A 1991 MPIXELS/S INTRA PREDICTION ARCHITECTURE FOR SUPER HI-VISION H.264/AVC ENCODER

Gang He, Dajiang Zhou, Jinjia Zhou, and Satoshi Goto

Graduate School of Information Production and Systems, Waseda University
2-7 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka, Japan

ABSTRACT

This paper presents an H.264/AVC intra prediction design for Super Hi-Vision (SHV) video. Due to huge throughput requirements, design challenges such as data dependency and complexity become even more critical. To solve these problems, we first propose an interlaced block reordering scheme together with a coarse-to-fine mode decision (CFMD) strategy to resolve the data dependency between intra mode decision and reconstruction. Circuits area is reduced in the meantime with CFMD. We also propose a probability-based reconstruction scheme to solve the problem from long pipeline latency. As a result, hardware complexity in terms of the product of area and frequency is reduced by 74%. The maximum throughput reaches 1991Mpixels/s for 7680x4320p 60fps video. Total logic gate count is 451.5k in 65nm library.

Index Terms—H.264/AVC, intra prediction, SHV, hardware architecture.

1. INTRODUCTION

While 1080p is the current main stream of video services, Super Hi-Vision (SHV, also known as ultra high definition) with 7680x4320 pixels at 60fps, or 32 times more resolution than 1080p HD, has been targeted by next-generation applications. Not only an enhanced sensation of reality Super Hi-Vision can provide, but also rich monocular cues for 3D impression such as visual field angle, linear perspective, and texture gradient [1]. To store and transmit the huge data of SHV, efficient video encoding and decoding technologies are indispensable. H.264/AVC [2] is a promising technology for its advancement in coding performance. As an important component of H.264, intra prediction uses neighboring pixel values to predict the current block. With intra prediction, H.264 I-frame coding delivers better coding efficiency than JPEG 2000. In P and B frames, intra prediction also improves the quality when motion estimation fails to get a good match. However, efficient design of intra prediction architecture is challenged by the critical data dependency. The long reconstruction loop limits the possibility of parallelism for hardware design.

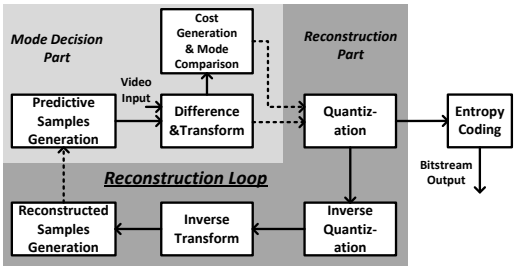

Fig. 1. H.264/AVC intra-frame encoding flow

Moreover, for SHV, the pipeline latency reduces the hardware efficiency due to serious timing constraint.

Previous contributions on intra prediction architectures can be briefly classified into three categories. The first common strategy is interleaved schedule for different prediction modes [3]. 16x16 prediction is inserted into 4x4 prediction to reduce the pipeline bubbles. However, it only benefits mode decision part of the architecture. The second strategy is the prediction with original pixels instead of reconstructed ones [6], so that data dependency can be removed from the mode decision. The demerit is the degradation of coding efficiency especially for lower-bit-rate cases. The third strategy is to modify the processing order of blocks in a macroblock (MB) by analyzing data dependency [4]. It increases the opportunity for parallel processing. However, it does not apply to 8x8 and 16x16 modes that are more important to high-resolution videos.

In this work, an efficient architecture for intra prediction is proposed. With the specifications of recent architectures limited to 1080p and 2160p, this design targets for SHV 7680x4320 60fps real-time encoding. The proposed schemes are characterized as follows. (1) An interlaced block reordering (IBR) scheme together with coarse-to-fine mode decision (CFMD) fully allows parallel operations of mode decision and reconstruction parts. Complexity is reduced in the meanwhile. (2) A probability-based reconstruction (PBR) scheme solves the problem from pipeline latency. Based on the results of coarse decision, reconstruction process is advanced, which improves hardware utilization. Overall hardware complexity is reduced by 74%. The implemented design can support real-time SHV 4320p 60frames/s encoding, running at 266MHz, and deliver 1991Mpixels/s throughput.

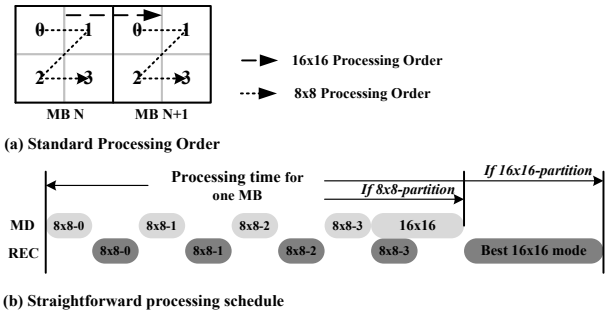

Fig. 2. (a) Standard zig-zag scanning order (b) processing schedule

The rest of this paper is organized as follows. Section 2 gives the analysis of H.264 intra prediction. In section 3, the proposed schemes are presented. The hardware architecture is described in Section 4. The implementation results are shown in Section 5. Section 6 concludes the paper.

2. ANALYSIS OF H.264/AVC INTRA PREDICTION

Fig. 1 shows the H.264/AVC intra-frame encoding flow. Firstly, predictive sample generation (PSG) unit refers to reconstructed pixels of neighboring block to generate predictive pixels for each mode. Then the residues are generated and transformed into coefficients. The costs are estimated with coefficients to compare the modes. Quantization unit processes the coefficients of the best mode and outputs results to both inverse quantization unit and entropy coding unit. The inverse quantization and inverse transform units translate quantized values back to residuals, which are used to reconstruct pixels for PSG operation of the next block. Entropy coding unit encodes quantized values and mode information for bit-stream output.

The realization of high hardware utilization is challenged by the reconstruction loop shown in Fig. 1. We divide all modules into a mode decision (MD) and a reconstruction (REC) part. Fig. 2(a) illustrates the standard zig-zag scanning order in H.264/AVC intra-frame coding. Fig. 2(b) shows the processing schedule with this order. For 8x8 predicting, MD process of each block needs reconstructed pixels of its previous block, which makes MD hardware idle during the REC time. Meanwhile, REC part is idle during MD's working time. For 16x16 predicting, although MD can work parallel with the REC of 8x8, it still introduces bubbles due to different workload. In addition, the REC of 16x16 starts after MD process of both 16x16 and 8x8 are finished. MD part has to be idle during the 16x16 REC time. The critical data dependency serious limits the parallelism of MD and REC. For an efficient architecture, this problem must be solved.

Pipeline latency influences the efficiency of pipeline with data dependency. The problem becomes critical in an SHV encoder, due to the limited processing time budget. Fig. 3 shows the principle of pipeline efficiency with latency influence. In execution time for MD, the efficiency can be calculated with an equation $p/(p+q)$, where p and q respectively denote the effective execution time and the latency of the pipeline. For the pipeline design of intra prediction, p is mainly decided by the throughput requirement. Meanwhile, q depends on the total number of pipeline stages, which is decided by maximum operating frequency and 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, thus has nearly no influence on the pipeline efficiency. However, for SHV encoding, every MB should be processed in only 30~40 clock cycles. Assuming a 4-cycle pipeline latency and 8-cycle execution time for each 8x8 block, the pipeline efficiency decrease from the latency can be 33%.

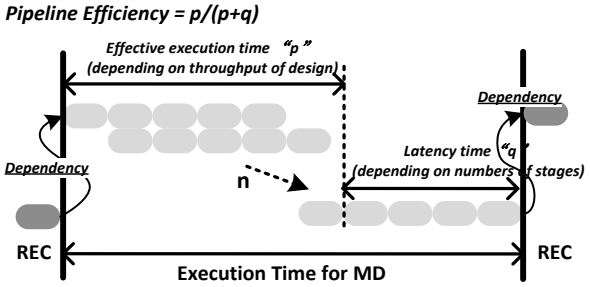

Fig. 3. Pipeline latency problem

3. PROPOSED SCHEMES

3.1. Interlaced Block Reordering

The design targets for very high throughput application such as 4320p and 2160p. Experimental result shows that 4x4 prediction is not significant to the performance for high-resolution sequences. Therefore, 8x8 and 16x16 modes are supported in this design.

As explained in section 2, the difficulty to parallelize the MD and REC parts is one of the major obstacles to realize high hardware utilization. In this work, an interlaced block reordering (IBR) scheme is proposed to solve this problem.

Firstly, we assume only using 8x8 mode prediction. The problem in this case is that blocks with data dependency are processed sequentially. By analyzing the dependency of blocks in two consecutive MBs, processing order is rearranged. The lower 2 blocks of L-MB (left MB) and the upper 2 blocks of R-MB (right MB) are processed with an interlaced manner, as shown in Fig. 4(a). This order ensures no dependency between two sequentially processed blocks, which parallels MD and REC. It is illustrated in Fig. 4(b). For a current block N, a non-reference block N-1 is inserted in the processing order, between current block and its neighboring reference block N-2. By doing so, MD of the current block N can start right after that of block N-1 without any idle time, as long as the REC of block N-2 is completed during MD of block N-1.

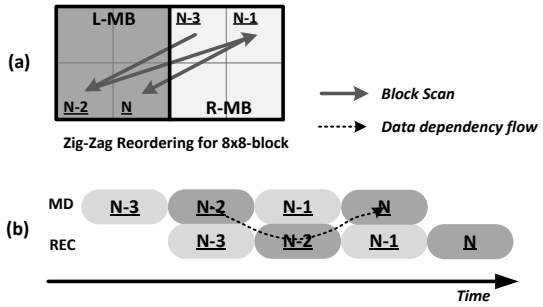

Fig. 4. (a) Zig-zag reordering for 8x8-partition case. (b) Proposed MD/REC parallel processing.

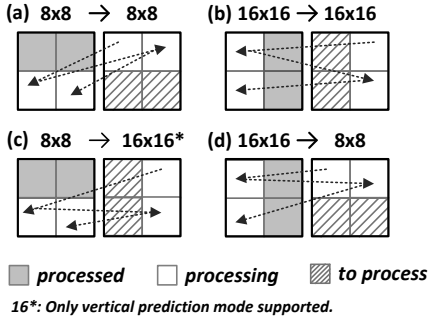

Fig. 5. Interlaced block reordering for variable-partition cases

Now we further take 16x16 modes into consideration, but still assume the following two statements. (a) the choice of whether 8x8 or 16x16 partitioning should be used is decided prior to MD. (b) If 16x16 partitioning is selected, the best 16x16 is also decided. Based on this given information, MD is focused on deciding the best mode for 8x8 blocks.

Fig. 5 illustrates all the 4 possible combinations of 8x8/16x16 partitioning of consecutive MBs. The scanning order shown in Fig.4(a) cannot apply to the cases with 16x16 partition. This is because when R-MB chooses 16x16 partition, the MD of each block within R-MB cannot start until the REC for right blocks of L-MD are all finished. The problem with R-MB 16x16 partitioning is discussed by considering two cases as follow. (1). L-MB chooses 16x16 partition (case(b) in Fig. 5). There is no data dependency between the blocks within the MB which chooses 16x16 partition. Therefore, the solution is to process right blocks first which have data dependency with the next MB. By doing so on all MBs with 16x16 partitioning, the scanning orders are arranged as shown. In addition, with this reordering, no problem of data dependency happens in case (d). (2) L-MB chooses 8x8 partition (case(c) in Fig. 5). It is infeasible to process the blocks in R-MB before the REC of L-MB is finished. This can be solved by supporting vertical mode only for R-MB since it only needs upper reference pixels.

By using IBR, MD and REC can operate in parallel in 8x8-block level. But a remaining problem is with the above assumption about the MB partitioning and 16x16 mode. This is to be solved with the coarse-to-fine mode decision scheme.

3.2. Coarse-to-Fine Mode Decision

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

In addition, the operations in MD need to be processed by many iterations. A straightforward implementation suffers from high complexity. An efficient way to prevent this is to reduce the candidate modes processed in the MD, which also requires pre-processing. Since 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 a coarse-to-fine mode decision (CFMD) scheme. Using original pixels as the reference, which involves no data dependency, all intra modes are computed and compared in the coarse decision (CD, as the pre-processing). Based on the evaluation in CD, a reduced number of candidate modes are sent to the fine decision (FD) stage for further refinement. With CFMD and IBR applied, CD, FD and REC can work in parallel. In addition, this approach can reduce computation complexity. Since CD only generates candidate modes, instead of making final decisions, a relatively inaccurate cost function is acceptable. Therefore, sum of absolute difference (SAD) is applied for CD.

In our design, CD decides the MB partitioning, and generates 4 best candidate modes for FD if the MB is partitioned into 8x8 blocks. If 16x16 partitioning is selected, a best 16x16 mode is directly passed to FD. Due to mode reduction of FD and the simpler cost function used by CD, 35% area is saved compared with the straightforward MD design. Different from the method in [6] that use original pixels to decide the best mode, CFMD supports a fine decision with reconstructed pixels. It involves only 1.46% BD-bitrate increase, as shown in Table 2.

3.3. Probability-Based Reconstruction

As mentioned, due to the limited processing time budget of SHV, the problem of pipeline latency becomes critical. Fig. 6(a) shows the pipeline design targeted at processing each 8x8 block in 8 cycles. FD processes every mode in 6 cycles with 2-cycle delay. REC takes another 8 cycles. Owing to IBR, reconstructed pixels of block N is not needed until block N+2 starts. But this still leads to a 4-cycle idle for every 2 blocks, due to the pipeline latency. Probability based reconstruction (PBR) is proposed to shorten this idle. As shown in Fig. 6(b), REC starts right after FD finishes the

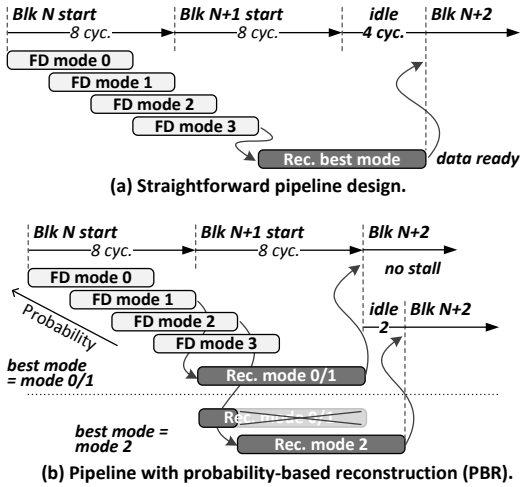

Fig. 6. Illustration of probability-based reconstruction.

Table 1. The variance of encoding speed with QPs (22, 27, 32, 37)

QP	Nebuta	Train	Ducks-TakeOff	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

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 0, 0, 2 or 4 cycles when mode0, 1, 2 or 3 is the best. To minimize the idle time, the modes are processed in an ascending order of SAD costs that have been calculated in CD. The mode with smaller SAD cost should have a high probability to be the best mode. The simulation results show that the first two modes own about 88% probability to cover the best.

By applying PBR, the processing time of 4 sequentially processed blocks varies from 32 to 40 cycles. The equivalent encoding speed can be considered as 32~40 cycles/MB. Owing to high probability of the hit-rate (first two modes being the best mode) in the video, the speed is close to 32 cycles/MB. Simulations are done with 6 videos under four QPs (37 32 27 22). Table 1 shows the result for each case. The time for an MB is the average value estimated from one frame. It varies 32.5~33.43 cycles and the worst case happens on Nebuta when QP equals to 32. The average time is 33.02 cycles, which reduces 18%, compared with the original 40.

4. HARDWARE ARCHITECTURE

Fig. 7 shows the overall intra prediction architecture based on above schemes. The entire architecture can be divided into MD and REC processing parts. In MD processing, CD engine with 96 pixels-parallelism and FD engine with 64 pixels-parallelism work together for luma component.

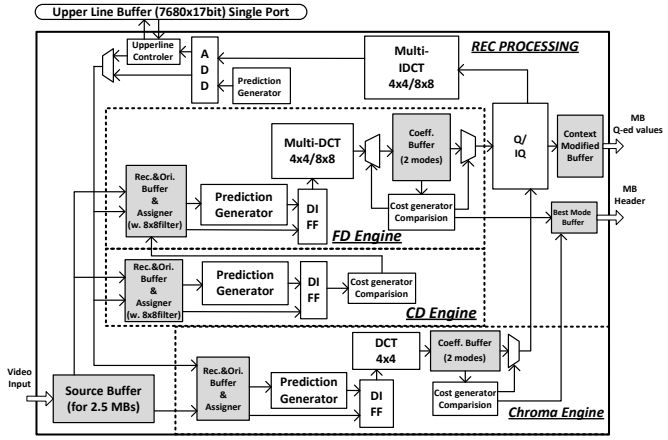

Fig. 7. Proposed intra prediction architecture.

Chroma component is processed by the chroma engine with 64 pixels-parallelism individually. Shared in a time-division manner for the luma and chroma, REC part adopt only 32 pixels-parallelism.

Due to limited processing time budget, the internal buffers are implemented by registers. In Fig. 7, gray blocks indicate register buffers in the design. 1). Originally, source buffer stores the pixels of an MB. IBR introduces additional storage for a half MB due to the interlaced operation for consecutive MBs. Moreover, since CD is the pre-processing operation to decide MB partition, which processes one MB prior. Thus, the storage for 2.5 MBs is needed in this design. 2). Buffer before the prediction generator stores and assigns the data including reference and original pixels. 3). Coefficient buffer stores coefficients according to the result of cost comparison, which saves the re-computation for the best mode. Due to the latency of comparison result, coefficients of 2 modes are stored. 4). Context modified buffer and best mode buffer are used to reorder quantized values and mode information and to output them as a standard order in MB level. In addition, to realize encoding system, the upper-line buffer is needed. In this design, it is implemented by the 16.3k bytes (7680x17bit) single port on-chip memory. The size is proportional to width of picture.

Fig. 8 shows the pipelining schedule of the design. It takes 32-40 cycles to perform an intra encoding process of an MB. The variation of encoding speed depends on MB partition and whether the best mode is in first 2 modes. Each block is processed with 8 cycles without the idle. In addition, chroma component is processed with standard zig-zag scanning order. The REC for chroma is divided into 4 parts. Every part is processed in one cycle by REC and interlaced with luma. For its MD operation, each mode is processed with 8 cycles by Chroma Engine. Vertical mode which does not need left reference is processed first. With reconstructing right part prior, horizontal and DC mode can be processed.

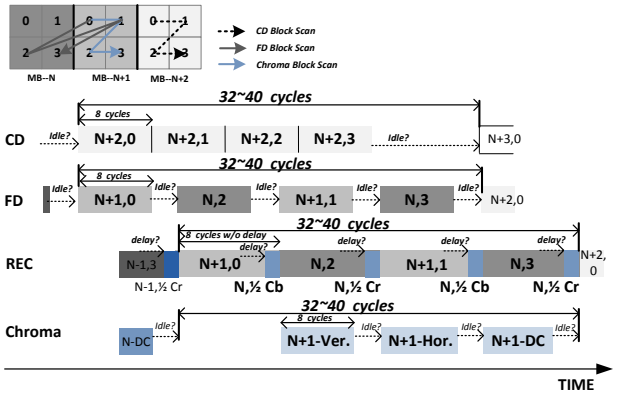

Fig. 8. Proposed pipelining schedule.

Table 2. The increase in BD-bit rate(%) by comparison with JM17.0 in QPs (22, 27, 32, 37)

Sequences	CFMD	CFMD+IBR
Nebuta(4320p)	1.23	1.85
Train(4320p)	0.81	1.36
DucksTakeOff(2160p)	1.37	1.41
CrowdRun(2160p)	1.49	1.94
Station(1080p)	1.56	1.78
Tractor(1080p)	2.34	2.46
Average	1.46	1.80

5. IMPLEMENTATION RESULTS

The proposed architecture is designed in Verilog HDL and implemented with e-shuttle 65nm SZ library. Supported prediction modes include all 9 directions for 8x8 and 3 directions (vertical, horizontal, DC) for 16x16. With adopted techniques, only 33 cycles are taken to process one MB average and 7680x4320 60fps real-time encoding can be achieved at 266 MHz. The total logic gate count is 451.5k in number of equivalent NAND2 gates (40.3k for CD engine, 141.1k for FD engine, 85.3k for chroma engine and 184.8k for REC). The maximum operating frequencies from synthesis and layout results are 400MHz and 300MHz respectively, which are enough for real time processing.

Table 2 shows the coding efficiency comparison with JM 17.0[7] in terms of BD-bit-rate increase. As shown in Table 2, 1.8% BD-bitrate increasing is introduced in this design. CFMD scheme takes the major part about 1.46%. Considering the contribution for hardware utilization and complexity reduction, it is acceptable.

According to the analysis in section 3, first CFMD can save 35% hardware cost first. Then, IBR and PBR can reduce 45%, 18% required frequency. The overall hardware complexity is reduced by 74%. The comparison with previous designs is shown in Table 3. The maximum throughput reaches 1991Mpixels/s, 32x to 49x faster than previous designs [3][4][5]. Furthermore, the hardware efficiency of the design is increased by 3.2-4.6 times. In addition, the 1080p 30fps encoding requires only less than 9MHz operating frequency, which is much lower when compared with previous works.

Table 3. Comparison with previous works

Design	Proposed	[3]	[4]	[5]
Max Throughput	4320 p @60fps	1080p @30fps	1080p @30fps	1408x960 @30fps
Technology	65nm	0.13um	0.13um	0.18um
Gate count *	451.5k	87.2k	150.4k	89k
Cycles/MB	33 avg.	560	464	620
Algorithm	CFMD	3 Steps	Full Search	Transform -Based
Supported Mode	8x8 & 16x16	4x4& 16x16	4x4 to 16x16	4x4 & 16x16
Freq. for 4320p @60 fps	266MHz	N/A	N/A	N/A
Freq. for 1080p @30fps	8.3MHz	140MHz	114MHz	N/A
Hardware efficiency*	16.6k	5.1k	3.6k	4.56k

*:Evaluated as *throughput/(gate_count*frequency)* pixel/(K-gate*M-Hz)

6. CONCLUSION

This paper discusses the design of intra prediction architecture for H.264/AVC encoder in 65 nm CMOS. An IBR scheme together with CFMD allows parallel operations of MD and REC and also reduces the complexity. Meanwhile, a PBR scheme solves the problem from pipeline latency. With adopted techniques, hardware complexity is reduced by 74%. Compared with state-of-the-art works, our design is the first architecture to support 7680x4320 real-time encoding at 60 fps when running at 266MHz.

7. REFERENCES

[1] Takayuki Ito, "Future television-Super Hi-Vison and beyond," *A-SSCC*, pp. 1-4, Nov. 2010.
[2] 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.
[3] 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. CSVT.*, vol. 19, no. 3, pp.432-436, 2009.
[4] 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 VLSI*, vol. 19, no. 6, pp. 925-938, June 2011.
[5] 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 CSVT*, vol. 20, no. 6, pp. 894-906, June 2010.
[6] 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," *JSSC*, vol. 45, no. 1, pp. 46-58, 2010.
[7] http://iphome.hhi.de/suehring/tml/download/ H.264/AVC Reference Software JM17.0, 2010.