# PIPELINING ARCHITECTURE DESIGN OF THE H.264/AVC HP@L4.2 CODEC FOR HD APPLICATIONS

*Kiwon Yoo[1,2], Jun Ho Cho[1], Chan-Sik Park[1], TaeKyoung Ahn[1], Jae-Hun Lee[1], and Kwanghoon Sohn[2]*

[1]Digital Media R&D Center, Samsung Electronics, Suwon, Korea
[2]Department of Electrical Engineering, Yonsei University, Seoul, Korea
E-mail: ykiwon@samsung.com

**ABSTRACT**

This paper presents the macroblock/slice level pipeline structure for an H.264/AVC HP@L4.2 codec. In H.264/AVC, level 4.2 (L4.2) in high profile (HP) describes the encoding/decoding capability of 1920×1088@64p sequence/ bitstream of up to 62.5 Mbps. To meet this tremendous specification, the novel hardwired architecture of the H.264/AVC codec is also presented. It supports both encoding and decoding and shares commonly used hardware modules. In our system, the video subsystem including the H.264/AVC codec is classified into four principle functions: video coding, memory management, reference cache-buffer control, and top control. With regard to H.264/AVC processing, the video coding function comprises eight modules. These modules are arranged as a six-stage macroblock pipeline for the encoder and a four-stage macroblock pipeline for the decoder. With the proposed schemes adopted, a software C model and an FPGA platform were developed for verification. The simulation results indicate that our design approach successfully performs the real-time encoding/decoding of the H.264/AVC HP@L4.2 sequence/bitstream at an operating frequency of 266MHz.

***Index Terms***: H.264/AVC, high profile, level4.2, VLSI, pipelining structure

## 1. INTRODUCTION

With the efforts of the Joint Video Team (JVT), formed by both the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG), the present standardization process of H.264/AVC has been finalized in 2004 and it provides the good video quality at substantially lower bitrates than previous standards and increased flexibility in the use of various applications [1]. H.264/AVC outperforms other previous coding standards in coding performance; however, this is achieved at the cost of extensive computational complexity. Specifically, the resulting coding gain comes from state-of-the-art coding tools such as context-adaptive entropy coding, intra prediction, multiple reference frames, more accurate pixel interpolation, variable block sizes, and in-loop filter particularly for low bitrates.

In recent years, several preceding researches pertaining to this subject have investigated the analysis, design, and implementation of the hardware architecture of the H.264/AVC BP/MP codec for HD application [2][3]. According to these researches, hardware acceleration is a prerequisite for the real-time application of H.264/AVC due to high computational complexity, memory access requirements, data dependency, and a long coding path. In addition, they drew the optimal design and implementation based on target specifications. However, adopting those designs intactly just by means of (1) raising the operating frequency or (2) putting an additional bus/memory/ pipeline stage to suit the design, because of the lower memory bandwidth requirement and longer processing cycles of macroblock pipeline stage, is undesirable. In particular, applying these designs is at best a suboptimal solution for our design specification in resource/power aspects.

In this paper, in order to overcome the design obstacles to achieve the efficient codec architecture and its real-time processing, the macroblock level pipeline structures for the H.264/AVC HP@L4.2 encoder/decoder are presented together with bandwidth reduction schemes for external memory. The proposed H264/AVC system comprises three principal sub-systems: audio, TS, and video. The video sub-system is further divided into four principal functions, namely, video coding, memory management, reference cache-buffer control, and top control. In consideration of data dependency, resource sharing, and processing cycle, we divide the video coding function into eight primitive coding modules: SDMA, IPME, SPMES, SPMEM, MCR, RECON, ENT, and DEBLK. These modules are rearranged as a six-stage macroblock pipeline for the encoder and a four-stage macroblock pipeline for the decoder according to different coding purposes and higher level of parallel processing. With the proposed schemes adopted, the software C model and FPGA platform were developed for verification. The simulation results indicate that our design approach successfully meets the real-time encoding/decoding of the H.264/AVC HP@L4.2 sequence/bitstream at an operating frequency of 266MHz.

The remainder of this paper is organized as follows. In Section 2 describes our system architecture. Section 3 presents the proposed macroblock level pipeline structures; Section 4, the implementation results. Finally, the concluding remarks are provided in Section 5.

## 2. DESIGN CONSIDERATION

In H.264/AVC, in comparison with the main profile (MP), the high profile (HP) represents some additional coding functionalities, which includes monochrome (4:0:0) support for input format flexibility and nine intra8×8 modes, 4×4/8×8 transform adaptivity, quantization scaling matrices, and separate Cb/Cr QP control in order to improve the coding performance and codec flexibility; moreover, the HP is regarded as 20% more complex than the MP with the target applications

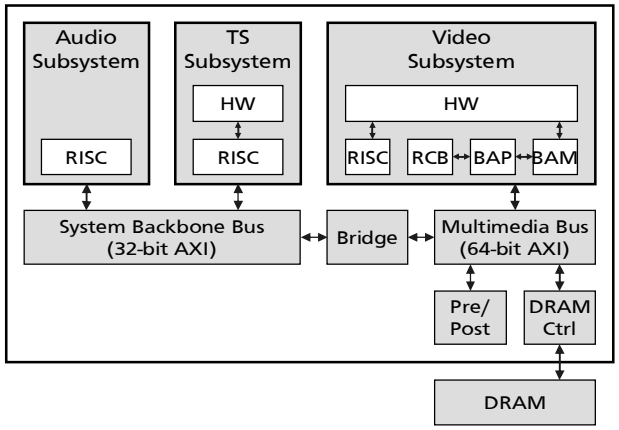


Fig. 1. Block diagram of the overall system

remaining the same. The term of level 4.2 (L4.2) in the high profile describes the encoding/decoding capability of 1920× 1088@64p sequence/bitstream of up to 62.5 Mbps, respectively. The bitrate (bps) and the macroblock throughput (MBps) of HP@L4.2 respectively need to be 3.125 and 2.125 times more than those of MP@L4.0, which is a specification for HDTVs.

The excessive HP@L4.2 specification results in two major problems for hardware design and its implementation. First, the processing cycles are inversely proportional to the operating frequency and the data rates. Subsequently, this greatly increases the hardware cost and the design overhead. For efficient hardware implementation, highly utilized parallel architectures with hardware-oriented encoding algorithms are required. Second, the bus bandwidth requirement also increases alongside higher data rates. For example, the encoding of a 4:2:0 1920×1088 reference-B picture (1 slice/picture) with 2 references requires a minimum of about 12.5 Mbytes. Hence, efficient memory management and bus arbitration must be considered to reduce the bus bandwidth.

Our system supports encoder/decoder functionalities together. Thus, the commonly employed hardware modules must be designed such that they can share both coding paths. This can be achieved by module interface (I/F) modification by means of a codec controller as well as hardware register setting by an RSIC processor.

According to the H.264/AVC HP@L4.2 specification, the maximum macroblock processing rate is 522,240 MBps for 1920×1088@64p. According to our peak specification, the target macroblock processing rate is 489,600 MBps for 1920×1088@60p. When the system runs at 266 MHz, the maximum processing cycles are represented by 543 clock/MB, that is, the value of the system operating frequency divided by the target macroblock processing rate. However, considering the time taken by the RISC unit to process the sequence parameter set, picture parameter set, and slice header information of H.264/AVC and to manage the system level control signal and the system margin for real-time applications, the target processing cycles of our design specification are set to 500 clock/MB with an approximate 10% margin.

## 3. PROPOSED PIPELINE ARCHITECTURE

### 3.1. System Overview

Our system is targeted for HD applications that include HDTVs and next-generation optical discs (e.g. Blu-ray, HD-DVD). On the other hand, one of the main aims of our research is to achieve more data throughput with less resources/power in the coming applications, which is scheduled to commence shortly. Fig. 1 shows the block diagram of this system.

Table 1. Coding modules and their operation description: Encoder only (e), decoder only (d)

| | |
|---|---|
| RC (e) | Rate-control by a scheme that maintains constant quality at target bitrate |
| INTE (e) | Intra-mode derivation |
| CC (e) | Correction of the missing color information caused by luma ME, by properly modifying MBQP |
| MD (e) | Mode decision |
| INTD | Predictor generation in an intra macroblock |
| TNQ | IDCT/DCT/Q/IQ |
| MVP | Derivation of motion vector predictor |
| SDMA | It performs reference data management for a given buffer size for IPME/SPMES/SPMEM /MCR, specifically pre-buffering of reference picture to improve bus efficiency. It requires an amount of reference cache-buffers to storage |
| IPME (e) | Integer-pel motion estimation |
| SPMES (e) | Searching part of sub-pel motion estimation |
| SPMEM (e) | MC part of sub-pel motion estimation |
| MCR (d) | It reads reference data from external memory for motion compensation when SDMA doesn't take it. |
| MCP | Motion compensation |
| RECON | It performs macroblock reconstruction from outputs of TNQ, INTD, and MCP. |
| DEBLK | In-loop filter |
| ENT | It performs the packing/unpacking of data. In fact, the coding is performed in a TS subsystem. |

The system comprises three principal sub-systems: audio, TS, and video. The audio subsystem performs the encoding/decoding of Blu-ray/HD-DVD audio using a dedicated audio RISC. The TS subsystem processes the packing/ unpacking of Blu-ray/HD-DVD wrapper, multiplexing/ demultiplexing of TS bitstream, and encoding/decoding of the H.264/AVC entropy coder (CAVLC/CABAC). The video subsystem is responsible for encoding/decoding the functionalities of the H.264/AVC video. The video subsystem is further divided into four principal functions, namely, video coding, memory management, reference cache-buffer control, and top control. Taking into account data dependency, resource sharing, and parallel processing, and the processing cycle, the video coding function is further separated into eight primitive coding modules: SDMA, IPME, SPMES, SPMEM, MCR, RECON, DBLK, and ENT. These coding modules may include several subcoding modules. For example, IPME includes RC. The following seven subcoding modules are in an increasing stage order of encoder: RC, INTE, CC, MD, INTD, TNQ, and MVP. Table 1 presents a brief description of all the coding modules. The memory management function consists of the following two modules: BAP (bus access port) and BAM (bus access multiplexer). All requests from the video subsystems to the external memory pass through BAM/BAP. According to our analysis on bus request types, patterns, timings, and lengths with respect to video sub-system, these modules performs bus arbitration and choose the most adequate request among complex request inputs.

The reference cache-buffer control (RCB) carries out pre-

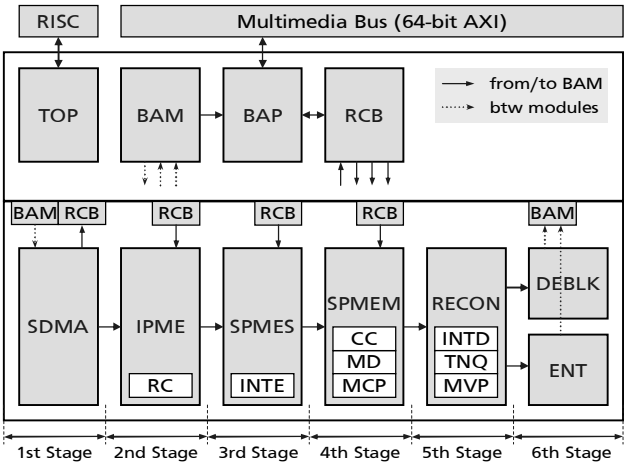


Fig. 2. Block diagram of the encoding system with the six-stage macroblock pipelining architecture

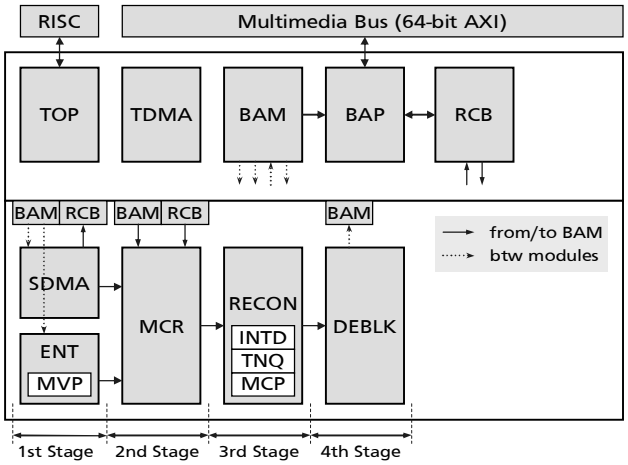


Fig. 3. Block diagram of the decoding system with the four-stage pipelining architecture

buffering of the reference data for inter prediction to reduce bus bandwidth by means of reusing reference data. The proposition for this is that the reference index and motion vector are distributed in limited ranges over most cases of inter prediction. According to our profiling, this proposition is more than 90% accurate. However, this approach has an inevitable shortcoming in that it needs a mass buffer for reference pictures.

The top control function (TOP) is responsible for overall controls for the video subsystem with regards to the following jobs: codec register load, coding flow control, I/F arrangements between coding modules, and stage buffer management.

### 3.2. Two-level pipeline architecture

We employ the two-level pipeline architecture in this system. First, a parsing part by a RISC processor and a coding part by a hardwired codec together form a slice-level pipeline structure. Subsequently, the coding part operates under the macroblock-level pipeline structure. In the slice-level pipeline process, each of the parsing and coding parts becomes a pipeline stage. In the parsing stage, the parsing part processes the slice data and the codec register setting. In the coding stage, the hardwired codec begins processing according to codec registers set by the RISC processor. The two-pipeline stage is executed concurrently and stalls whenever the previous stages are not completed on time.

The second level pipelining is carried out on a macroblock level in the coding part. The detailed description of the macroblock-level pipelining structures is described in the following subsection.

### 3.3. Macroblock-level pipeline architecture

The proposed system architecture and the macroblock-level pipeline structure, with regard to the encoder are shown in Figure 2. We employ the six-stage macroblock pipeline structure, which contains seven primitive coding modules, namely, SDMA, IPME, SPMES, SPMEM, RECON, DEBLK, and ENT. A brief description of these has already been presented in Table 1. According to our analysis, we design all stages and primitive modules under the target processing cycles with evenly distributed computation and bus requirements.

With regard to the decoder, its design is very similar to that of the encoder system. The decoder system architecture and its macroblock-level pipelining structure are presented in Figure 3. The decoder system employs the four-stage macroblock pipelining structure, containing five primitive coding modules, namely, SDMA, ENT, MCR, RECON, and DEBLK. The modules, which are commonly adopted by both the encoder and decoder, are reused.

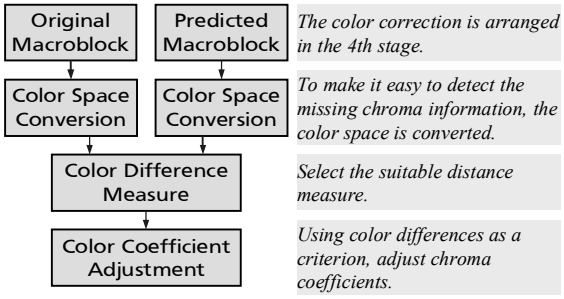


Fig. 4. Flow diagram of the color correction

### 3.4. Hardware algorithm modification

To improve the codec performance, we address the four principal issues that are adopted in our system. First, the primitive coding modules, such as SDMA, DEBLK, ENT, and MCR, need the external memory access. Here, to reduce waiting cycles, these modules adopt pre/post-buffering. For example, in the (N-1)th MB, pre-buffering prepares any information for Nth MB. Thus, a processing in the Nth MB begins without data waiting. Even if it requires the store buffer of 2 Mbytes or more, it deserves to be adopted, particularly for real-time HD applications.

Second, the typical ME process uses only luma data due to issues pertaining to human vision and the trade-off of visual quality and computational complexity. However, it cannot search the accurate MV if the current region and reference region have the same luma values but different chroma values. According to our approach, following the application of the conventional ME process, our color correction (CC) method is applied to compensate for the missed color residual. This procedure is illustrated in Figure 4. The color correction can be regarded as a suboptimal method to compensate missing color information with much less resource/computational complexity than the luma/chroma ME.

Table 2. Encoder profiling results of processing cycles

| | I-slice | P-slice | B-slice |
|---|---|---|---|
| stage 1 | 254 | 487 | Same as P |
| stage 2 | 280 | 488 | |
| stage 3 | 448 | 448 | |
| stage 4 | 2 | 442 | |
| stage 5 | 328 | 419 | |
| stage 6 | 448 | 425 | |

Table 3. Decoder profiling results of processing cycles

| | I-slice | P-slice | B-slice |
|---|---|---|---|
| stage 1 | 285 | 278 | 274 |
| stage 2 | 2 | 50 | 168 |
| stage 3 | 156 | 130 | 138 |
| stage 4 | 340 | 340 | 340 |

Third, the entropy coder in H.264/AVC has a great amount of data dependencies by means of context adaptation; hence, it requires sequential processing by bin-wise recursive processing. Moreover, its processing cycles are proportional to data rates and thus, in a worst case scenario, they are far beyond the target processing cycles to prevent real-time processing, while its average processing cycles are definitely lower than the target cycles. To resolve this problem, we arrange the entropy coder into the TS subsystem, and data exchange between the TS subsystem and the video subsystem is achieved through external memory. Here, a hardware accelerator is for entropy encoding/decoding by 1 clock/bin processing.

Lastly, IPME requires an accurate MVP for a search center and an MV cost. However, IPME and MD are in a different stage, and thus, IPME never uses final MVs to derive MVPs. As a suitable alternative, IPME uses the intermediate MVs, which is derived from itself as inputs for MD. It operates well with minor quality degradation.

## 4. IMPLEMENTATION RESULTS

The H.264/AVC HP@L4.2 codec that adopts the proposed pipeline architecture has been developed and tested on our FPGA platform. The real-time processing was verified using 2 1920×1088@60p sequences and its encoded bitstreams with 40 Mbps bitrate. The profiling results of the processing cycles at each stage are described in Table 2 for the encoder and Table 3 for the decoder. Figure 4 presents the codec configuration employed in the verification. According to the profiling results, all stages are under the target frequency cycles of 500 clock/MB for the real-time applications. Thus, the proposed architecture meets the design constraints in terms of the H.264/AVC HP@L4.2 codec.

Figure 5 shows the RD performance comparison our hardware-like software codec with JM12.2. The encoding configuration of JM12.2 is as similar to that of our codec as possible. The RD curve shows that the encoded result of our design is considerably comparable with that of JM at a minor PSNR degradation of 0.1 to 0.3dB.

## 5. CONCLUSION

We presented the novel hardware architecture and the macroblock-level pipeline structure for the H.264/AVC HP@L4.2 codec. This design supports both encoding and decoding and shares commonly used hardware modules. The proposed pipeline structure contains a two-level slice pipeline, six-stage macroblock pipeline for the encoder, and four-stage macroblock pipeline for the decoder. The design has been verified on the hardware-like C model and the FPGA platform. According to the simulation results, the proposed pipeline architecture was suitable for the real-time processing. Besides, through the RD performance comparison, our codec is considerably comparable to JM12.2 at an ignorable PSNR loss.

Table 4. Configuration of the proposed codec for a test

| | |
|---|---|
| Encoding Features | High Profile @ Level 4.2 (1920×1088@60p) |
| Decoding Features | Fully compliant to HP@L4.2 (up to 1920×1088@60p for real-time processing) |
| Maximum Number of Reference | 2 in P/B |
| Maximum SR | H[-2048.75,2047.75] V[-56.75,55.75] in quarter pel unit |
| Period I | Every 30 frames (0.5sec) |
| GOP structure | Two-level pyramid coding |

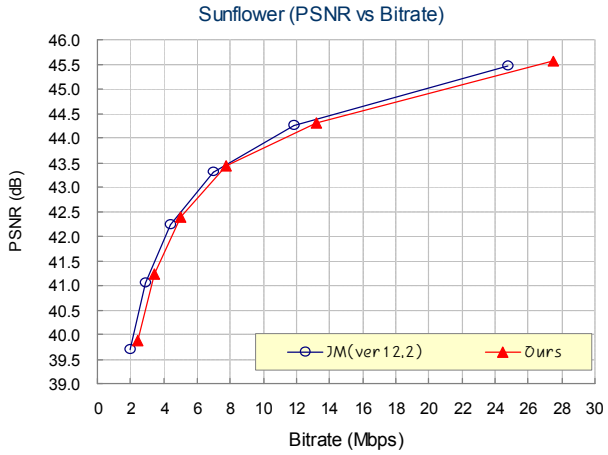


Figure 5. RD performance comparison of the proposed hardware-like software codec with JM12.2 (Sunflower, 1920×1088@30p [4])

## REFERENCES


[1] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC; tools, performance, and complexity," *IEEE Circuits Syst. Mag.*, Vol. 4, No. 1, 1Q, 2004.

[2] Y.-W. Huang, B.-Y. Hsieh, T.-C. Chen, L.-G. Chen, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC Encoder," *IEEE Trans. on CSVT,* Vol.16, No.16, Jun 2006.

[3] C.-C. Lin, J.-W. Chen, H.-C. Chang, Y.-C. Yang, Y.-H. O. Yang, M.-C. Tsai, J.-I. Guo, J.-S. Wang, "A 160K Gates/4.5 KB SRAM H.264 Video Decoder for HDTV Applications," *IEEE Journal of Solid-State Circuits,* Vol.42, No.1, Jan 2007.

[4] FTP server from Munich Technical University. [Online]: ftp://ftp.ldv.e-technik.tu-Menchen.de/pub/test_sequences/