coding unit, can be implemented as an intra-frame encoder chip, such as the work done by [35]. This intra-frame encoder is suitable for some portable consumer electronics which cannot afford the complex motion estimation due to the low power and hardware cost [24]. Secondly, a complete encoder chip with inter prediction can be realized by combining the intra encoder with motion estimation and de-blocking filter modules. When inter frames are processed, the intra prediction design can also be applied as an alternative compression tool to improve the coding efficiency. However, to efficiently design the intra prediction architecture for high resolutions is a great challenge from two aspects: 1) the critical data dependency of the intra prediction algorithm causes difficulties in parallelism and pipelining; 2) the high resolution results in huge computational complexity.

Several optimized architectures for intra prediction have been proposed in

[16]-[29]. The contributions for solving the data dependency problem can be briefly classified into three categories. The first strategy is the interlaced schedule for the prediction modes of different sizes [16]-[18][20]-[24]. It improves the pipeline efficiency, but can only be applied to the mode decision part. For example, mode decision of 16x16 prediction mode can be inserted into the pipeline bubbles of 4x4 prediction mode, while reconstruction operation of 16x16 cannot be inserted since it has to be executed after all size mode decisions are performed. The second strategy is to use the original boundary pixels for prediction, instead of reconstructed ones [25][26], so that data dependency can be removed from the mode decision. This causes the degradation of coding efficiency, especially for lower-bit-rate cases. The third strategy is to modify the processing order of 4x4 blocks in a macroblock (MB) by analyzing the data dependency [16][27]-[29]. It increases the opportunity for parallel processing, but cannot be applied to the 8x8 and 16x16 modes that are more important to high-resolution videos.

In this work, we propose an efficient architecture of intra prediction. This design targets 7680x4320p 60fps real-time encoding. The proposed design is characterized as follows. An interlaced block reordering (IBR) scheme, together with preliminary mode decision (PMD), allows the fully parallel operation of the mode decision and reconstruction parts. At the same time, the complexity is reduced by PMD. Furthermore, a probability-based reconstruction (PBR) scheme solves the problem of pipeline latency. Based on the preliminary decision results, reconstruction process is advanced, which improves hardware utilization. In addition, hardware reuse (HR) strategies such as a shared fine decision module and processing element(PE)-reusable prediction generator are applied to further optimize the architecture. As a result, the hardware complexity is reduced by77%. The design processes an MB in an average of 33 cycles and can support up to 4320p 60fps when running at 273MHz. The 1080p 30fps encoding requires less than 9MHz operating frequency, which is much lower than that in previous works.

The rest of this chapter is organized as follows. Section II first reviews H.264/AVC intra encoding and analyzes the data dependency for hardware design.

Section III introduces the proposed optimization schemes and the detailed hardware implementation is discussed in Section IV. Our experimental results are shown in Section V. Finally, Section VI concludes this chapter.

3.2 H.264/AVC Intra Prediction

3.2.1 Overview of Intra Prediction

Fig. 14 shows the H.264/AVC intra-frame encoding flow. Firstly, a prediction generator (PG) unit refers to reconstructed pixels of neighboring block to generate predictive pixels for each mode. The residues are then generated and transformed into coefficients. Based on a given cost function, the costs are calculated to perform mode decision. The quantization unit processes the coefficients of the best mode and outputs the results to both the inverse quantization unit and the entropy coding unit. The inverse quantization and inverse transform units translate quantized values back to residuals, which are used to reconstruct pixels for PG operation of the next block. The entropy coding unit encodes quantized values and mode information for bit-stream output.

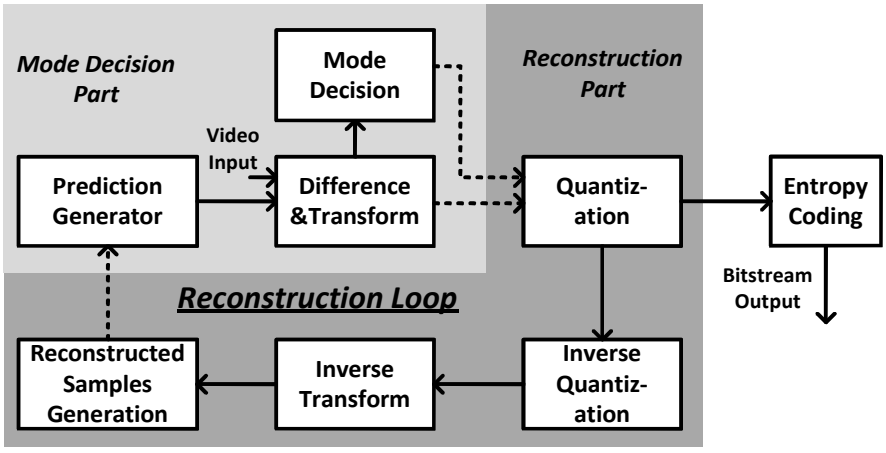
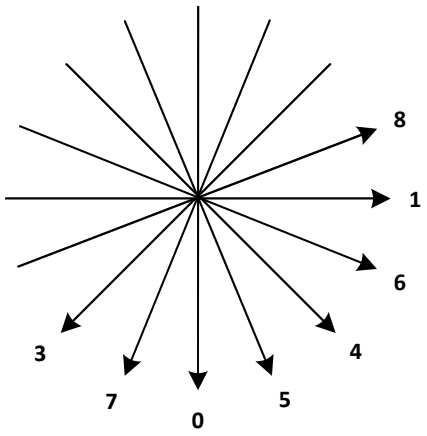


Fig.14. H.264/AVC intra-frame encoding flow.



MODE	DIRECTION	4x4/ 8x8	16x16
0	Vertical	y	y
1	Horizontal	y	y
2(DC)	N/A	y	y
3	Diagonal down-left	y	
4	Diagonal down-right	y	
5	Vertical-right	y	
6	Horizontal-down	y	
7	Vertical-left	y	
8	Horizontal-up	y	
Plane	N/A		y

Fig.15. The H.264 intra prediction modes.

3.2.1.1 Prediction Mode

The prediction modes for luma component are categorized into three sizes. As shown in Fig. 15, both 4x4 and 8x8 predictions contain eight directional and a DC modes, while 16x16 prediction has two directional (vertical and horizontal), a DC and a plane modes. The 16x16 pixels in an MB can be encoded as 16 predicted blocks with the 4x4 mode, or four predicted blocks with the 8x8 mode, or just a predicted block with the 16x16 mode. In addition, four 8x8 chroma modes, which are similar to 16x16 luma modes, are adopted to predict the two Cb and Cr 8x8 blocks within an MB.

3.2.1.2 Cost Function

The optimal cost function for coding efficiency is the rate distortion optimization (RDO), which brings high computational complexity. In the hardware design that targets at real-time coding for high resolution videos, RDO leads to large hardware cost and power dissipation. For lower complexity, the sum of absolute transform difference (SATD) is commonly used in conventional designs. The cost of each mode is estimated with transformed coefficients. The computational formula is defined as follows:

$$Cost = SATD + \lambda(QP) \bullet R \qquad (1)$$

$$SATD = \sum\sum\left|T(Cur - Pre)\right| \quad (2)$$

In (1), R is set to 0 for the most probable mode and to 4 for the other modes. The value of λ is a function of the quantization parameter (QP). In (2), T(x) can be either a Hadamard transform (HT) or an integer discrete cosine transform (DCT). Cur and Pre denote the original and predictive pixels. In addition, the sum of absolute difference (SAD) cost function is also suitable for hardware design. It does not need to perform the transform operation, thus introduces the least complexity.

3.2.2 Data Dependency Analysis

3.2.2.1 Parallelism of Mode Decision and Reconstruction

The realization of efficient hardware is challenged by the reconstruction loop shown in Fig. 14. To generate a predictive block, its neighboring left, upper and upper-right reconstructed pixels are used. We divide all modules into mode decision (MD) and reconstruction (REC) parts. Fig. 16 (a) illustrates the standard zig-zag scanning order for 8x8 blocks in H.264/AVC intra-frame coding. Fig. 16 (b) shows the processing schedule with this order. The MD process of each block needs reconstructed pixels from the previous block, which makes the MD hardware idle during the REC period. Meanwhile, the REC part is idle during the MD operation. The critical data dependency limits the parallelism of MD and REC. This problem must be solved to allow an efficient architecture.

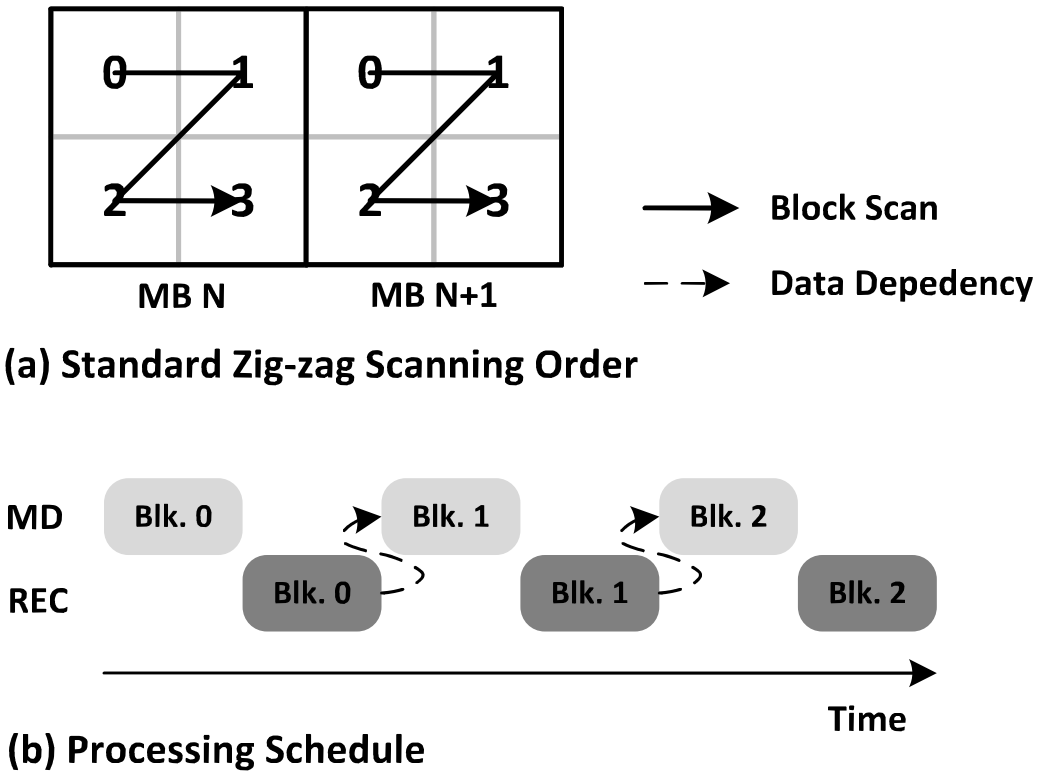


Fig.16. (a) Standard zig-zag scanning order for 8x8 block (b) Processing schedule with data dependency

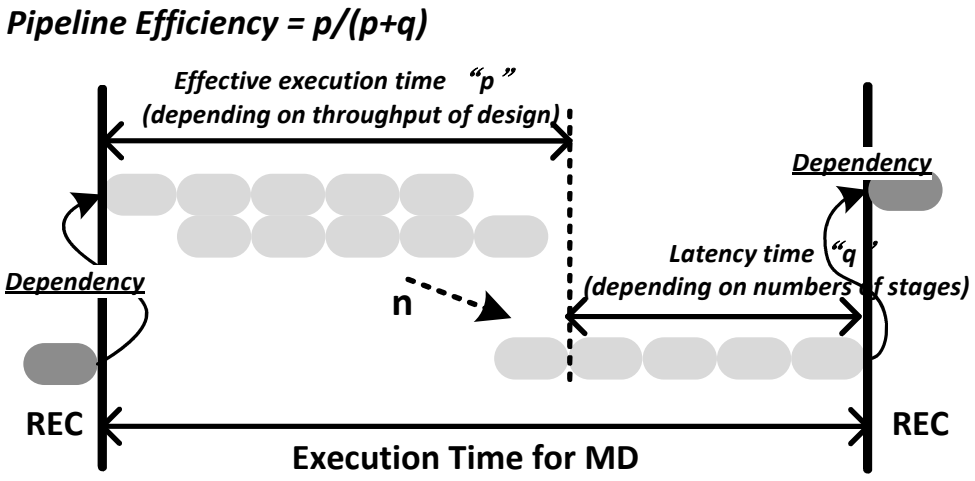


Fig.17. Pipeline latency problem.

3.2.2.2 Pipeline Latency

Pipeline latency influences the efficiency of pipelines with serious data

dependency. The problem becomes critical for 4320p60 encoding, due to the limited processing time budget. Fig. 17 shows the principle of pipeline efficiency with the influence of latency. During the MD execution time, the efficiency can be calculated as *p/(p+q)*, where *p* and *q* denote the effective execution time and the latency of the pipeline, respectively. For the pipeline design of intra prediction, *p* is mainly determined by the throughput requirement. Meanwhile, *q* depends on the total number of pipeline stages, which is determined by the maximum operating frequency and is not so relevant to the throughput. For HD-oriented design, the processing time budget for each MB can be hundreds of clock cycles. As a result, *q* is very small compared to *p*, and thus has nearly no influence on the pipeline efficiency. However, for 4320p60 encoding, every MB should be processed in only 30~40 clock cycles. Assuming a 4-cycle pipeline latency and an 8-cycle execution time for each 8x8 block, the decrease in pipeline efficiency due to the latency can be 33%.

Table 6 BD-bitrate and BD-psnr [38] Difference between 8x8/16x16 and 4x4/16x16 Modes in JM17.0 (RDO-off, Four QPs 22, 27, 32, 37)

Videos		BD-bitrate (%)*	BD-psnr (dB)*
4320p	Nebuta1	-18.82	1.04
	Nebuta2	-15.57	0.77
	Train1	-12.42	0.47
	Train2	-13.31	0.53
2160p	CrowdRun	-9.08	0.41
	DucksTakeOff	-7.83	0.25
	Splash	-12.19	0.64
	ParkJoy	-9.34	0.36
1080p	Pedestrian	-13.48	0.45
	Station	-18.41	0.76
	BlueSky	-5.2	0.29
	Tractor	-13.57	0.78
Average		-12.44	0.56

*: Positive value for BD-bitrate and negative value for BD-psnr mean

8x8/16x16 modes provides better coding performance than 4x4/16x16 modes.

3.3 Algorithm and Hardware Optimizations

Table I shows the coding performance comparison between 8x8/16x16 modes and 4x4/16x16 modes for the high resolutions from 1080p to 4320p. As the experimental results show, 8x8 prediction mode can provide better coding efficiency than 4x4 prediction mode for high resolution videos. Our design targets for ultra-HD video applications such as 4320p and 2160p. Therefore, 8x8 and 16x16 modes are supported.

3.3.1 Interlaced Block Reordering

As explained in Section II.B, under the standard scanning order, the difficulty of parallelizing MD and REC processing is one of the major obstacles to realizing a high level of hardware utilization. In this design, an interlaced block reordering (IBR) scheme is proposed to solve the problem.

Firstly, we assume that only 8x8 mode prediction is being used. As the processing order of 8x8 blocks within an MB cannot be modified, the 8x8 block scan is allowed across MB boundaries. In this way, any block, even one outside the current MB, can be processed only if the referenced pixels are reconstructed. Fig. 18 (a) shows the data dependency of blocks within two consecutive MBs. The dependency of a block comes from its left, upper and upper-right sides. Firstly, the problem of the upper and upper-right sides can be solved if the block in the upper-row is processed two blocks before the one in the down-row. Secondly, to avoid the dependency on the left side, the blocks of two rows should be interlaced with each other. Thus, the scanning order is set down, as shown in Fig. 18 (b). The modified scanning order ensures no dependency between two sequentially

processed blocks, which parallelizes the MD and REC. This is illustrated in Fig. 18 (c). For the current block *N*, a non-reference block *N-1* is inserted in the processing order, between the current block *N* and its neighboring reference block *N-2*. By doing so, the MD of block *N* can start immediately after that of block *N-1* without any idle time, as long as the REC of block *N-2* is completed during the MD of block *N-1*.

Further, we now take the 16x16 mode into consideration, but still assume the following two statements: (a) the choice of whether to use 8x8 or 16x16 partition is decided prior to MD; (b) if 16x16 partition is selected, the best 16x16 is also decided. Based on this information, MD focuses on deciding the best mode for the 8x8 blocks.

The scanning order shown in Fig. 18 (b) cannot be applied to MBs with 16x16 partition. This is because, when the right MB (R-MB) chooses 16x16 partition, the MD of each block within the R-MB cannot commence until the REC of the right blocks within the left MB (L-MB) has finished. Fig. 18 illustrates the four possible combinations of 8x8/16x16 partition of consecutive MBs. The problem with R-MB 16x16 partition is discussed by considering two cases as follow: (1) L-MB chooses 16x16 partition (case (b) in Fig. 18). There is no data dependency between the blocks within the MB that chooses 16x16 partition. Therefore, the solution is to first process those right blocks that have data dependency with the next MB. By doing so, for all MBs with 16x16 partition, the scanning orders will be arranged as shown. In addition, with this reordering, there is no problem of data dependency in case (d); (2) L-MB chooses 8x8 partition (case (c) in Fig. 18). It is infeasible to process the blocks in R-MB before the REC of L-MB has finished. This can be solved by supporting the vertical mode only for R-MB, as it only requires the upper reference pixels. Using IBR, MD and REC can operate in parallel at the 8x8 block level. The contribution to the pipeline design will be analyzed in Section IV.A.

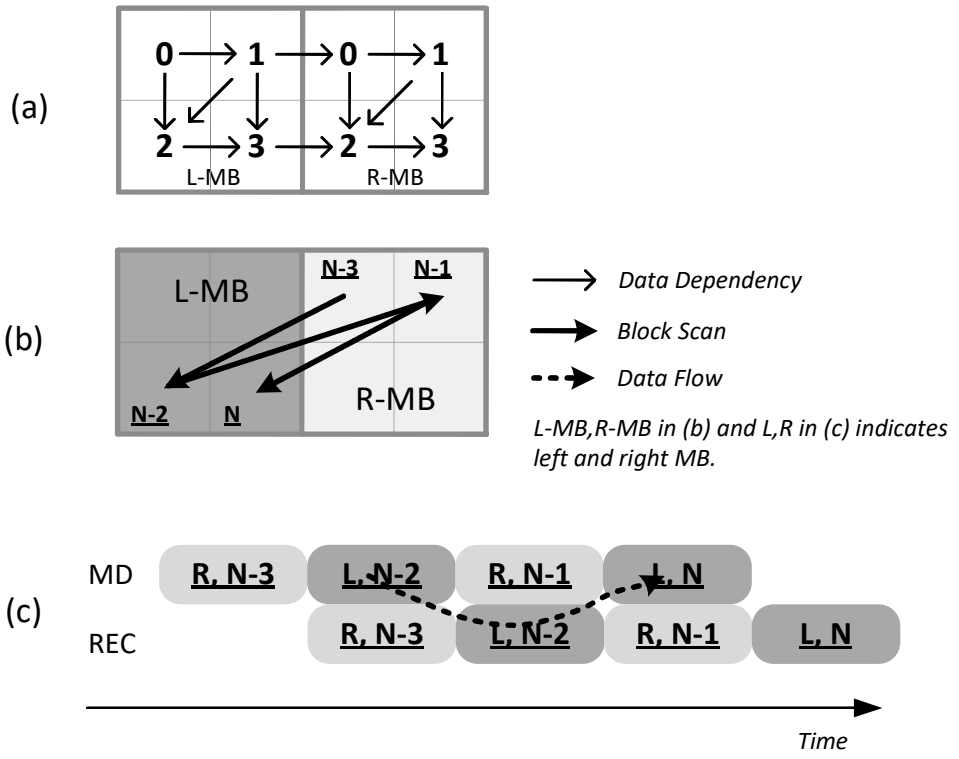


Fig.18. (a) Data dependency of the 8x8 block. (b) Zig-zag interlaced block reordering. (c) MD/REC parallel processin

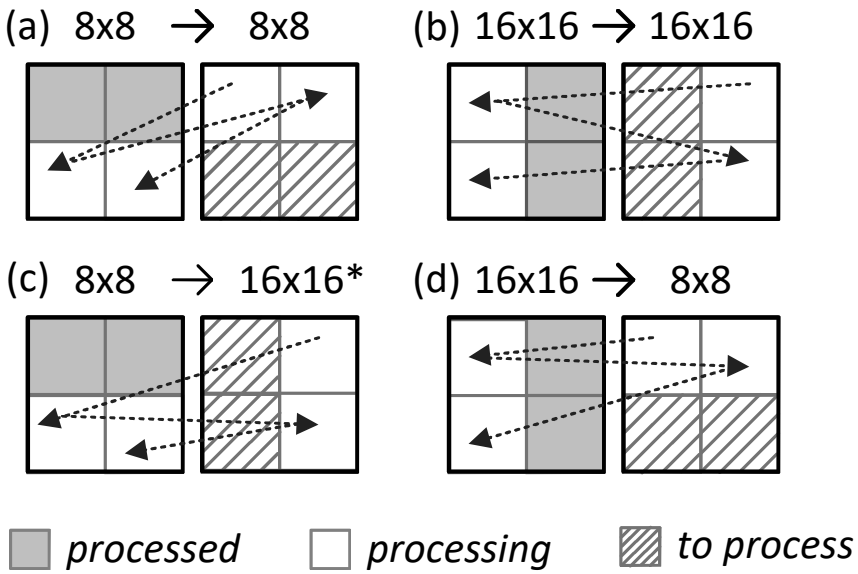


Fig.19. Interlaced block reordering for variable MB-partitions.

In addition, IBR avoids data dependency with small overhead of hardware cost. The overhead for this block-level reordering is the additional storage, which is

determined by the operation region. IBR operates in a region of two MBs, and thus introduces an additional storage of an MB. According to the analysis of data dependency, the schemes with larger regions cannot provide more benefit to the pipeline design, but brings more overhead of hardware cost. For example, if we choose a region of four MBs, the pipeline efficiency is the same with IBR, while an additional storage of three MBs is needed. The scheme of MB-level reordering can also avoid the dependency by reordering MBs within consecutive MB-rows. In this case, a DRAM buffer system is needed to reorder the data of MB-rows, which leads to additional DRAM bandwidth and power consumption.

It should be noted that, the design with IBR can compress videos to bit streams that are fully conformed to the H.264 standard. The reordering is only applied to the intra prediction progress. During the progress, the reordering does not break the data dependency defined by the standard. Each block is still processed with an H.264 prediction mode by referring to the neighboring reconstructed pixels. After the intra progress, we reorder the output coefficients to the standard processing order before the entropy coding unit.

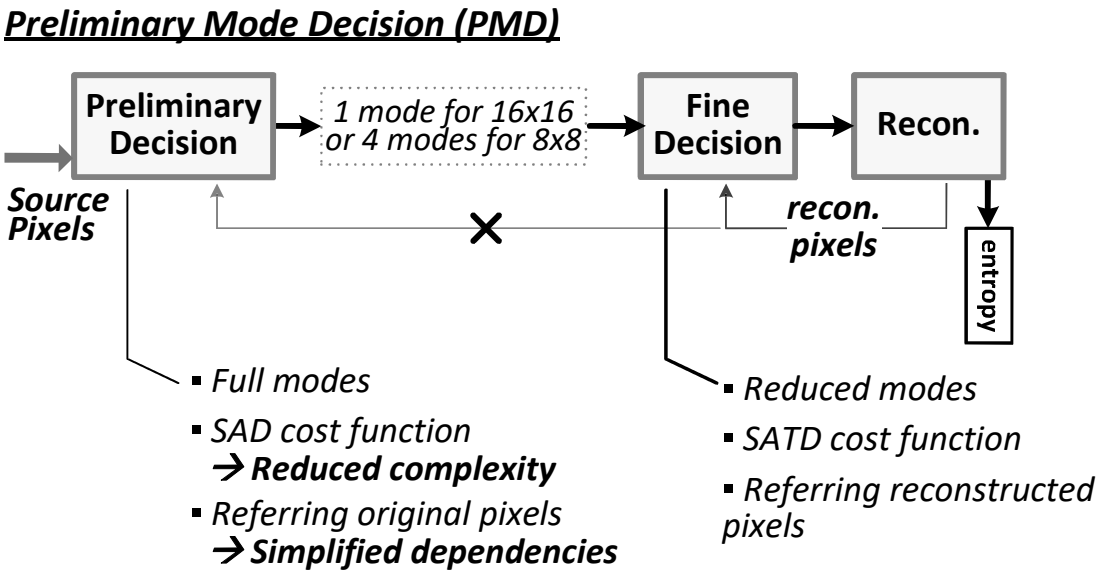


Fig.20. The scheme of preliminary mode decision.

3.3.2 Preliminary Mode Decision

IBR aims to enable parallelism between MD and REC, but it works on the assumption that MB partition and the best 16x16 mode are already available prior to MD. We introduce a pre-processing stage to obtain this information. Obviously, as the purpose of IBR and pre-processing is to resolve the influence of data dependency, the pre-processing stage should not introduce new dependency problems.

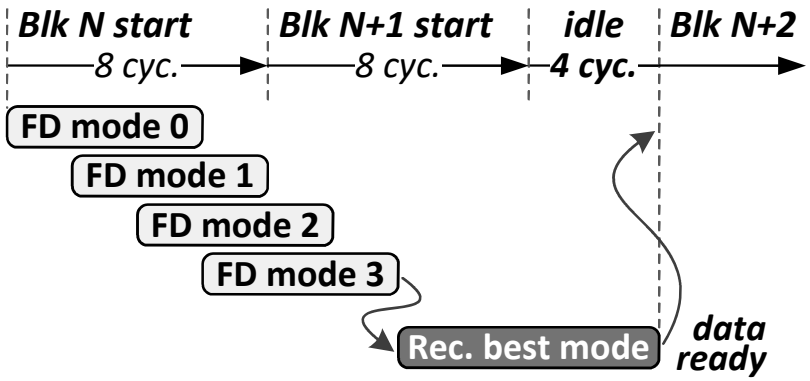
In addition, the operations in the MD require many iterations to be processed. A straightforward implementation suffers from high complexity and many pipeline stages. An efficient way to prevent this is to reduce the candidate modes processed in the MD stage, which also require pre-processing. Because our target here is complexity reduction, the overhead introduced by pre-processing must be small.

To work together with IBR and reduce complexity, we propose the preliminary mode decision (PMD) scheme. Using original pixels as the reference, which involves no data dependency, all intra modes are computed and compared in the preliminary decision (PD, as the pre-processing). Based on the evaluation in PD, a reduced number of candidate modes are sent to the fine decision (FD) stage for further refinement which refers to the reconstructed pixels. With PMD and IBR applied, PD, FD and REC can work in parallel. In addition, this approach can reduce the computational complexity. As PD only generates candidate modes, instead of making final decisions, a relatively inaccurate cost function is acceptable. Therefore, SAD is applied for PD. Meanwhile, FD still uses the accurate SATD, but it only processes a reduced number of prediction modes.

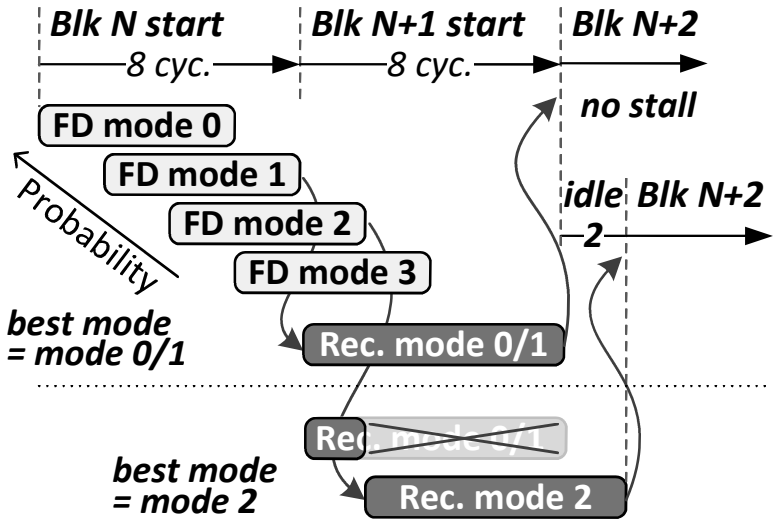
In our design, PD decides the MB partition, and generates the four best candidate modes for FD if the MB is partitioned into 8x8 blocks. If 16x16 partition is selected, the best 16x16 mode is passed directly to FD. The scheme is illustrated in Fig. 20. Due to the mode reduction of FD and the simpler cost function used by PD, the complexity is reduced compared with the straightforward MD design. Furthermore,

PD involves no data dependency, and thus can work in parallel with FD and REC. The processing speed can be increased. The optimization on hardware implementation will be explained in Section IV.A.

In addition, different from the method in [25][26] that use original pixels to decide the best mode, PMD supports a fine decision with reconstructed pixels. This involves less degradation of coding performance.



(a) Straightforward pipeline design.



(b) Pipeline with probability-based reconstruction (PBR).

Fig.21. (a) Straightforward pipeline design. (b) Pipeline with probability-based reconstruction (PBR)

Table 7 Probability Statistics with QP 32

Seq.	MB-partition statistics (%)		Rate distribution of the best mode with 8x8 partition (%)		
	16x16	8x8	Mode 0/1	Mode 2	Mode 3
Nebuta (4320p)	27.18	72.82	81.93	11.37	6.70
Train (4320p)	62.59	37.41	82.33	10.95	6.72
DucksTakeOff(2160p)	8.48	91.52	89.16	6.81	4.03
CrowdRun (2160p)	25.25	74.75	84.64	9.64	5.72
Station (1080p)	42.35	57.65	88.54	7.47	3.99
Tractor (1080p)	12.30	87.70	87.93	7.91	4.15

Table 8 Equivalent Encoding Speed under Different Situations

QP	Nebuta	Train	Ducks-Take Off	Crowd-Run	Station	Tractor
37	33.39	32.50	33.02	33.23	32.73	33.40
32	33.43	32.73	33.09	33.26	32.71	33.14
27	33.26	32.88	32.94	33.14	32.72	32.94
22	33.18	33.08	32.76	33.05	32.86	32.86

3.3.3 Probability-Based Reconstruction

As mentioned, due to the limited processing time budget for 4320p60, the problem of pipeline latency becomes critical. Fig. 21 (a) shows the pipeline design targeted at processing each 8x8 block in eight cycles. With 6-stage FD pipeline, four prediction modes are processed in the pipelining manner and each one is processed with a 2-cycle delay. After the best mode is decided, REC takes eight cycles for a block. Owing to IBR, the reconstructed pixels of block N are not needed until block N+2 starts. But this still leads to a 4-cycle being idle for every two blocks. We propose probability based reconstruction (PBR) to shorten this idle time. As shown in Fig. 21 (b), REC starts as soon as FD finishes the first two modes. If a subsequent mode turns out to have a lower cost, REC is promptly restarted for the new mode. As a result, the idle time is zero, zero, two or four cycles when mode 0, 1, 2 or 3 is the best. In our design, the FD candidate modes are calculated and decided by PD with the SAD cost function. The mode with a smaller SAD cost has a higher probability of being the best mode. Therefore, to minimize the idle time, the modes are processed in ascending order of SAD costs.

Table II shows the MB-partition statistics and distribution of the best mode for 8x8 partition. It should be mentioned that the operation on the blocks of the 16x16-partition MB does not introduce any idle time since only one mode is processed. For the blocks of the 8x8 partition MB, the first two candidate modes own about 86% probability averagely of being the best.

In the design, the processing time of four sequential blocks can be considered as equivalent to the encoding time for an MB. After applying PBR, the speed varies from 32 to 40 cycles depending on the MB-partition and hit-rate (first two modes being the best). It takes 32 cycles if the four sequential blocks are the blocks from 16x16 partition or hit-blocks from 8x8 partition. At most, 40 cycles are taken if the four blocks are from 8x8 partition and each block chooses the last candidate mode as the best mode. Owing to the high probability of hit-rate, the average speed

should be close to 32 cycles. Table III shows the simulation results of processing speeds under four QPs (37 32 27 22) with six video sequences. It varies from 32.5 to 33.43 cycles and the worst situation is the Nebuta video with QP 32. The average time is 33.02 cycles, an 18% reduction from the original 40. The PBR optimization for the pipeline design is explained in detail in Section IV.A.

We now discuss the processing speeds in both the local and statistical worst cases. In the local worst case, four sequential blocks are from 8x8 partition MB and each one chooses the last candidate mode as the best mode. As we have analyzed, this takes 40 cycles. However, we should consider the statistical processing speed, which is related to the operating frequency in real-time encoding applications. In our design, FD candidate modes are processed in ascending order of costs calculated in PD. Statistically, the mode with a smaller SAD cost has a higher probability of being the best mode. The extreme case is when there is no relationship between this ascending order based on SAD costs and the best mode chosen with the SATD cost function, which means the four candidate modes each have 25% probability. If the best modes are all mode 0, 1, 2 or 3, the encoding speed will be 32, 32, 36 or 40 cycles/MB, respectively. Thus, in the statistical worst case with even probability for each mode, an average of 35 cycles is taken for an MB. In this case, it requires a 273 operating frequency for the specification of 4320p 60fps. If we take the 16x16 partition into consideration, the processing time should be less than 35 cycles. In this design, the maximum frequency of the implemented design is 300MHz. It can support up to 37.2 cycles/MB for 4320p 60fps, which is enough for real-time processing.

3.4 Design Implementation

In this section, the detailed hardware implementation is discussed. Pipeline designs after applying proposed schemes are analyzed one by one in subsection IV.A. Then system pipeline and architecture are summarized and descried in subsections IV.B and IV.C. In addition, two hardware reuse (HR) strategies are explained in subsections IV.D and IV.E.

3.4.1 Pipeline Analysis

To realize the target application, we analyze the data throughput and the computation cycles before the hardware design, as shown in Table IV. Targeting 4320p 60fps, our design is required to have capability to achieve the data throughput of 1991 Mpixels/s, which is equivalent to 7776 KMBs/s. When running at a generally acceptable frequency of 250MHz, the timing budget for an MB is limited to 32 cycles. This speed requirement is far beyond than that of HD.

Table 9 Estimation of Timing Budget for Different Specifications

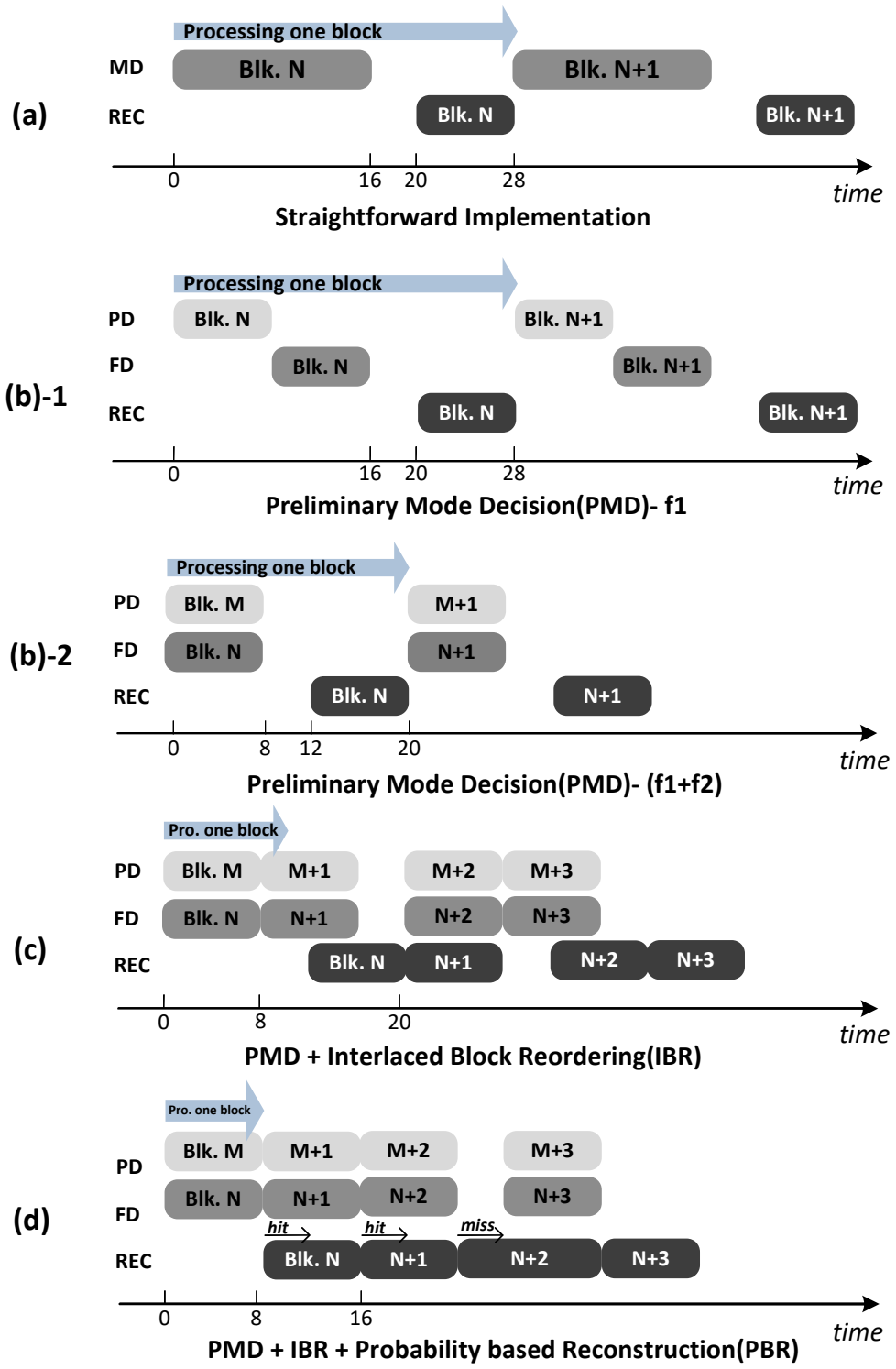
Resolution	Data Throughput (K-MBs/sec)	Timing Budget* (cycles/MB)
HD (1920x1080@30fps)	243	1029
QHD (3840x2160@60fps)	1944	128
UHD (7680x4320@60fps)	7776	32

*: Assuming 250 MHz working frequency.

Table 10 The Optimization of Required Frequency

Techniques	Cycles per block	Cycles per MB	Required Frequency*
(a) Straightforward Implementation	28	112	870MHz
(b) PMD	20	80	622MHz
(c) PMD+IBR	10	40	311MHz
(d) PMD+IBR+PBR	8.3	33.02	257MHz

*: For the 7680x4320p 60fps specification.



PMD-f1: simpler SAD cost function in PD, feature 1
PMD-f2: PD prediction by referring original pixels, feature 2

Fig.22. Pipeline schedules of designs with different technologies.

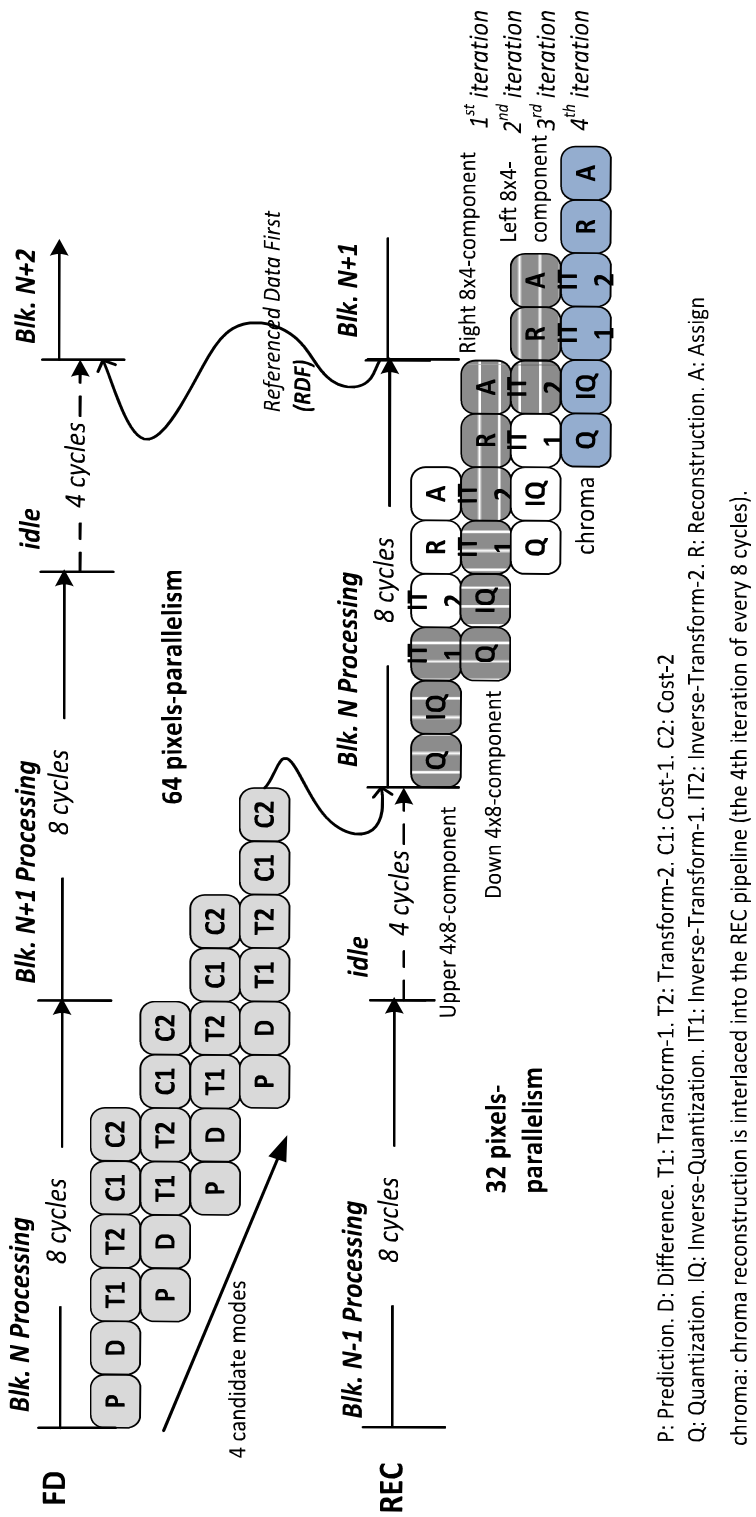


Fig.23. Detailed pipeline latency problem.

For an efficient architecture that achieves such high throughput, proposed schemes are applied to optimize the pipeline design in aspect of the required

frequency and hardware cost. The pipeline schedules for designs using different techniques are illustrated in Fig. 22. Table V shows the corresponding required frequency for the specification of 4320p 60fps.

This discussion is first limited to luma processing since chroma part is processed to cooperate with the optimized pipeline of luma processing. For luma processing, operations can be performed at the block level. For each block, there are 12 prediction modes, made up of nine 8x8 modes and three 16x16 modes. To simplify the discussion, we assume that the 8x8 partition is chosen for each MB after the MD operation. With this assumption, REC processes only the best 8x8 mode.

3.4.1.1 Pipeline Design of Straightforward Implementation

For the hardware design of intra prediction, the major obstacle is the parallelism of MD and REC, which results in low hardware utilization and long latency for waiting. An example of pipeline schedule in a straightforward implementation (SI) is shown in Fig. 22 (a). For each block, MD and REC take 16 and 8 cycles, respectively. In addition, the MD pipeline involves 4-cycle latency before REC starts. In total, 28 cycles are taken for each block. The encoding speed achieves 112 cycles/MB. In this case, the required frequency for 4320p60 encoding is 870MHz.

3.4.1.2 Pipeline Design with PMD

To analyze the contribution of the proposed schemes clearly, PMD is first applied to optimize the design. PD processes all intra modes and provides candidate modes to FD for further refinement. Two adopted features contribute to the optimization of hardware design. The first is the simpler cost function used by PD, denoted as PMD-f1. Without the transform operation unit and transport buffering memory, SAD takes less than a third of the area of SATD. With this applied, the overall hardware cost is reduced by 15% compared with SI. The pipeline schedule with PMD-f1 is shown in Fig.22 (b)-1, where PD and FD take eight cycles to process a block. The second feature is the PD prediction by referring to the original pixels, denoted as PMD-f2. No data dependency is involved in PD, so it can process

the data prior to provide MB-partition and candidate modes. Furthermore, PD and FD can operate in parallel. The system pipeline schedule is shown in Fig.22 (b)-2, where blocks M and M+1 are processed by PD. After applying PMD, it takes 20 cycles to process one block, which reduces the required frequency to 622 MHz.

Fig. 23 illustrates the pipeline schedule of PD. The 4-stage (P-D-C1-C2) pipeline is designed due to the SAD cost function. With 96-pixel parallelism, the pipeline can process 12 prediction modes within 8 cycles for each block. As PD does not involve any dependency, it can process blocks prior and work in parallel with FD. In addition, it will be further optimized by applying fine decision module sharing (FDMS) explained in Section IV.D. After that, PD processes only eight prediction modes, which reduces the pixel parallelism of pipeline from 96 to 64.

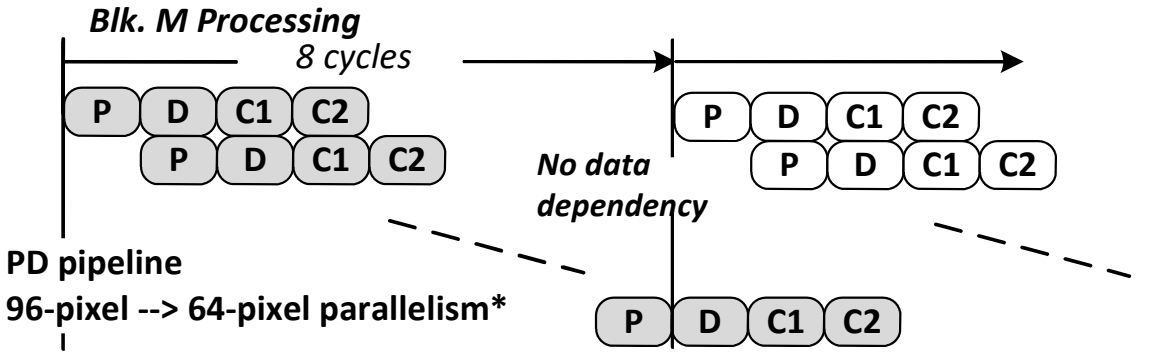
3.4.1.3 Pipeline Design with IBR

IBR is proposed to parallelize MD (which here indicates FD) and REC. It ensures there is no data dependency between two sequentially processed blocks. The pipeline schedule is shown in Fig. 22 (c). Owing to IBR, the reconstructed pixels of block *N* are not needed until block *N+2* starts. In this way, 20 cycles are taken to process two sequential blocks. The equivalent encoding time for an MB is reduced to 40 cycles, which contributes to a 50% reduction in the required frequency.

3.4.1.4 Pipeline Design with PBR

After applying PMD and IBR, the pipeline latency becomes the critical problem, as shown in Fig. 22 (c). To clearly illustrate this, we describe the detailed pipelines of FD and REC in Fig. 24. The 6-stage (P-D-T1-T2-C1-C2) and (Q-IQ-IT1-IT2-R-A) pipelines are designed for MD and REC, respectively. Four candidate modes are processed by the FD pipeline with 64-pixel parallelism. To improve the hardware utilization, transform unit is made reusable for the first and second dimensional transform operations (T1 and T2). By doing so, the utilization of transform unit is greatly improved. Other units can achieve the same utilization by applying the proposed FDMS (explained in Section IV.D). In this way, the

candidate modes are processed in the pipelining manner and each one is processed in a 2-cycle delay. It takes two cycles to evaluate a mode. Only the best mode is processed by the REC pipeline with 32-pixel parallelism. Q and IT are made configurable to process Q/IQ and IT1/IT2, since these two units composes 90% of the area of REC. The pipeline processes each iteration with a 2-cycle delay. To process an 8x8 block, it should take two iterations with the pipeline of 32-pixel parallelism. However, due to the transpose in two-dimensional transform, an additional iteration is required. For each block, the second dimensional transform operation cannot start until the first dimensional one has finished. As shown in Fig. 24, the operations of Q, IQ and IT1 are performed in the first and second iterations for the upper and down 4x8-componets of a block. After that, the operations of IT2, R and A are executed in the second and third iterations for the transposed right and left 8x4-components. In addition, to optimize the pipeline schedule, reference data first (RDF) scheme [16] is applied in the REC pipeline. The IT2, R and A in the second iteration process critical reference component (right 8x4-component) first, so that REC can provide required reconstructed data within 8 cycles.



P: Prediction. D: Difference. C1: Cost-1. C2: Cost-2
*: The pixels-parallelism is reduced from 96 to 64 by FDMS, explained in Section IV.D

Fig.24. Pipeline schedule of PD.

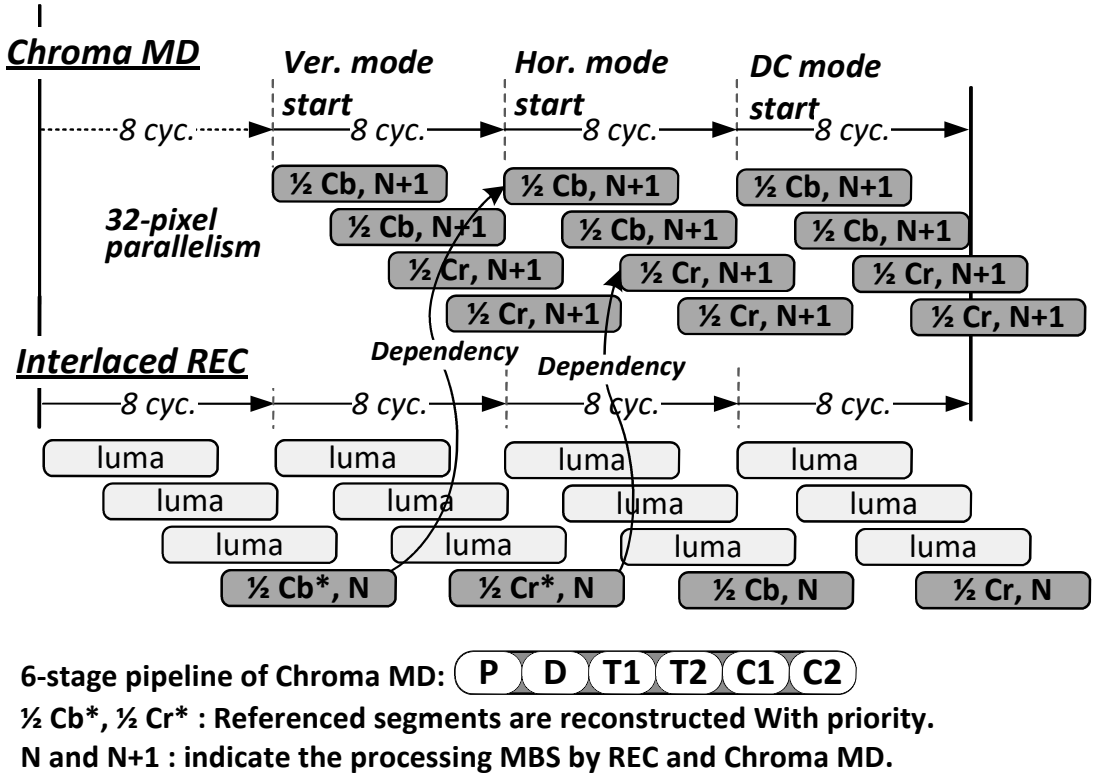


Fig.25. Pipeline schedule for chroma processing.

This pipeline design aims to process each block in 8 cycles. However, due to the data dependency of intra prediction, the latency of the FD pipeline introduces a 4-cycle idle for every two sequential blocks in the schedule, as shown in Fig. 24. For example, REC waits four cycles before the processing of block *N*, since it cannot start until the FD of block *N* has finished. FD has to be idle for 4 cycles to wait for the referenced data of block *N* before it processes block *N+2*. To shorten the idle time caused by the pipeline latency, PBR is applied. REC starts as soon as FD finishes the first two modes. It removes the idle cycles if these two modes hit the best mode. Fig. 22 (d) shows the modified pipeline schedule where the hit happens on blocks *N* and *N+1*, while the miss happens on block *N+2*. Due to the high probability of the hit-rate, the processing time for an MB is reduced to an average of 33.02 cycles, which reduces the required frequency to 257MHz.

The above discussion is based on the assumption that the 8x8 partition is chosen for each MB. If the 16x16 partition is chosen, the MB can also be processed at the

8x8 block level. The T and IT units in the FD and REC pipelines are configured to process four 4x4 operations [34]. After PD decides the best mode in advance, FD processes the blocks with only one candidate mode in a modified scanning order. REC processes blocks as for the 8x8 partition. In addition, the entire MB including four 8x8 blocks has to be processed by FD to obtain DC coefficients during the operation time of the first processed block. This is because the DC component is used in the REC operation of each block.

We now consider the pipeline design of the chroma part. Firstly, the chroma REC is processed with the REC luma pipeline. It can be observed that in Fig. 24 for the REC pipeline, four iterations are taken in every 8 cycles and the first three are used for luma. To remove the pipeline bubble, the chroma part of an MB is divided into four segments (two 4x8 Cb and two 4x8 Cr) and each segment is inserted into the fourth iteration of every 8 cycles, as illustrated in Fig. 26. To avoid data dependency, the critical reference segments are reconstructed with priority. Secondly, the 6-stage pipeline (P-D-T1-T2- C1-C2) is designed for the chroma MD with SATD cost function. With 32-pixel parallelism, the pipeline processes a chroma prediction mode within four iterations, which takes 8 cycles in the pipelining manncer. In addition, the processing order for chroma prediction modes is also arranged to avoid the dependency. The vertical mode, which does not refer to the reconstructed pixels of former MB, is processed first. Then, horizontal and DC modes are processed when their referenced pixels are ready. It should be mentioned that the discussion is based on the processing speed of 8 cycles/block. In other cases with additional processing time, the chroma MD pipeline should be idle.

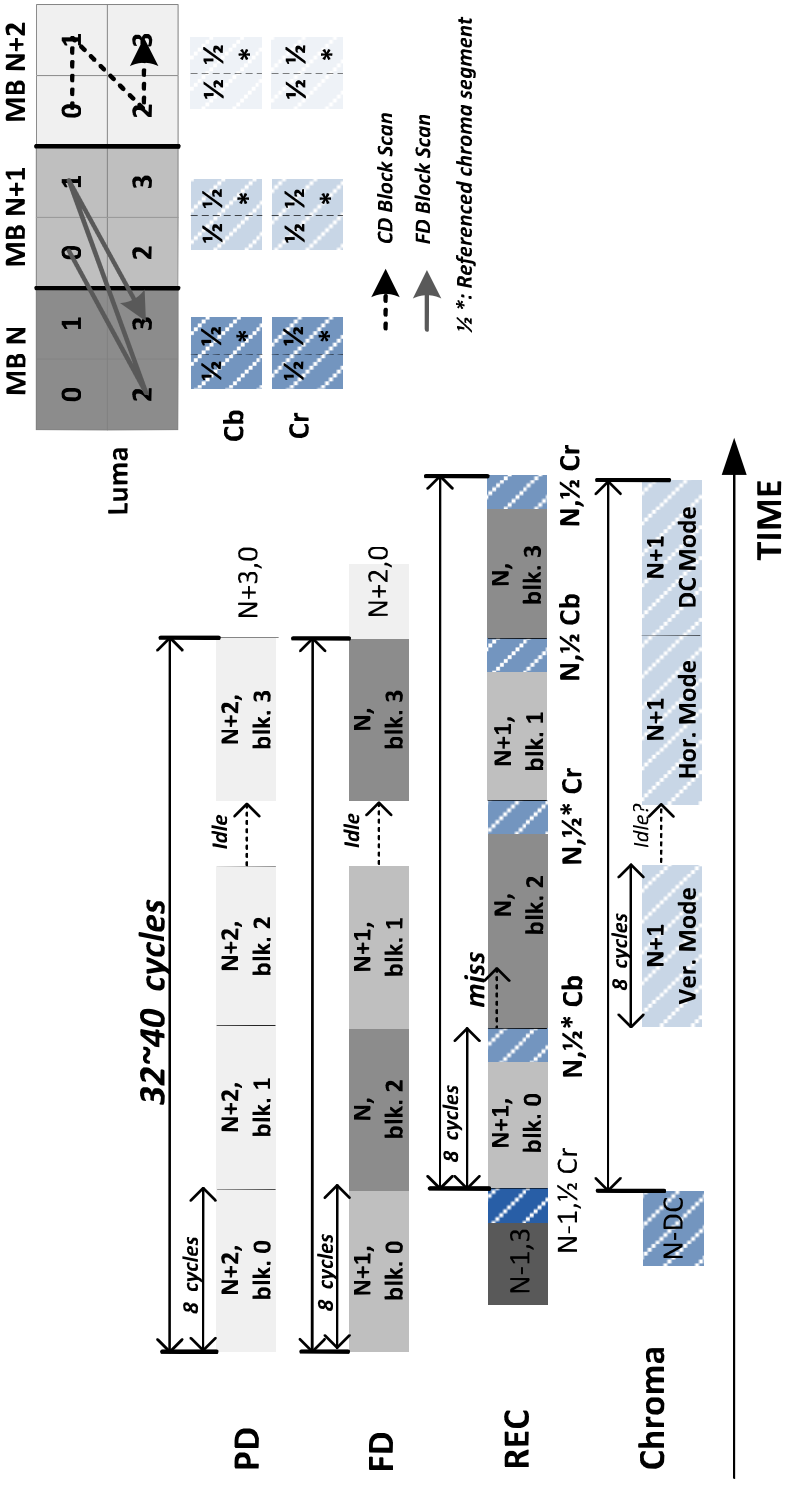


Fig.26. Overall pipeline schedule of the design.

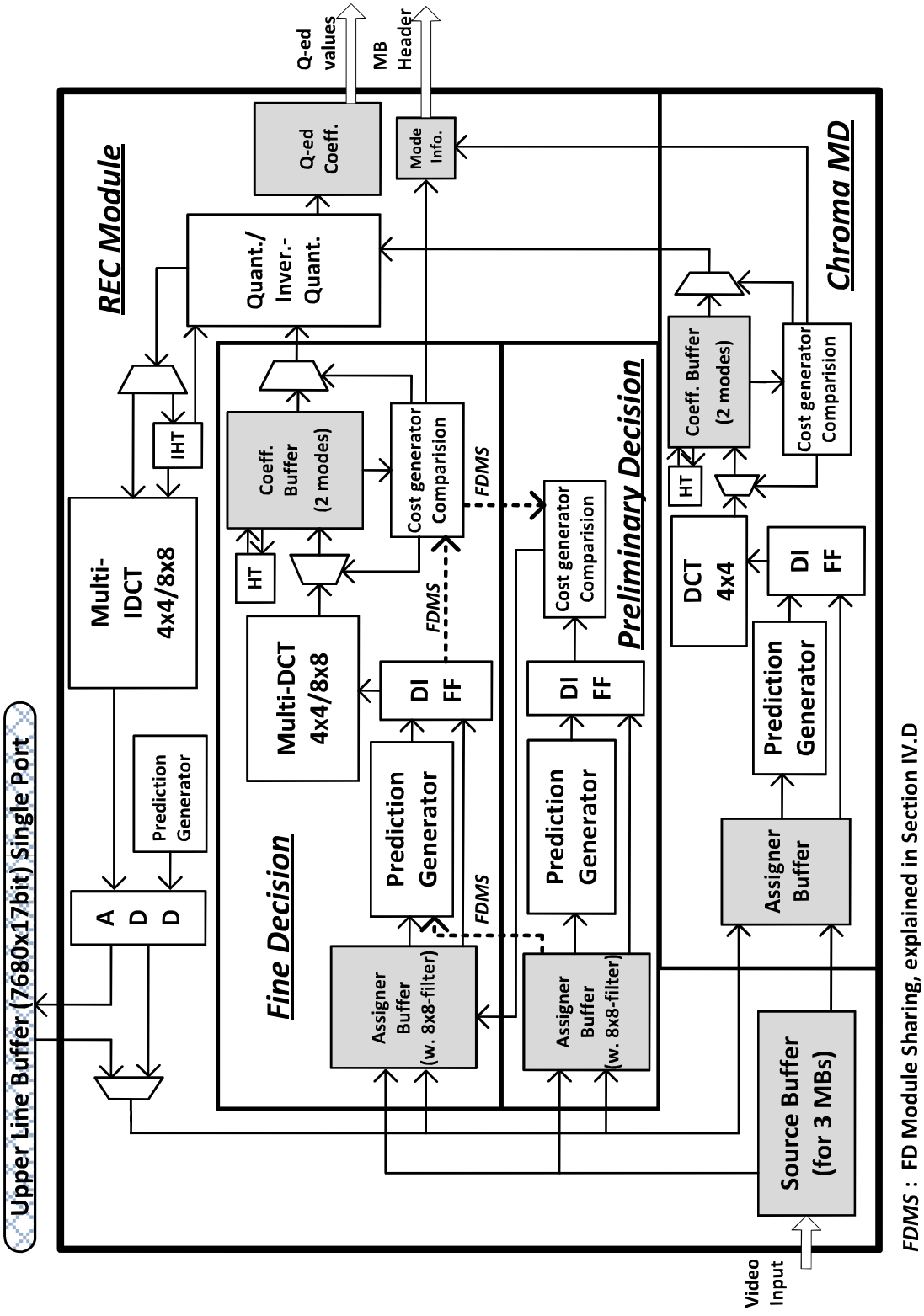


Fig.27. System architecture.

3.4.2 Overall Pipeline Schedule

Fig. 26 summarizes the overall pipeline schedule of the design. PD and FD work in parallel for the luma, with specific scanning orders. As shown in Fig.26, PD pre-processes the MB *N+2* with the standard zig-zag order, while FD processes MBs *N* and *N+1* with the proposed IBR. Operations are performed at the block level and each block takes eight cycles. For chroma MD, prediction modes are processed in the specific order of vertical, horizontal and DC. Each chroma mode takes eight cycles.

REC processes the luma and chroma parts in an interlaced manner. The processing time for an 8x8 block of luma and a 4x8 segment of chroma varies depending on the MB-partition and whether the best mode is in the first two FD modes (denoted as hit or miss). It takes eight cycles if the block is from 16x16 partition or the hit-block from 8x8 partition. In the worst case, additional four cycles are required for two sequential blocks. During these additional REC time, the other three modules have to be idle. For example, when a miss happens on block 2, MB N in Fig. 26, it causes additional processing for REC and idle time for the others. In this way, the equivalent processing time for an MB is 32-40 cycles.

3.4.3 System Architecture

Fig. 27 shows the intra prediction architecture. The entire design can be divided into PD, FD, chroma MD and REC modules. With SAD cost function, PD module contains theprediction generator, difference, and cost computation units. Using DCT-based SATD, the additional multi-DCT unit is included in the FD module. HT is also implemented to process DC coefficients. FD module is utilized to decide the best mode and generate the coefficients. With a similar architecture to the FD, the chroma MD module is implemented to perform the MD operation for the chroma part. The REC module is composed of configurable Q/IQ [16], multi-IDCT, IHT, prediction generator and the adder. It outputs the quantized values for entropy

coding and reconstructs the pixels for PD, FD and chroma MD modules.

In the design, the main buffers are indicated as gray blocks in Fig. 27. They are all implemented with registers. This is because the processing speed can be as fast as 32-40 cycels/MB, which requires very high throughput for the buffers. The register buffers are categorized into 4 classes and described as follows: 1) source buffer stores the pixels of three MBs. Compared with the original storage of an MB, IBR introduces an additional MB due to the interlaced operation for two consecutive MBs. Moreover, PD is a pre-processing operation which needs to process another MB prior. 2) assigner buffers in the FD, PD and Chroma MD modules store the original pixels of the blocks that are currently being processed and the boundary pixels that are referenced; 3) coefficient buffers in the FD and chroma MD modules store the coefficients of the transformed blocks being currently processed. The storage for two modes is needed, including the best and the temporary modes. With these buffers, computation for the best mode is saved after it is decided; 4) the buffers store quantized coefficients and mode information as the output of this design. It should be mentioned here, for the source buffer, it takes less silicon area if we implement it with the SRAM (64 length, 80 bits width, 2 banks), which can be a reasonable design choice.

In addition, a buffer is utilized to store the referenced reconstructed pixels and mode information of the upper MB-row. In this design, it is indicated as the upper-line buffer in Fig. 27 and implemented with 16.3k bytes (7680x17bit) single port on-chip memory. The buffer size is proportional to the width of picture.

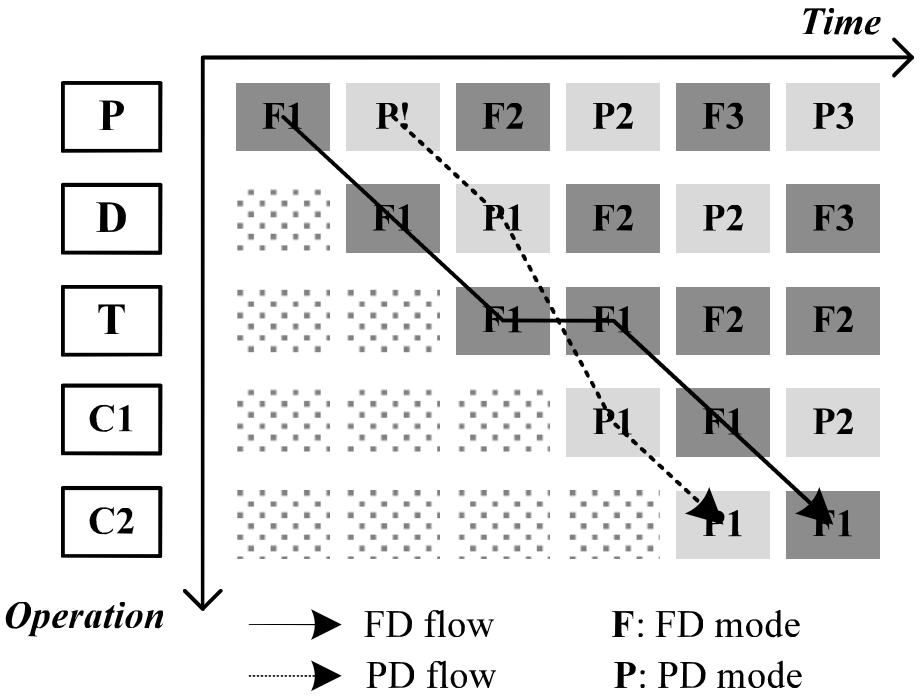


Fig.28. Timing chart of FD module sharing for FD and PD modes

3.4.4 FD Module Sharing

In this design, the FD transform unit can be utilized for eight cycles during the operation time for each block, while the other units of FD are only utilized for four cycles. It is observed that PD and FD operations are the same except for the transform, which means these units can also be used for PD processing. To improve the FD utilization and save the PD cost, FD units including prediction generator, difference and cost computation are shared by the PD and FD processing. Fig. 28 illustrates the timing chart of FD units. As shown, when the transform unit processes each FD mode within two cycles, the other units process FD and PD modes in an interlaced manner. By doing so, for each block, four PD modes can be inserted into the idle bubbles of the FD pipeline. Thus, the parallelism of the PD module is reduced from 96 to 64 pixels. Meanwhile, the utilization of these shared FD units is improved as the same as the transform unit.

3.4.5 PE-reusable Prediction Generator

Several designs for the prediction generator have been proposed [16]-[23][25][28][29]. The architecture in [25] is shown in Fig. 29. It is efficient for high pixel-parallelism as it greatly reuses the partial sums. The hardware cost of this architecture depends on the number of the processing elements (PEs). In the conventional method, the design adopts the same number of PEs as the parallelism of pixels [25].

In our design, after applying the FDMS, the pixel- parallelisms of prediction generators in PD and FD are 64. The 8x8 block is predicted with one cycle. By analyzing the prediction, it is observed that the predictive values in the direction are equivalent. Fig. 30 shows examples of modes 4 and 5. Among all the modes, at most 22 different values exist in a predictive block. With this characteristic, 22 PEs are used for the prediction generators of PD and FD modules.

In addition, we further reduce the number of PEs for the PD prediction generator, by utilizing the characteristic of the four PD modes (the horizontal and vertical ones in both 8x8 and 16x16 modes). The four modes do not need any operation but bypass the referenced pixels directly. They are arranged to be processed in an interlaced manner with the four non-bypass modes in every eight cycles. By doing so, it takes two cycles to process a non-bypass mode for a block, and thus the number of PEs for the PD prediction generator is reduced from 22 to 11.

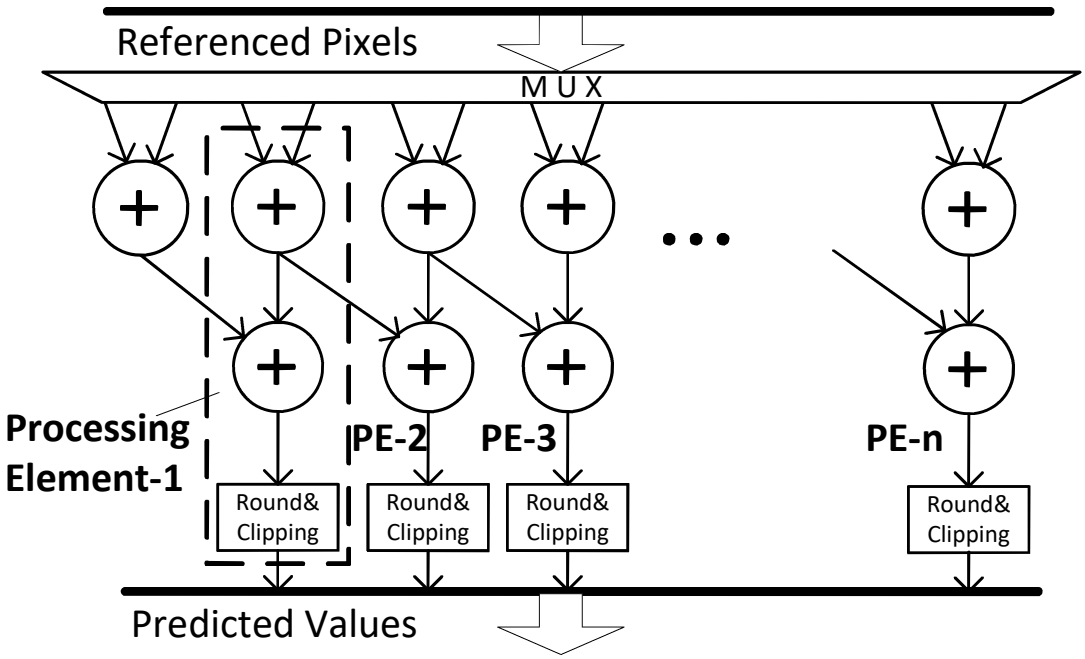


Fig.29. Architecture of prediction generator.

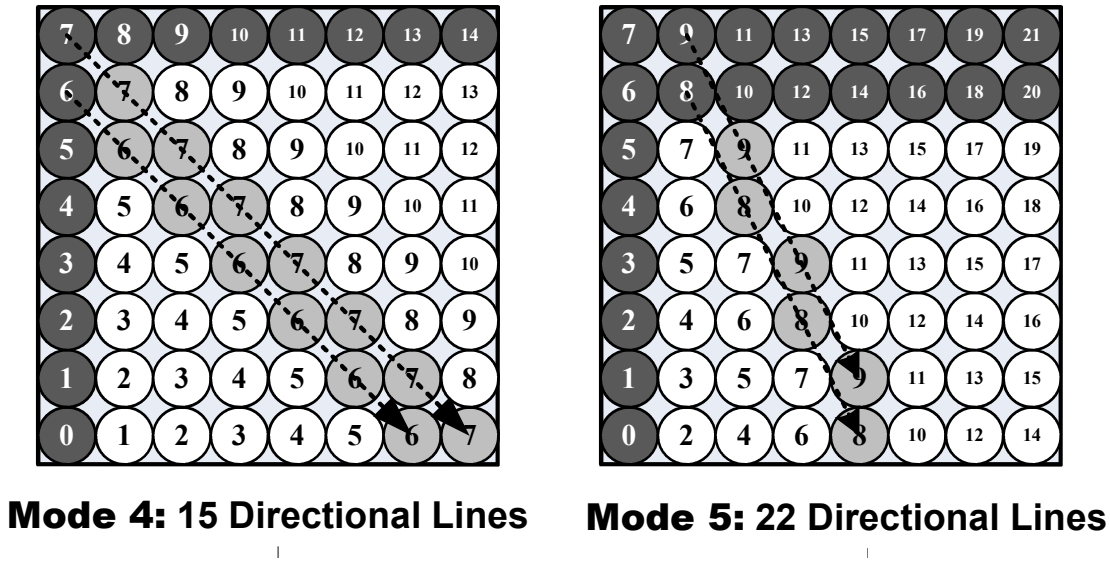


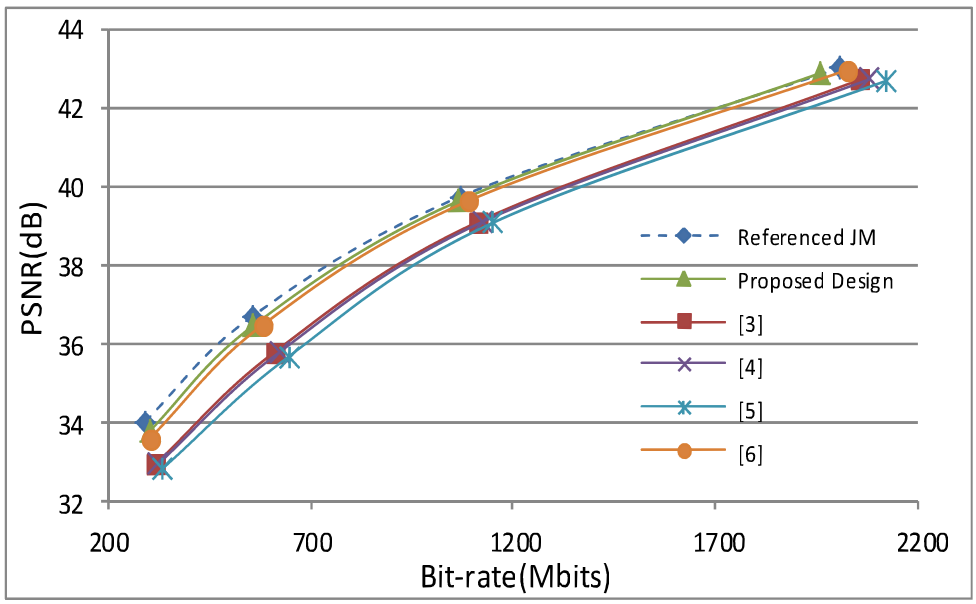
Fig.30. Illustration of equivalent values on predictive direction

3.5 Experimental results

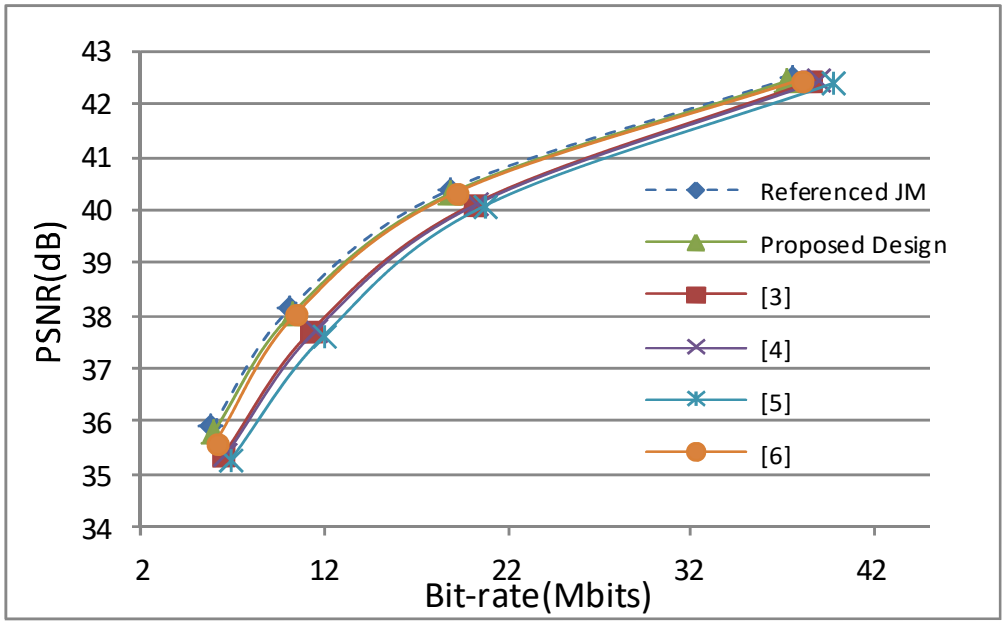
3.5.1 The implementation result

The proposed intra prediction architecture is applied in an H.264/AVC intra-frame encoder, which is implemented as a chip [35]. The chip implementation uses e-shuttle 65nm 1P12M CMOS technology. Fig. 21 shows the chip layout and the detailed distribution of intra modules. The specifications of the intra design are given in Table VI. The supported modes include all nine 8x8 prediction modes and three 16x16 modes (vertical, horizontal, DC). Table VII lists area breakdown of the design in logic gate count and on-chip memory. The logic gate count is 451.5k in total. In our design, it takes an average of 33 cycles to process one MB and 7680x4320 60fps real-time encoding could be achieved at 273 MHz. The maximum operating frequency is 300MHz, which is good enough for real-time 4320p60 encoding.

Table VIII lists the coding efficiency performance of the proposed design, compared with software JM17.0 [37]. The referenced JM supports 8x8 and 16x16 modes with plane mode removed. BD-bitrate and BD-psnr are used to estimate the average PSNR and bit-rate difference. Results are calculated under 4 different QPs (22, 27, 32, 37). The overall PSNR degradation of proposed design is about 0.07 dB or an approximately 1.86% bit-rate increasing. PMD scheme takes the major part about 0.06 dB. Considering the contribution for both complexity reduction and the cooperation with other schemes, it is acceptable. Fig. 22 shows the rate-distortion (RD) curves of the referenced JM, proposed design and references [16]-[19] for the 4320p video “Nebuta2” and 1080p video “Station”. The curves of proposed design are very close to the referenced JM and similar to [19] which supports the 4x4, 8x8 and 16x16 modes. Besides, they are better than previous designs [16]-[18], as they do not support the 8x8 prediction mode, which provides better coding efficiency for high resolution applications.



(a) 4320p Nebuta2



(b) 1080p Station

Fig.31. Rate-distortion curves of the referenced JM, proposed design and references [16]-[19] for (a) 4320p Nebuta2 and (b) 1080p Station.

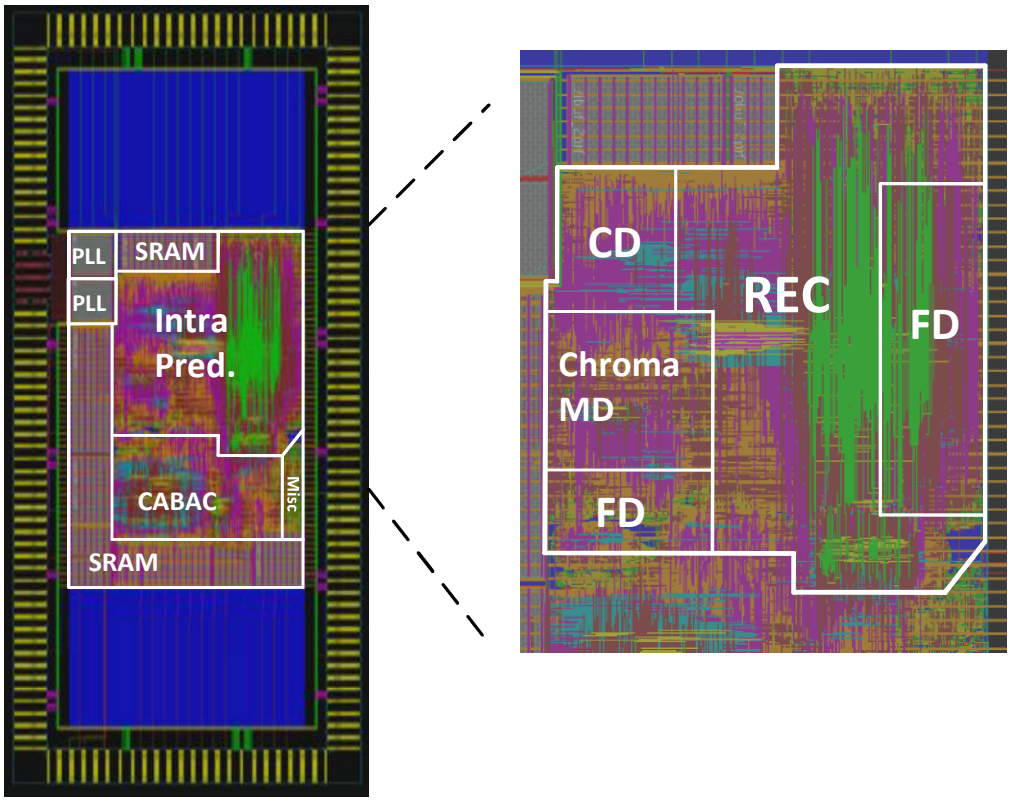


Fig.32. Chip layout.

Table 11 The Coding Efficiency Comparison with JM17.0* under QPs (22, 27, 32, 37)

Videos		PMD		PMD+IBR	
		BD-bitrate (%)	BD-psnr (dB)	BD-bitrate (%)	BD-psnr (dB)
4320p	Nebuta1	1.23	-0.07	1.85	-0.10
	Nebuta2	1.58	-0.08	2.03	-0.10
	Train1	0.81	-0.03	1.36	-0.05
	Train2	1.03	-0.04	1.52	-0.06
2160p	CrowdRun	1.49	-0.06	1.94	-0.08
	DucksTakeOff	1.37	-0.04	1.41	-0.04
	Splash	1.87	-0.05	2.08	-0.06
	ParkJoy	1.03	-0.04	1.31	-0.05
1080p	Pedestrian	2.09	-0.07	2.5	-0.08
	Station	1.56	-0.05	1.78	-0.06
	BlueSky	1.17	-0.05	1.39	-0.06
	Tractor	2.34	-0.11	2.46	-0.12
Average		1.46	-0.06	**1.80**	**-0.07**

*: JM17.0 supporting 8x8 and 16x16 modes with plane mode removed.

Table 12 Specifications of Intra Prediction Design

Technology	e-Shuttle 65nm 1P12M CMOS
Coding tools	nine 8x8 and three 16x16 prediction modes
Logic gate count	451.5K (2-input NAND gate)
Max. frequency	300 MHz
Max. resolution	7680x4320p 60fps
Max. pixel rate	1991 Mpixels/s

Table 13 Area Breakdown in Logic Gate Count and On-chip Memory

Module	Logic gate count	On-chip memory
Preliminary decision	40 285	0
Fine decision	141 063	0
Chroma mode decision	85 321	0
Upper-line buffer	1 272	16.3KB (7680x17bit) single port
Reconstruction	183 576	0
Total	451 517	0

Fig. 33 summarizes the optimization results with the proposed techniques. According to the analysis in Section IV.A, the design is optimized in terms of area and frequency. By applying PMD-f1, the hardware cost is first reduced by 15%. The PMD-f2, IBR, and PBR are then used to reduce the required frequency by 28%, 50%, and 18%, respectively. In addition, hardware reuse (HR) strategies including FDMS and PE-reusable prediction generator are applied. These two strategies

contribute to a 10% reduction in overall area. As a result, the hardware complexity in terms of the product of area and frequency is reduced by 77%.

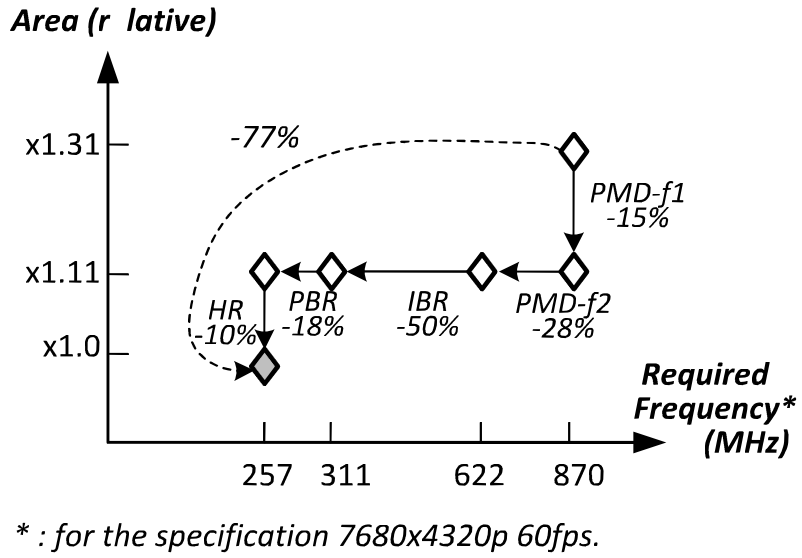


Fig.33. Complexity reduction with adopted techniques.

Table IX shows the comparison between this work and the state-of-the-art. The proposed design delivers a maximum throughput of 1991Mpixels/s for real-time 7680x4320 60fps encoding at 273MHz, which is 13.2 to 32 times faster than previous designs [16]-[19]. The throughput improvement comes from both the high parallelism and hardware efficiency. Moreover, the 1080p 30fps encoding requires less than 9MHz operating frequency, which is much lower than that of previous works.

3.5.2 The difference between two proposed reordering methods

With different reordering schedule and other schemes, this design avoids the buffer system of MB-rows in chapter 2 and improves area efficiency. In chapter 2, a block/MB reordering in MB-rows is proposed. While entropy coding (CABAC) cannot be modified like that, a DRAM buffer system is needed to reorder syntax elements. For 8kx4k UHD, the caused DRAM bandwidth is very huge. Thus,

compression & decompression operations are needed, which leads to additional area and power consumption. Besides, the bandwidth problem still cannot be solved completely.

3.5.3 Adaptation on HEVC standard

Recently, a new video coding standard is announced as High Efficiency Video Coding (HEVC), which is developed by Joint Collaborative Team on Video Coding (JCT-VC). It almost achieves double compression performance of H.264. During the intra coding process, two new coding tools are involved. One is so called quard-tree block coding which allows the intra prediction is processed based on the blocks from 64x64 to 4x4. The other is the highly increased numbers of the prediction modes including 34 angular and 1 DC. Compared with H.264, due to these new coding tools, the complexity of intra prediction becomes much higher [44].

With some modification, proposed algorithms and architectures can also be utilized to reduce the complexity and improve the pipeline utilization. Firstly, more complex direction modes needs to be pre-processing, which should be suitable for our proposed preliminary mode decision (PMD). Using the PMD, a certain number of candidates modes can be selected. Moreover, since it refers to the original pixels instead of reconstructed ones to do the prediction, it doesn't involve any data dependency problem. Secondly, there are 5 variable block partitions in HEVC. The data dependency problem becomes more serious. A dedicated reordering strategy should be arranged to solve the problem. Since CTU-rows the parallel processing of CTU-rows are support in HEVC CABAC, the reordering involve the blocks not only within a CTU, but also the CTU-rows. Based on that, an efficient pipeline can be designed.

3.6 Conclusion

This chapter discusses the design of an intra prediction architecture for H.264/AVC encoder in 65 nm CMOS. An IBR scheme together with PMD allows parallel operations of MD and REC and also reduces the complexity. Meanwhile, a PBR scheme solves the problem from pipeline latency. In addition, HR strategies are applied to further reduce the cost. With these adopted techniques, the overall hardware complexity in terms of area and frequency is reduced by 77%. Its maximum throughput reaches 1991Mpixels/s for real-time 7680x4320p 60fps encoding at 273MHz. The 1080p 30fps encoding requires less than 9MHz operating frequency, which is much lower than that of previous works.

4. An HEVC Fractional Motion Estimation Architecture for 8kx4k UHD

4.1 Introduction

Recently, the video compression standard called high efficiency video coding (HEVC/H.265), has been approved by Joint Collaborative Team on Video Coding (JCT-VC) [38]. It doubles the data compression ratio compared to the precursor of AVC/H.264. Besides, high resolutions such as ultra-high definition (Ultra-HD 7680x4320, 3840x2160) are supported for next-generation applications.

As an important component of video encoding, fractional motion estimation (FME) provides the sub-pixel refinement. It improves the rate-distortion performance significantly about 3-6dB, but results in high computation complexity (40% encoding time) due to complex interpolation and fractional search process. Previous works [39]-[42] have proposed efficient designs in H.264. Due to the new features in HEVC, most of them cannot be applied directly. Quad-tree structure enables the motion estimation in coding blocks from 64x64 to 8x8. Moreover, the interpolation is processed with complex 7/8-tap filters. In addition, the transform coding has been added as residual quad-tree (RQT).

In this work, an efficient FME architecture in HEVC is proposed. The design targets Ultra-HD 7680x4320 30fps real-time encoding. The co-optimization of algorithm and hardware are characterized as follows. (1) By using bilinear quarter pixel approximation, we reduce 76% interpolation complexity and save transform operation for quarter candidates. (2) A 5T12S search pattern is proposed to achieve a tradeoff between hardware cost and coding quality. 48% hardware cost is reduced with negligible quality loss, compared with conventional 9T25S. (3) Exhaustive size-hadamard transform (ES-HAD) is adopted in FME. It avoids unifying all blocks into small transform ones. Furthermore, it determines the best transform size,

rather than using the complex RQT. Besides, data reusing in ES-HAD is exploited and 58% hardware cost is reduced, compared with the straightforward implementation.

The rest of chapter is organized as follows. Section II reviews FME algorithm in HEVC referenced software. In section III, we present our proposed algorithm. Hardware architecture is described in Section IV. Implementation results are shown in Section V. Section VI concludes the chapter.

4.2 FME Algorithm In Referenced Software

In HEVC, an input picture is divided into coding tree blocks (CTBs, typically 64x64) and each CTB can be further divided into coding blocks (CBs, limited to 8x8) through a recursive quad-tree structure. The three symmetric and four asymmetric prediction modes are supported for each CB. For each prediction mode, the procedure of motion estimation is classified into three steps. Firstly, integer motion estimation (IME) performs motion search to find integer motion vectors (IMVs). Next, FME provides the sub-pixel refinements around IMVs. At last, RQT processes the transform coding. The transform sizes are allowed in the range of 4x4 to 32x32 and the best one is decided with rate distortion optimization (RDO). The detailed FME are carried out as follows: the fractional pixels (pels) with quarter accuracy are first interpolated with 7/8-tap filter. Then, two refinements from half pixel to quarter pixel are performed sequentially. In each refinement, nine neighboring points around the former best result are searched. The cost function adopted in FME is the hadamard transform absolute difference (HAD). It includes the operations of difference generation (DG), 4x4/8x8 hadamard transform (HT), absolute value and cost accumulation.

4.3 Proposed FME Algorithm

4.3.1 Bilinear Quarter pixel Approximation

In HEVC, fractional pixels are classified into a, b and c types and performed with corresponding independent 7/8/7-tap filters. This is defined for motion compensation (MC). If we apply this into motion estimation, it introduces a high complexity interpolation. Moreover, relative to H.264, the operation linearity cannot be utilized and hardware cost increases largely. In H.264, the interpolation adopts a half-then- quarter model. It first generates half pixels with 6-tap filter, and then uses half pixels to generate quarter ones with a 2-tap bilinear filter. Due to the linearity of DG and HT, the operation results of quarter pixels can be directly calculated from the results of half pixels, with the bilinear filter.

(a)

Intp.	Filter(tap)	Add.s	Num.(per int.)	Add.s(per int.)
a/b/c types	7/8/7	12/9/12	5/5/5	165
half/quarter	8/2	8/1	3/12	39

(b)

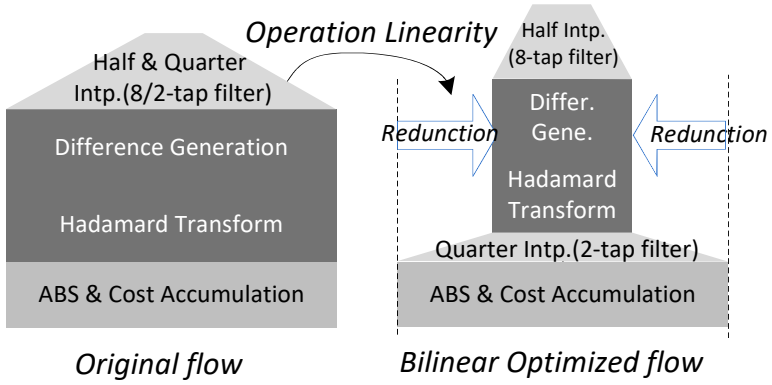


Fig.34. (a)Additions reduction (b) Saving DG and HT for quarter pixels.

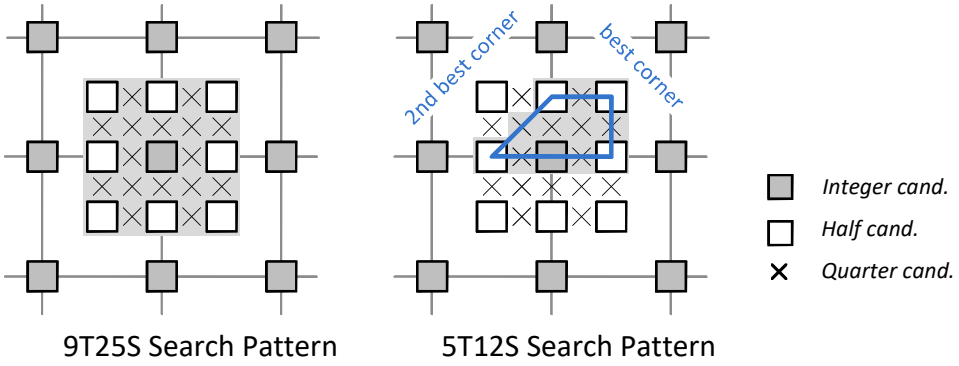


Fig.35. (a)conventional 9T25S (b) proposed 5T12S search pattern.

In this design, we propose a bilinear quarter pixel approximation (BQA) scheme. It employs the half-then-quarter model and uses bilinear filter for quarter pixels. The 8-tap filter is still adopted for half pixels to maintain the coding quality. As shown in Fig. 34 (a), the interpolation complexity is reduced about 76%, from 165 to 39 additions. The PSNR is degraded only 0.02dB. Moreover, FME process flow is optimized with the operation linearity, as shown in Fig. 34 (b). We calculate the coefficients of quarter pixels with bilinear 2-tap filter, and the DG and HT operations are saved. It is noted MC is not included here and needs to be handled separately.

4.3.2 5T12S Search Pattern

The search pattern in referenced software takes two iterations, and introduces long latency. One iteration search is suitable and widely used for high throughput hardware design. With one iteration, most existing works focus on the reduction of search candidates (SCs), since they generally determine the hardware cost. However, with proposed bilinear optimized flow shown in Fig. 34 (b), hardware cost is largely dependent upon the number of transformed candidates (TCs), since the transform takes most of the complexity.

The conventional search area of central 5x5 provides high coding quality [42], shown in Fig. 35 (a). Using the bilinear optimized flow, nine candidates are

processed with transform among the total 25 ones. It is denoted as 9T25S. To further

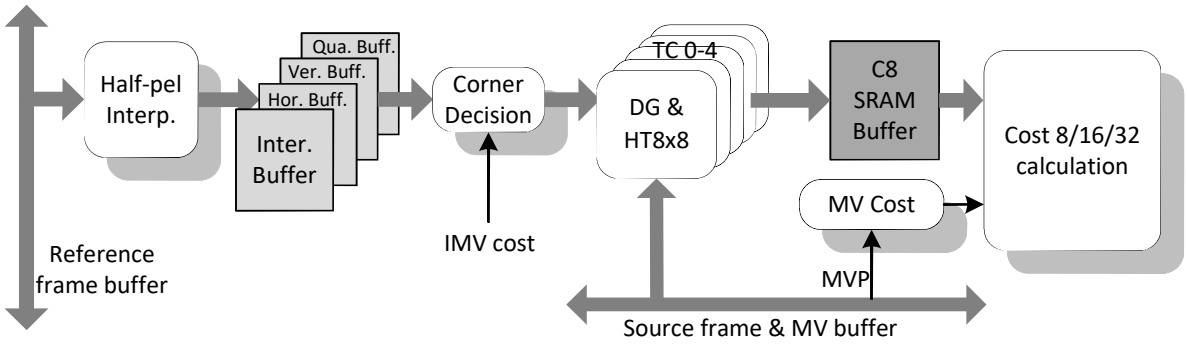


Fig.36. Block diagram of the FME design

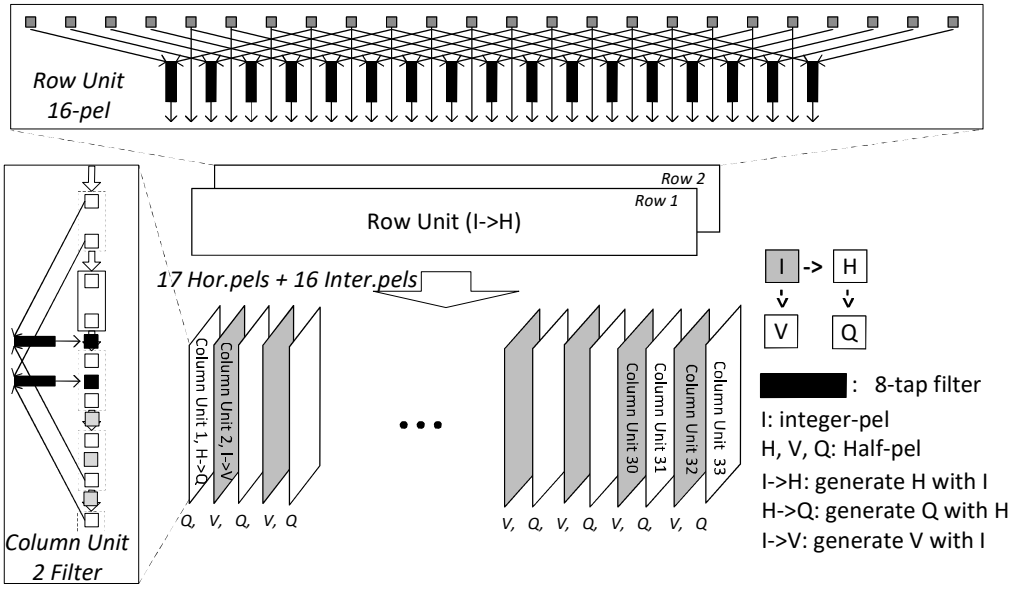


Fig.37. Block diagram of half-pel interpolation module.

reduce the hardware cost, we propose a 5T12S search pattern. Integer evaluation results are utilized to predict the high possibility area (best corners) for FME. Saving fewer transform candidates and maintaining the coding quality, the search pattern is decided as shown in Fig. 35 (b). Five TCs are processed to generate the results of 12 SCs, denoted as 5T12S. Relative to the 9T25S, it reduces 48% hardware cost with only 0.02dB PSNR degradation.

4.3.3 Exhaustive Size-HAD

In HEVC, the residual quadtree (RQT) allows transform sizes in the range of 4x4 to 32x32. The approach enables the adaptation of transform to the varying space-frequency characteristics of the residual. However, the adopted HAD as rate distortion estimation in FME, is limited to 4x4/8x8 transform sizes.

In this design, we apply an exhaustive size HAD (ES-HAD). The residuals are processed by HT8x8, HT16x16, and HT32x32, and then compared recursively to get rate distortion cost. The technique avoids unifying variable blocks into small size blocks and improves the encoding performance for FME, up to 0.1dB video quality. Moreover, it determines the best transform size, with much lower complexity, relative to the complex RDO adopted in RQT. The average PSNR drop is about 0.05dB. In addition, we exploit data reusable features in ES-HAD and reduce hardware cost largely, which is explained in Section 4.2.

4.4 Hardware Architecture

Fig. 36 shows the block diagram of proposed FME design. With referenced integer pixels, half fractional pixels are generated by half-pel interpolation module, and then stored in the buffers. After the corner decision, five TCs are processed parallel by five DG&HT8x8 units. The coefficients of HT8x8 (C8) are stored in a SRAM buffer, and reordered to be accessed for the reuse operations of HT16x16 and 32x32. At last, all 12 SCs with ES-HAD costs are generated and compared in cost calculation module.

$$H_{2x}=\begin{pmatrix} H_x & H_x \\ H_x & -H_x \end{pmatrix} \qquad A=\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

$$\exists\ T_{x-ij}=H_x \bullet A_{ij} \bullet H_x^{T} \qquad T_{2x}=H_{2x} \bullet A \bullet H_{2x}^{T}:$$

$$T_{2x}=\begin{pmatrix} T_{x-11}+T_{x-12}+T_{x-21}+T_{x-22} & T_{x-11}-T_{x-12}+T_{x-21}-T_{x-22} \\ T_{x-11}+T_{x-12}-T_{x-21}-T_{x-22} & T_{x-11}-T_{x-12}-T_{x-21}+T_{x-22} \end{pmatrix} \quad (1)$$

Table 14 comparison for Individual implementation and resue implementation

Addition Layer	HT8x8	HT16x16	HT32x32
Individual	6	8	10
Date Reuse	6	2	2

4.4.1 Parallelism

To design the FME architecture with high throughput and minimal hardware cost, the degree of parallelism has to be carefully considered. For the interpolation, an efficient way is to design the processing unit based on the smallest block. Others are decomposed to re-utilize the hardware. Based on the 16x8 block in this design, we use the horizontal unit with 16-pixel parallelism. It generates 17 half horizontal-pels with 8-tap filter and directly outputs 16 inter-pel. The corresponding 33 vertical units are needed to generate half vertical-pels and quarter-pels. However, it has to work at a high frequency 380MHz to achieve targeted throughput. Thus, we double the parallelism by adopting two horizontal units and embedding two 8-tap filters into a vertical unit, as shown in Fig. 37. By doing so, it takes 8 cycles to processes a 16x8 block. Other processing modules are designed based on 5T12S search pattern. The DG&HT8x8 units are duplicated five times and the cost calculation module contains 12 processing units. Being fully pipelined, all these units adopt 16-pixel parallelism, in accordance with the processing speed of interpolation (16x8 block/8cycles). It should be noted, in order to achieve high throughput, we support three selected modes from CB 64x64 to 16x16 by using IME evaluation results, with 0.07dB quality loss. A CTB (64x64) can be decomposed into 32 16x8 blocks. Thus we calculate the total required cycle count per CTB as 3 x 32 x 8 = 768 cycles.

4.4.2 Data Reuse in ES-HAD

In this design, ES-HAD is applied, where the residuals are processed with HT 8x8, 16x16 and 32x32. As shown in Table I, with individual implementation, six, eight, ten addition (ADD) layers are needed for pipeline design, respectively. Due to the recursive feature of HT, we exploit the reusable relationship between different sizes. As shown in the derivation (1), the transform calculation for T2x can reuse the transformed result Tx. Only two bufferfly ADD layers are needed. Based on that, we only implement the architecture of HT8x8. Two and two

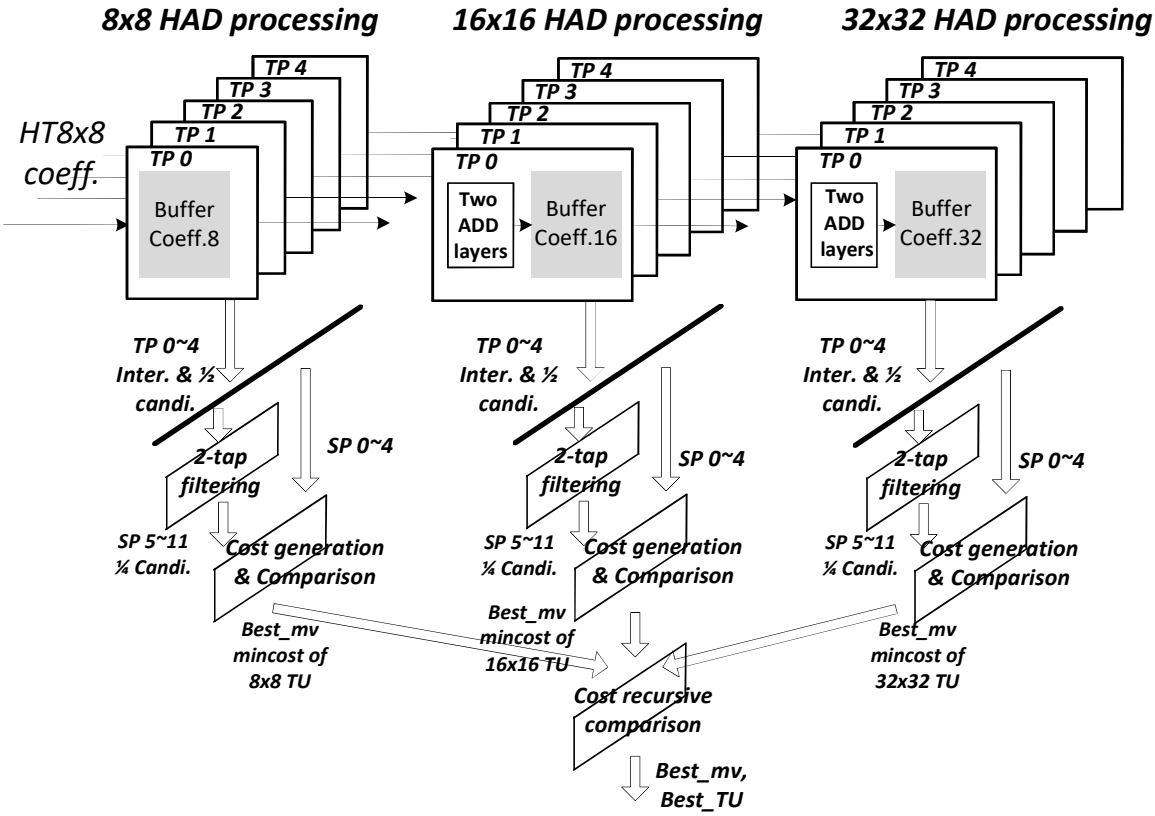


Fig.38. Block diagram of the cost calculation module.

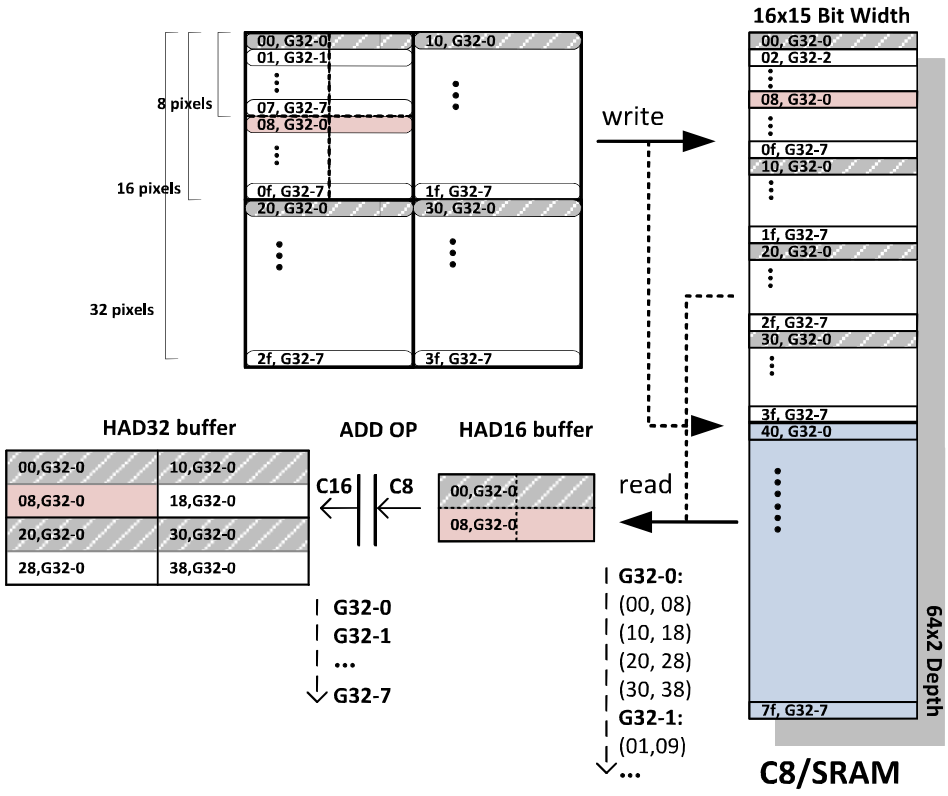


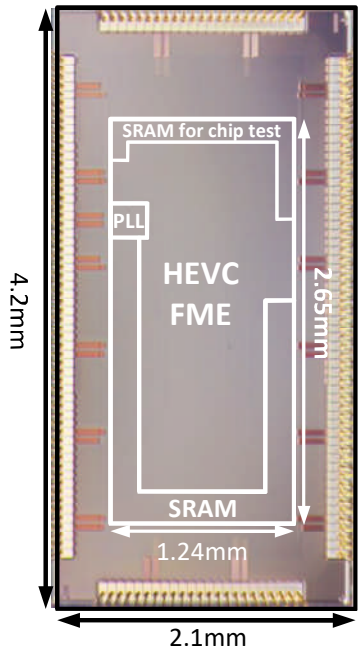
Fig.39. Organization and access pattern of the on-chip SRAM.

ADD layers are used to calculate HT16x16 and 32x32. Comparing with individual implementation, 58% hardware cost is reduced. Fig. 38 shows the block diagram of proposed cost calculation unit. It contains the processing for 8x8, 16x16, and 32x2 HAD. The coefficients are first used to calculate the RD cost, and also utilized to generate the coefficients of larger transform.

4.4.3 Memory Organization

According to derivation (1), ADD operation for a larger HT needs the transform results of four sub-blocks. In this design, HT8x8 adopts 16-pixel parallelism and processes a 16x16 block row by row. We employ a two port SRAM to store the C8. Being accessed with a dedicated order, they are used to calculate the larger HTs in a pipelining manner. Word size of SRAM is 240 bits (16-pel, 15 bits per pel). Two 32x32 blocks are stored with 128 depth. It is written and read in a ping-pong way. Fig. 39 illustrates the organization and access pattern. The data

needs to read every eight and 16 addresses to perform the



Technology	E-Shuttle 65nm CMOS
I/O pads	162
Die area	8.82mm^2
Core area	3.29mm^2
Density	81%
Logic Gate Count	1183k (2-input NAND)
SRAM	19.2KB/5.1KB*
Max. resolution	7680x4320 30fps (10bits/pel)
Max. pixel rate	995Mpixels/s

Measured core power

198.6mW for 4320p30 @1.20v, 188MHz
48.3mW for 2160p60 @0.70v, 95MHz
25.9mW for 2160p30 @0.70v, 48MHz
6.3mW for 1080p30 @0.70v, 12MHz

*: 5.1KB SRAM for chip test

Fig.40. Chip specification and micrograph.

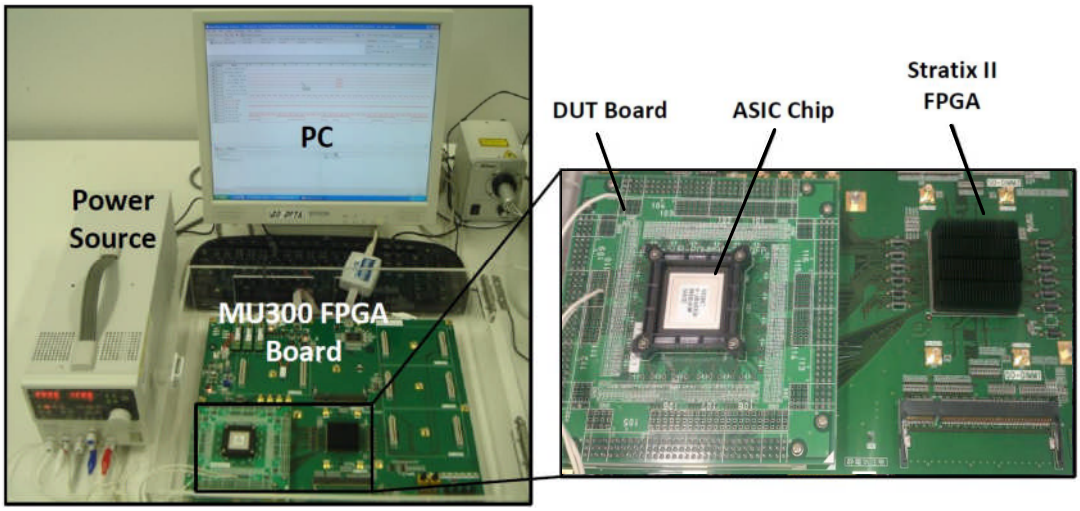


Fig.41. Photos of verification system.

ADD operations for HT16x16 and 32xx32. Thus, we classify a 32x32 block into 8 groups (G32) by every eight addresses and access the data of groups as the ascending order. By doing so, the HAD16 and HAD32 processing are fully pipelined.

4.5 Implementation Result

To verify the proposed architecture, it is implemented with 65nm 1P12M LVT CMOS. Fig. 40 shows the die photo and specification of the chip. 1183K logic gates, 24.3KB SRAM (including 5.1KB for test) and a PLL are integrated into a 3.29mm2 core. The chip is verified by FPGA based evaluation system, as shown in Fig. 41. At 1.2V supply and 188MHz frequency, the design achieves 7680x4320 30fps real-time encoding. The corresponding power dissipation is 198.6mW with 199.4pJ/pixel energy efficiency. At 0.7v, the 1080p30 processing dissipates only 6.3mW when running at 12MHz. The best energy efficiency is 97.0pJ/pixel as measured at 0.7v and 95MHz. Fig. 42 shows the RD performance curves of the design and referenced software HM10.0 for 2160p video “Nebuta”. With much lower complexity, our design delivers similar coding efficiency with referenced software.

The comparison with the state-of-art designs on specification and performance are concluded in Fig. 43. With algorithm and architecture co-optimization, this design reaches 995Mpixels/s high throughput for 7680x4320 30fps video, at least 4.7 times faster than previous designs. Despite high complexity in HEVC, our chip achieves much better power efficiency (0.2nJ/pixel) than previous works in H.264, at least 56% improvement, even considering the technique scaling.

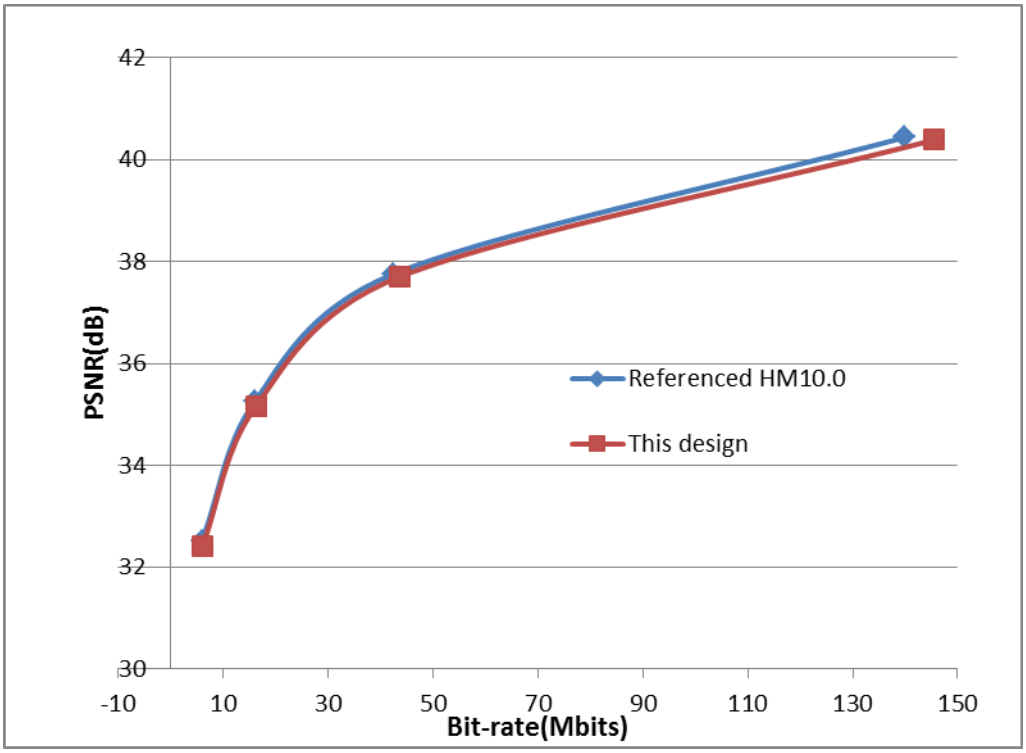


Fig.42. Rate-distortion curves of referenced HM10.0 and proposed design.

	This Work	TVLSI'10[2]	ISSCC'09[3]/ICME'09[4]
Max. Resolution	**7680x4320@30fps**	1920x1080@30fps	4096x2160@24fps
Max. Throughput	**995Mpixels/s**	62.2Mpixels/s	212Mpixels/s
Standard	**HEVC(H.265)**	H.264	H.264
Transform	**HT8x8/16x16/32x32**	HT4x4	HT4x4/8x8
Algorithm	**8/2-tap & 5T12S**	6/2-tap & 9T9Sx2	2-tap & 9T49S
Technology/Supply	**65nm/1.2V**	0.18um/1.2V	90nm/1.2V
Logic gates/SRAM	**1183K/19.2KB**	321K/9.72KB	448K/-
Cycles/pixel[1)]	**0.19**	2.46	1.32
FME power	**198.6mW**	374mW	135.02mW[2)]
Norm. FME power[3)]	**198.6mW**	135mW	97.49mW
Power efficiency	**0.2nJ/pixel**	2.17nJ/pixel	0.46nJ/pixel

1): The speed normalized to the processing cycles for each pixel.
2): FME power calculated from encoder core power and FME area proportion.
3): FME power normalized to 65nm/1.2V. (P65nm/1.2V = P90nm/1.2V /1.385 = P0.18μm/1.2V / 2.77)

Fig.43. Design comparison with the state-of-art designs.

4.6 Conclusion

In this chapter, we propose an HEVC FME architecture. A BQA scheme, together with a 5T12S search pattern is proposed to reduce the complexity. ES-HAD is proposed to improve coding quality of FME and determine transform size without RQT. Furthermore, a data reuse method of ES-HAD is applied to reduce hardware cost. The implemented chip achieves 7680x4320 30fps real-time encoding, at 1.2V supply and 188MHz frequency. Despite high complexity in HEVC, the design achieves at least 56% improvement on power efficiency (0.2nJ/pixel) than previous works in H.264.

5. Conclusion of the dissertation

In this dissertation, the efficient architectures of intra prediction and FME in H.264 or HEVC targeting for UHD videos are proposed. Based on two reordering strategies, the data dependency problem of intra prediction is solved. Other architecture optimization such as hardware reusing and module sharing are also proposed. The 4kx2k and 8kx4k intra prediction architecture are designed. In addition, a high-performance and low-power HEVC FME architecture is designed, with the co-optimization in algorithm and hardware architecture.

Firstly, we propose a novel intra prediction hardware architecture that can support H.264/AVC 4096x2160@60fps real-time encoding. MB/block co-reordering process is proposed to alleviate data dependency and make the pipeline more efficient with negligible quality loss. Moreover, PE-reusable 8x8 intra predictor, hybrid SAD & SATD mode decision, interlaced reconstruction and other scheduling techniques are applied to improve hardware utilization and save hardware cost. Only 160 cycles are used to process one MB. Compared with previous designs of 1080p@30fps, throughput of proposed design increases for 8 times with small area overhead.

Secondly, the design of an intra prediction architecture for H.264/AVC encoder in 65 nm CMOS is proposed. An IBR scheme together with PMD allows parallel operations of MD and REC and also reduces the complexity. Meanwhile, a PBR scheme solves the problem from pipeline latency. In addition, HR strategies are applied to further reduce the cost. With these adopted techniques, the overall hardware complexity in terms of area and frequency is reduced by 77%. Its maximum throughput reaches 1991Mpixels/s for real-time 7680x4320p 60fps encoding at 273MHz. The 1080p 30fps encoding requires less than 9MHz operating frequency, which is much lower than that of previous works

Thirdly, we propose an HEVC FME architecture. A BQA scheme, together with a 5T12S search pattern is proposed to reduce the complexity. ES-HAD is proposed

to improve coding quality of FME and determine transform size without RQT. Furthermore, a data reuse method of ES-HAD is applied to reduce hardware cost. The implemented chip achieves 7680x4320 30fps real-time encoding, at 1.2V supply and 188MHz frequency. Despite high complexity in HEVC, the design achieves at least 56% improvement on power efficiency (0.2nJ/pixel) than previous works in H.264.

In conclusion, the proposed intra prediction and FME architectures achieves the throughput of UHD real-time application. They have been also proved as the efficient designs with high hardware utilization.

Reference

[1] Joint Video Team, “Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264|ISO/IEC 14496-10 AVC),” JVT-G050, May 2003.

[2] Li-Fu Ding, Wen-Yin Chen, Pei-Kuei Tsung, Tzu-Der Chuang, Pai-Heng Hsiao, Yu-Han Chen, Hsu-Kuang Chiu, Shao-Yi Chien and Liang-Gee Chen, “A 212Mpixels/s 4096x2160p Multiview Video Encoder Chip for 3D/Quad Full HDTV Applications,” IEEE Journal of Solid-State Circuits, vol. 45, no. 1, pp. 46-58, 2010.

[3] Yu-Kun Lin, Chun-Wei Ku, De-Wei Li, and Tian-Sheuan Chang, “A 140-MHz 94K Gates HD1080p 30-Frames/s Intra-Only Profile H.264 Encoder,” IEEE Trans. Circuits Syst. Video Technol., vol. 19, no. 3, pp.432-436, 2009.

[4] J Huang-Chih Kuo and Youn-Long Lin, “An H.264/AVC full-mode intra-frame encoder for 1080HD video,” IEEE International Conference on Multimedia and Expo, pp.1037-1046, June 2008.

[5] Tzu-Der Chuang, Yi-Hau Chen, Chen-Han Tsai, Yu-Jen Chen, and Liang-Gee Chen, “Algorithm and architecture design for intra prediction in H.264/AVC High profile,” Picture Coding Symposium, 2007.

[6] Ching-Yeh Chen, Chao-Tsung Huang, Yi-Hau Chen, and Liang-Gee Chen, “Level C+ Data Reuse Scheme for Motion Estimation With Corresponding Coding Orders,” IEEE Trans. Circuits Syst. Video Technol., vol. 16, no. 4, 2006.

[7] Dajiang Zhou, Jinjia Zhou, Xun He, Ji Kong, Jiayi Zhu, Peilin Liu, and Satoshi Goto, “A 530Mpixels/s 4096x2160@60fps H.264/AVC high profile video decoder chip,” Symposium on VLSI Circuits, pp.171-172, June 2010.

[8] Seiji Mochizuki, Tetsuya Shibayama, Masaru Hase, Fumitaka Izuhara, Kazushi

Akie, Masaki Nobori, Ren Imaoka, Hiroshi Ueda, Kazuyuki Ishikawa, and Hiromi Watanabe, “A 64mW High Picture Quality H.264/MPEG-4 Video Codec IP for HD Mobile Applications in 90nm CMOS,” IEEE Journal of Solid-State Circuits, vol. 43, no. 11, pp. 2354-2362, 2008.

[9] Chun-Wei Ku, Chao-Chung Cheng, Guo-Shiuan Yu, Min-Chi Tsai, and Tian-Sheuan Chang, “A high-definition H.264/AVC intra-frame codec IP for digital video and still camera application,” IEEE Trans. Circuits Syst. Video Technol., vol. 16, no.8, pp. 917-928, August 2006.

[10] G. Bjontegaard, “Improvements of the BD-PSNR model,” ITU-T SC16/Q6, 35th VCEG Meeting, Berlin, Germany, Doc. VCEG-AI11, July 2008.

[11] Y.-W. Huang, B.-Y. Hsieh, T.-C. Chen, and L.-G. Chen, “Analysis,fast algorithm, and VLSI a .264/AVC intra frame coder,” IEEE Trans. Circuits Syst. Video Technol., vol. 15, no. 3, pp.378-401, March 2005.

[12] Yu-Kun Lin, De-Wei Li, Chia-Chun Lin, Tzu-Yun Kuo, Sian-Jin Wu, Wei-Cheng Tai, Wei-Cheng Chang, and Tian-Sheuan Chang, “A 242mW, 10mm 1080p H.264/AVC High Profile Encoder Chip,” IEEE International Solid-State Circuits Conference, pp. 314-615, 2008.

[13] http://iphome.hhi.de/suehring/tml/download/ H.264/AVC Reference Software JM17.0, 2010.

[14] T. Ito, “Future television-Super Hi-Vision and beyond,” in *Proc. IEEE Asian Solid-State Circuits Conf.,* pp. 1-4, November 2010.

[15] Joint Video Team, “Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264|ISO/IEC 14496-10 AVC),” JVT-G050, May 2003.

[16] H.-C. Kuo, L.-C. Wu, H.-T. Huang, S.-T. Hsu, and Y.-L. Lin, “A Low-Power

High-Performance H.264/AVC Intra-Frame Encoder for 1080pHD Video," *IEEE Trans. Very Large Scale Integr.(VLSI) Syst.*, vol. 19, no. 6, pp. 925-938, June 2011.

[17] Y.-K. Lin, C.-W. Ku, D.-W. Li, and T.-S. Chang, "A 140-MHz 94K Gates HD1080p 30-Frames/s Intra-Only Profile H.264 Encoder," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 19, no. 3, pp.432-436, March 2009.

[18] C. Diniz, B. Zatt, C. Thiele, A. Susin, S. Bampi, F. Sampaio, D. Palomino and L. Agostini, "A High Throughput H.264/AVC Intra-Frame Encoding Loop Architecture for HD1080p," in *Proc. IEEE Int. Symp. Circuits Syst.*, pp. 579-582, May 2011.

[19] G.-L. Li, T.-Y. Chen, M.-H. Wen and T.-S. Chang, "135-MHz 258-K Gates VLSI Design for All-Intra H.264/AVC Scalable Video Encoder," *IEEE Trans. Very Large Scale Integr.(VLSI) Syst*., vol. 21, no. 4, pp.636-647, April 2013.

[20] Y.-W. Huang, B.-Y. Hsieh, T.-C. Chen, and L.-G. Chen, "Analysis, Fast Algorithm, and VLSI Architecture Design for H.264/AVC Intra Frame Coder," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 15, no. 3, pp. 378-401, March 2005.

[21] K. Suh, S. Park, and H. Cho, "An efficient hardware architecture of intra prediction and TQ/IQIT module for H.264 encoder," *ETRI J.,* vol.27, no. 5, pp. 511-524, October 2005.

[22] C.-W. Ku, C.-C. Cheng, G.-S. Yu, M.-C. Tsai, and T.-S. Chang, "A High-Definition H.264/AVC Intra-Frame Codec IP for Digital Video and Still Camera Applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 8, pp. 917-928, August 2006.

[23] H.-Y. Lin, K.-H. Wu, B.-D. Liu, and J.-F. Yang, "An Efficient VLSI Architecture for Transform-Based Intra Prediction in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 6, pp. 894-906, June 2010.

[24] D.-W. Li, C.-W. Ku, C.-C. Cheng, Y.-K. Lin, and T.-S. Chang, "A 61 MHz 72K gates 1280 × 720 30 frames/s H.264 intra encoder," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 2, pp. 801–804, April 2007.

[25] L.-F. Ding, W.-Y. Chen, P.-K. Tsung, T.-D. Chuang, P.-H. Hsiao, Y.-H. Chen, H.-K. Chiu, S.-Y. Chien and L.-G. Chen, "A 212Mpixels/s 4096x2160p Multiview Video Encoder Chip for 3D/Quad Full HDTV Applications," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 1, pp. 46-58, February 2010.

[26] S. Mochizuki, T. Shibayama, M. Hase, F. Izuhara, K. Akie, M. Nobori, R. Imaoka, H. Ueda, K. Ishikawa, and H. Watanabe, "A 64mW High Picture Quality H.264/MPEG-4 Video Codec IP for HD Mobile Applications in 90nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 11, pp. 2354-2362, November 2008.

[27] G. Jin, J.-S. Jung and H.-J. Lee, "An Efficient Pipelined Architecture for H.264/AVC Intra Frame Processing", in *Proc. IEEE Int. Symp. Circuits Syst.*, pp. 1605-1608, May 2007.

[28] S.-B. Wang, X.-L. Zhang, Y. Yao and Z. Wang, "H.264 Intra Prediction Architecture Optimization", in *Proc. IEEE Intl. Conf. Multimedia and Expo,* pp. 1571-1574, July 2007.

[29] H. Ren, Y. Fan, X. Chen and X. Zeng, "A 16-pixel Parallel Architecture with Block-level/Mode-level Co-reordering Approach for Intra Prediction in 4kx2k H.264/AVC Video Encoder," in *Proc. Asia and South Pacific Design Auto. Conf.* (ASP-DAC), pp. 801-806, January 2012.

[30] Y.-W. Huang, T.-C. Chen, C.-H. Tsai, C.-Y. Chen, T.-W. Chen, C.-S. chen, C.-F. Shen, S.-Y. Ma, T.-C. Wang, B.-Y. Hsieh, H.-C. Fang and L.-G. Chen, "A 1.3 TOPS H.264/AVC single-chip encoder for HDTV applications," in *IEEE ISSCC Dig. Tech. Papers*, pp.128–129, February 2005.

[31] Y.-K. Lin, D.-W. Li, C.-C. Lin, T.-Y. Kuo, S.-J. Wu, W.-C. Tai, W.-C. Chang

and T.-S. Chang, “A 242 mW 10 mm 1080p H.264/AVC high profile encoder chip,” in *IEEE ISSCC Dig. Tech. Papers*, pp.314–315, February 2008.

[32] Z. Liu, Y. Song, M. Shao, S. Li, L. Li, S. Ishiwata, M. Nakagawa, S. Goto, and T. Ikenaga, “A 1.41 W H.264/AVC real-time encoder SOC for HDTV1080P,” in *VLSI Circuits Symp. Dig.*, pp. 12–13, June 2007.

[33] Y.-H. Chen, T.-D. Chuang, Y.-J. Chen, C.-T. Li, C.-J. Hsu, S.-Y. Chien and L.-G. Chen, “An H.264/AVC scalable extension and high profile HDTV 1080p encoder chip,” in *VLSI Circuits Symp. Dig.*, pp. 104-105, June 2008.

[34] G. Pastuszak, “Transforms and Quantization in the High-Throughput H.264/AVC Encoder Based on Advanced Mode Selection”, in *Proc. IEEE Symp. VLSI*, pp. 203-208, April 2008.

[35] D. Zhou, G. He, W. Fei, Z. Chen, J. Zhou, and S. Goto, “A 4320p 60fps H.264/AVC Intra-Frame Encoder Chip with 1.41Gbins/s CABAC,” in *VLSI Circuits Symp. Dig.*, pp.154-155, June 2012.

[36] G. Correa, C. Diniz S. Bampi, D. Palomino, R. Porto and L. Agostini, “Homogeneity and Distortion-Based Intra Mode Decision Architecture for H.264/AVC,” in *Proc. IEEE Intl. Conf. Elec. Circuits and Systems,* pp. 591-594, December 2010.

[37] http://iphome.hhi.de/suehring/tml/download/ H.264/AVC Reference Software JM17.0, 2010.

[38] Joint Collaborative Team on Video Coding, “High Efficiency Video Coding (HEVC) text specification draft 10,” JVT-G050, January 2013.

[39] C.-Y. Kao, C.-L. Wu, Y.-L. Lin, “A High-Performance Three-Engine Architecture for H.264/AVC Fractional Motion Estimation”, IEEE Trans. VLSI Syst., vol.18, No. 4, pp. 662-666, April, 2010.

[40] L.-F. Ding, W.-Y. Chen, P.-K. Tsung, T.-D. Chuang, H.-K. Chiu, Y.-H.Chen, P.-H. Hsiao, S.-Y. Chien, T.-C. Chen, P.-C. Lin, C.-Y. Chang, W.-L. Chen, and L.-G. Chen, "A 212 MPixels/s 4096 x2160p multiview video encoder chip for 3D/quad HDTV applications," in IEEE ISSCC Dig. Tech. Papers, February 2009.

[41] P.-K. Tsung, W.-Y. Chen, L.-F. Ding, C.-Y. Tsai, T.-D. Chuang, and L.-G. Chen, "Single-iteration full-search fractional motion estimation for quad full HD H.264/AVC encoding," in Proc. IEEE International Conference on Multimedia and Expo (ICME), pp. 9–12, 2009.

[42] T.-C. Chen, Y.-H. Chen, C.-Y. Tsai, S.-F. Tsai, S.-Y. Chien and L.-G. Chen, "2.8 to 67.2 mW low-power and power-aware H.264 encoder for mobile applications," in VLSI Circuits Symp. Dig., pp. 222–223, 2007.

[43] https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags

/HM-10.0/ HEVC Reference Software HM10.0, 2013.

[44] H. Sun, D. Zhou and S. Goto, "A low-complexity HEVC intra prediction algorithm based on level and mode filtering," IEEE International Conference on Multimedia and Expo, pp.1085-1090, June 2012.

Publication List

Journal:

[1] Gang He, Dajiang Zhou, Wei Fei, Zhixiang Chen, Jinjia Zhou, and Satoshi Goto, "A High-Performance H.264/AVC Intra Prediction Architecture for Ultra-HD Video Applications," IEEE Transactions on Very Large Scale Integration (VLSI) systems, 14 pages, Early Access, Issue: 99, pp.1, January. 2013.

[2] Gang He, Dajiang Zhou, Jinjia Zhou, and Satoshi Goto, "A 530Mpixels/s intra prediction architecture for ultra high definition H.264/AVC encoder, " IEICE Transactions on Electronics, 9 pages, Vol. E94-C, No. 4, pp. 419-427, April, 2011.

[3] Jinjia Zhou, Dajiang Zhou, Gang He, and Satoshi Goto,"Cache based motion compensation architecture for quad-HD H.264/AVC video decoder," IEICE Transactions on Electronics, 9 pages, Vol. E94-C, No. 4, pp. 439-447, April, 2011.

International Conference:

[1] Gang He, Dajiang Zhou, Zhixiang Chen, Tianruo Zhang and Satoshi Goto, "A 995Mpixels/s 0.2nJ/pixel fractional motion estimation architecture in HEVC for Ultra-HD", IEEE Asian Solid State Circuit Conference (A-SSCC'2013), 4 pages, Singapore, pp. 301-304, November, 2013.

[2] Gang He, Dajiang Zhou, and Satoshi Goto, "Transform-based fast mode and depth decision algorithm for HEVC intra prediction", IEEE International Conference on ASIC (ASICON'2013), 4 pages, Shenzhen, China, pp. 621-624, October, 2013.

[3] Jiayi Zhu, Dajiang Zhou, Gang He, and Satoshi Goto, "A combined SAO and De-blocking filter architecture for HEVC video decoder", International Conference on Image Processing (ICIP'2013), 4 pages, Melbourne, Australia, pp. 1967-1971, September, 2013.

[4] Jinjia Zhou, Dajiang Zhou, Gang He, and Satoshi Goto, "A 1.59Gpixel/s Motion Estimation Processor with -211-to-211 Search Range for UHDTV Video Encoder", Symposium on VLSI Circuits (VLSI'2013), 2 pages, Kyoto, Japan, pp. 287-288, June, 2013.

[5] Dajiang Zhou, Gang He, Wei Fei, Zhixiang Chen, Jinjia Zhou, and Satoshi Goto, "A 24.5-53.6pJ/pixel 4320p 60fps H.264/AVC intra-frame video encoder chip in 65nm CMOS", Asia and South Pacific Design Automation Conference (ASP-DAC'2013), 2 pages, Yokohama, Japan, January, 2013, University Design Contest.

[6] Gang He, Dajiang Zhou, Jinjia Zhou, and Satoshi Goto, "A 1991 Mpixels/s intra prediction

architecture for Super Hi-Vision H.264/AVC encoder", European Signal Processing Conference (EUSIPCO'2012), 5 pages, Bucharest, Romania, pp. 1054-1058, August, 2012.

[7] Dajiang Zhou, Gang He, Wei Fei, Zhixiang Chen, Jinjia Zhou, and Satoshi Goto, "A 4320p 60fps H.264/AVC intra-frame encoder chip with 1.41Gbins/s CABAC", Symposium on VLSI Circuits (VLSI'2012), 2 pages, Honolulu, USA, pp. 154-155, June, 2012.

[8] Jinjia Zhou, Dajiang Zhou, Gang He, and Satoshi Goto, "A 16-65 cycles/MB H.264/AVC motion compensation architecture for quad-HD applications", European Signal Processing Conference (EUSIPCO'2011), 5 pages, Barcelona, Spain, pp. 728-733, August, 2011.

[9] Gang He, Dajiang Zhou, Jinjia Zhou, and Satoshi Goto, "In a prediction architecture for H.264/AVC QFHD encoder", Picture Coding Symposium (PCS'2010), 4 pages, Nagoya, Japan, pp. 450-453, December, 2010.

[10] Jinjia Zhou, Dajiang Zhou, Gang He, and Satoshi Goto, "A bandwidth reduction scheme and its VLSI implementation for H.264/AVC motion vector decoding", 10 pages, Pacific Rim Conference on Multimedia (PCM'2010), Shanghai, China, pp. 52-61, September, 2010.