Low Bandwidth Decoder Framework for H.264/AVC Scalable Extension

Tzu-Der Chuang, Pei-Kuei Tsung, Pin-Chih Lin, Lo-Mei Chang, Tsung-Chuan Ma, Yi-Hau Chen and and Liang-Gee Chen
DSP/IC Design Lab, Graduate Institute of Electronics Engineering
National Taiwan University, Taipei, Taiwan
Email: {peterchuang, iceworm, lpg, cutedolphin, tcm, ttchen, lgchen}@video.ee.ntu.edu.tw

***Abstract*—In the process of scalable video coding (SVC) decoding, large external memory bandwidth is required for SVC inter-layer prediction. In this paper, a low bandwidth decoder framework is proposed for SVC. Two main decoding schemes are developed to reduce the external memory bandwidth. Macroblock-level on-the-fly padding and on-line upsampling is proposed for SVC spatial scalability decoding. This scheme reduces 36% of decoding bandwidth and 34% of processing cycles. For SVC quality scalability, a layer-interleaving decoding scheme is proposed to eliminate all inter-layer prediction bandwidth which is 41~51% of decoding bandwidth. The corresponding hardware architectures of these two decoding schemes are also provided. This low bandwidth framework can save 33~52% of DRAM access power for SVC decoding.**

I. Introduction

In the past, video coding efficiency is always the main target of traditional video coding standards, such as MPEG-2/4 and H.264/AVC. However, due to the prevalence of streaming multimedia applications, more and more researchers pay attention to functionality and scalability. The Joint Video Team of ITU-T VCEG and ISO/IEC MPEG has standardized Scalable Video Coding (SVC) extension of the H.264/AVC standard [1][2] in July 2007. In SVC, three types of scalability, temporal scalability, spatial scalability, and quality scalability are provided. Frame-rate adaptation, multi-resolution variation, and bandwidth-fluctuated transmission applications could be easily supported.

In a video decoder, the most important design issue is the external memory bandwidth requirement. Large decoding bandwidth may exceed the bus bandwidth limitation and compete intensively with other IPs in an SoC system. Moreover, large external memory bandwidth results in huge DRAM power consumption, which can be several times larger than the power of the video decoder core. Several decoder designs have been proposed for low decoding bandwidth. Chuang proposes a cache-based motion compensation architecture to reduce data access bandwidth [3]. Zhu implements a dedicate DRAM controller to improve DRAM access efficiency [4]. Zhou integrates these two techniques to provide a low bandwidth video decoder platform [5]. However, all of these works focus on H.264/AVC only. In the SVC multi-layer decoding structure, large bandwidth is required for inter-layer prediction (ILP). This type of bandwidth may occupy over 50% of the total decoding bandwidth. In this paper, we first analyze the bandwidth overhead of SVC decoding. Then, two schemes are proposed to reduce the decoding bandwidth. For spatial scalability, frame-level padding and upsampling processes are replaced with macroblock-level on-the-fly padding and on-line upsampling. For quality scalability, the layer-interleaving decoding scheme is proposed to save all the bandwidth of ILP. The corresponding hardware architectures are also proposed.

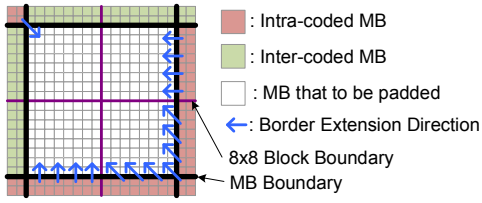

Figure 1. Illustration of padding process. The direction of border extension depends on the adjacent MBs' modes.

The rest of the paper is organized as follows. The bandwidth overhead analysis of SVC decoding is illustrated in section 2. Then, the proposed SVC decoding flow and hardware architecture are given in section 3 and 4, respectively. The hardware implementation results are addressed in section 5. Section 6 will conclude this paper.

II. Bandwidth Overhead Analysis of SVC Decoding

SVC adopts a multi-layer coding structure to provide various kind of video scalability. Compared with single layer decoding in H.264/AVC, the bandwidth overhead of SVC decoding is analyzed in this section.

A. *Temporal Scalability*

In SVC, it adopts H-B coding structure for temporal scalability. The more temporal layers are decoded, the more frame-rate is provided. However, the H-B coding structure is not a new coding tool because it has been defined in H.264 High Profile already. Therefore, there is no bandwidth overhead for SVC temporal scalability.

B. *Spatial Scalability*

In SVC, spatial scalability is achieved by coding different spatial resolution layers sequentially from low resolution to high resolution. In order to improve the coding efficiency, inter-layer motion/residual/intra predictions are adopted to remove the redundant information between spatial layers. ILP utilizes the base layer (BL) information to predict that in the enhancement layer (EL) thereby improving the coding efficiency. By use of inter-layer motion prediction, the macroblock (MB) modes and associated motion parameters in BL are reused. Inter-layer residual prediction (ILRP) reduces the energy of EL's residual by subtracting BL's residual. Inter-layer intra-prediction (ILIP) is to utilize reconstructed BL intra-signals (so called texture) as predictors. However, for SVC spatial scalability, the resolution of BL and EL is different. Hence, the upsampling process is required to synchronize the data amount between these two layers for ILP.

This work was supported in part by National Science Council, Republic of China, Taiwan, under the grant number NSC 97-2221-E-002-191-MY3.

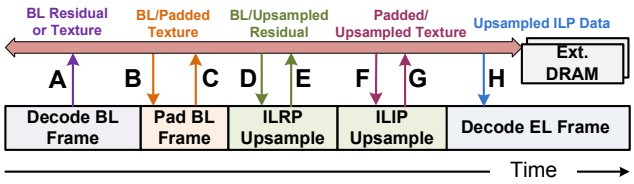

Figure 2. Typical decoding schedule for SVC spatial scalability.

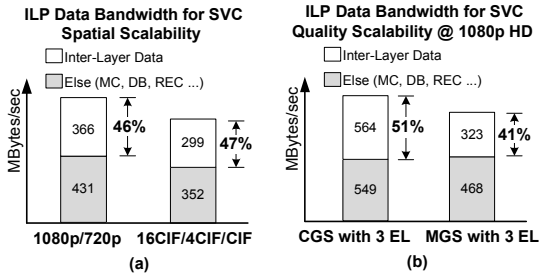

Figure 3. Bandwidth overhead for SVC spatial/quality scalability.

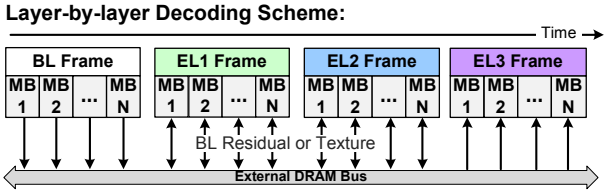

Figure 4. Typical decoding schedule for SVC quality scalability.

For ILRP, the residual signals in BL sub-block is block-wise upsampled with bilinear filter. The input pixels of bilinear filter do not across the block boundaries of transform. For ILIP, the prediction signals of EL MBs are obtained by upsampling the BL's reconstructed intra-signals. In the ILIP upsampling procedure, 4-tap and 2-tap FIR filters are applied for luma and chroma components respectively. Different from ILRP, filtering is always performed across sub-block boundaries. Therefore, when the neighboring blocks are not intra-coded and do not exist reconstructed intra-signals, the padding procedure is required prior to the upsampling procedure to generate the required samples for FIR filter input. Padding is a directional border extension operation as shown in Fig. 1 [2]. The direction of padding depends on the adjacent MBs' MB_modes.

Figure 2 shows a typical spatial scalability decoding schedule of the conventional SVC decoder hardware. The SVC decoder first decodes the BL frame and stores the BL residual and reconstructed intra-signals into the external memory. Then, padding is performed to generate required input samples for ILIP. After padding, the data of ILRP and ILIP are generated by upsampling. Then, the SVC decoder decodes the EL frame data with the prepared ILP data. From Fig. 2, two additional time slots are required for padding and upsampling; meanwhile, these two procedures require large memory bandwidth for ILP data transmission. The bandwidth can be classified into type A-to-H. Figure 3 (a) shows the bandwidth overhead for the decoding scheme in Fig 2. 46% of the decoding bandwidth is occupied by ILP in SVC spatial scalability.

C. *Quality Scalability*

In SVC, quality scalability is realized by coding more quality enhancement layers which are composed of refinement coefficients. Quality scalability is provided via two strategies, coarse-grain quality scalability (CGS) and medium-grain quality scalability (MGS) [2]. Quality scalability can be regarded as a special case of spatial scalability that the resolution of BL and EL is the same, but the quality of EL is better. Three ILP are also adopted in quality

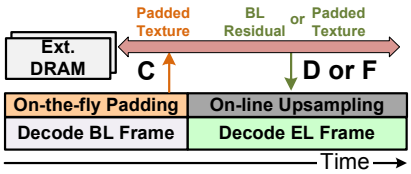

Figure 5. Proposed frame-level decoding schedule for SVC spatial scalability.

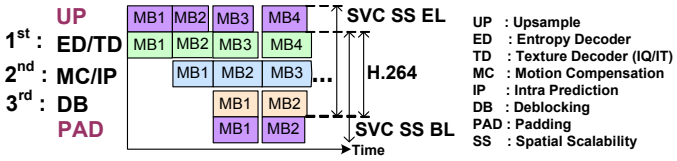

Figure 6. Proposed MB-level decoding schedule for SVC spatial scalability.

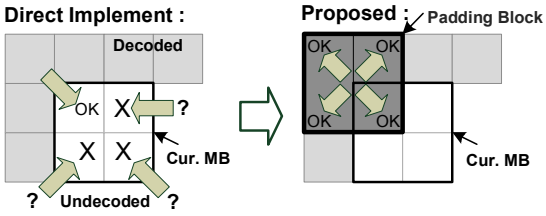

Figure 7. Proposed MB-level padding data flow

scalability. Figure 4 shows a typical quality scalability decoding schedule. Base on the content order of bitstream, the SVC decoder decodes one frame in a layer-by-layer order. The decoded ILP coefficients are stored in the external memory. The bandwidth overhead of quality scalability comes from frequently accessing these intermediate coefficients through the system bus. The bandwidth overhead increases with the number of ELs. From our simulation results, as shown in Fig 3 (b), the bandwidth overhead of quality scalability in a decoder is 51% and 41% for CGS with 3 ELs and MGS with 3 ELs respectively.

III. Proposed Decoding Scheme

In order to reduce bandwidth for SVC decoding, two low bandwidth decoding schemes are proposed in this section. The bandwidth overhead comes from accessing the ILP data through the system bus. These two schemes focus on how to reduce data transmission between the decoder and the external memory.

A. *MB-level Decoding Flow for Spatial Scalability*

To reduce processing cycles and bandwidth in Fig. **2**, the padding and upsampling processes are merged with BL and EL decoding. The MB-level on-the-fly padding and on-line upsampling schemes are proposed. As shown in Fig. 5, unlike independent processes of padding and upsampling in Fig. 2, padding is on-the-fly executed with BL frame decoding, and the ILP predictors are on-line upsampled with EL frame decoding. Figure 6 shows the proposed MB-level decoding schedule with 3-stage MB-pipeline. However, to merge the padding process into MB-level decoding schedule, the data dependency problem will arise. As shown in Fig 1, the padding procedure cannot be performed until all adjacent MBs are decoded. To solve the problem, we change the data flow of the padding procedure from padding the four 8x8 blocks of the current MB to padding its up-left four 8x8 blocks as shown in Fig. 7. In this flow, all the side information, such as neighboring MB modes and the pixel values of the boundary intra-signals, for border extension is decoded. This padding flow can be integrated into the MB-level decoding flow

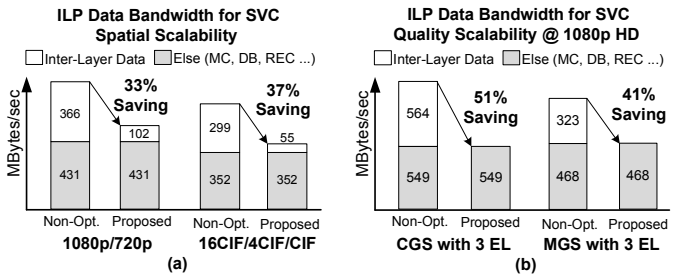

Figure 8. Bandwidth reduction of proposed decoding schemes

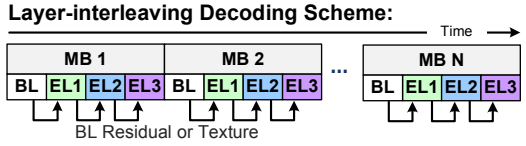

Figure 9. Proposed decoding schedule for SVC quality scalability

easily without the data dependency problem. After all MBs in BL frame are decoded, the padded reconstructed intra-signals are ready in the external memory. When decoding EL, the padded BL intra-signals or BL residual are loaded from the external memory, and these data are on-line upsampled with proper FIR filter according to BL MB modes. The upsampling is executed in the first MB-pipeline stage, and the upsampled data are stored in the on-chip SRAM for EL reconstruction in the second MB-pipeline stage. The proposed decoding scheme only needs to access BL residual and padded BL intra-signals as shown in Fig. 5. It eliminates the bandwidth of type A, B, E, G and H in Fig. 2. The bandwidth reduction of the proposed method is shown in Fig. 8 (a). Furthermore, it saves the processing cycles for padding and upsampling in Fig. 2. In average, 34% of processing cycles are saved.

B. Layer interleaving decoding flow quality scalability

The bandwidth overhead of quality scalability decoding comes from accessing BL coefficients for ILP, as shown in Fig. 4. Fortunately, the frame resolution of BL and EL is identical in quality scalability, so the padding and upsampling procedures are unnecessary. To reduce the decoding bandwidth, so we propose a layer-interleaving decoding scheme for quality scalability as shown in Fig. 9. In this scheme, the SVC decoder decodes MB data of the same location in each layer in an interleaved manner. The ILP coefficients are stored in the on-chip SRAM without accessing the external memory. This method can reduce all ILP bandwidth in SVC quality scalability decoding. As shown in Fig. 8 (b), 51% and 41% of total decoding bandwidth is saved for CGS and MGS respectively.

IV. Proposed Hardware Architecture

In this section, the hardware architectures for low-bandwidth SVC spatial/quality scalability decoding are presented.

A. System architecture

Figure 10 shows the proposed system architecture of a 3-stage MB-pipeline SVC decoder. It comprises entropy decoder, texture decoder, prediction engine, deblocking filter and two new coding tools for SVC, the padding and the upsampling engines. The padding and the upsampling engines lie in the first and third MB-pipeline stage respectively, which is in accordance with the decoding schedule in Fig. 6.

B. Upsampling hardware architecture

Figure 11 shows the proposed upsampling hardware architecture. The upsampling engine is composed of an FIR filter array, a controller and two dual-port SRAMs. To generate the predictors for

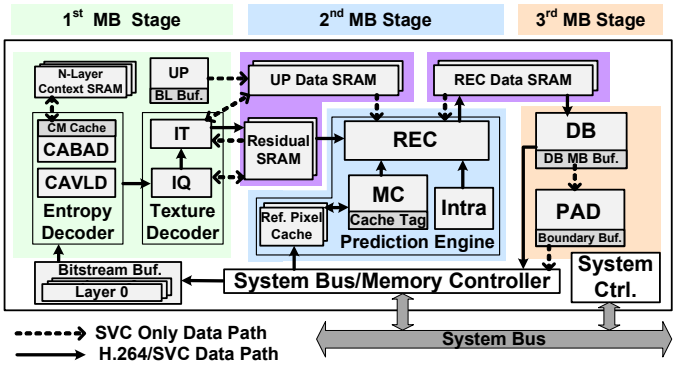

Figure 10. Proposed system architecture of SVC decoder

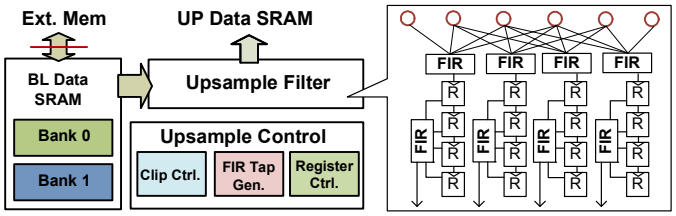

Figure 11. Proposed upsampling hardware architecture

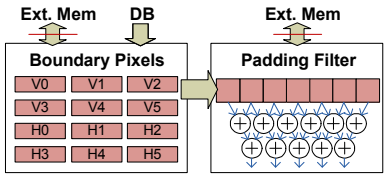

Figure 12. Proposed padding hardware architecture

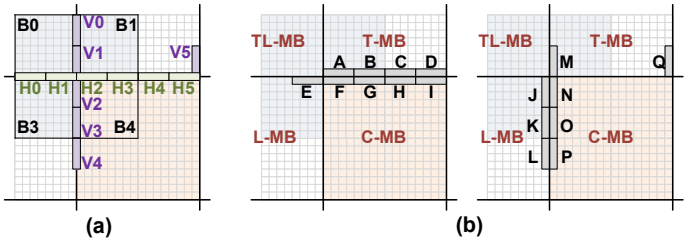

Figure 13. Boundary pixels in padding architecture

ILP, the BL MB modes and corresponding residual or reconstructed intra-signals are loaded from the external memory, and stored in two dual-port SRAMs. Then, filtering is performed on the BL data to generate ILP required predictors. The FIR filter array is composed of multipliers, adders and shift-registers. It acts like half-pixel interpolation in H.264/AVC. After pushing the input pixels into the FIR filter array in 2~4 cycles, four output pixels are generated per cycle. The result of ILIP data needs to be clipped into [0, 255], while ILRP data does not. The FIR tap generator, clipped control and shift-register control are here to provide the proper filter parameters and guarantee the correctness of the interpolation results.

C. Padding Hardware Architecture

Figure 12 shows the proposed padding hardware architecture. The upsampling engine contains twelve 4x1 pixel lines, which are H0 to H5 and V0 to V5 in Fig. 13 (a), and one padding filter. Except V0, all other pixel line data come from deblocking or pipelined data of the previous MB. Since the padding procedure is to generate the

Table 1. Boundary pixel assignment for padding pixel lines.

	MB Mode of {TL,T,L,C} MB. (1: Intra-coded; 0 : Inter-coded)													
	{0,0,0,1}	{0,0,1,0}	{0,0,1,1}	{0,1,0,0}	{0,1,0,1}	{0,1,1,0}	{0,1,1,1}	{1,0,0,0}	{1,0,0,1}	{1,0,1,0}	{1,0,1,1}	{1,1,0,0}	{1,1,0,1}	{1,1,1,0}
V1	-	-	-	M	M	M	M	V5^	V5^	V5^	V5^	M	M	M
V2	N	J	J	-	N	J	J	-	N	J	J	-	N	J
V3	O	K	K	-	O	K	K	-	O	K	K	-	O	K
V4	P	L	L	-	P	L	L	-	P	L	L	-	P	L
V5	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
H0	-	H4^	H4^	-	-	H4^	H4^	H4^	H4^	H4^	H4^	H4^	H4^	H4^
H1	-	E	E	-	-	E	E	H5^	H5^	E	E	H5^	H5^	E
H2	F	-	F	A	F	A	F	-	F	-	F	A	F	A
H3	G	-	G	B	G	B	G	-	G	-	G	B	G	B
H4	H	-	H	C	H	C	H	-	H	-	H	C	H	C
H5	I	-	I	D	I	D	I	-	I	-	I	D	I	D

Note: V0 is always loaded from external memory. The "^" means the pipelined data in previous MB. The "-" means don't care.

Table 2. Gate count distribution of proposed architecture

	Logic Gates	SRAM
Upsampling	14.2K	0.38KB
Padding	6.9K	-
Entropy Decoder Modification	13.3K	0.67KB
System Modification	2.2K	0.77KB
Total	36.6K	1.82KB

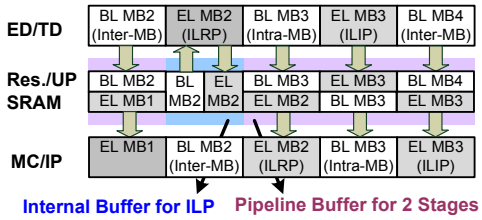

Figure 14. Example of MB-level decoding schedule for SVC quality scalability.

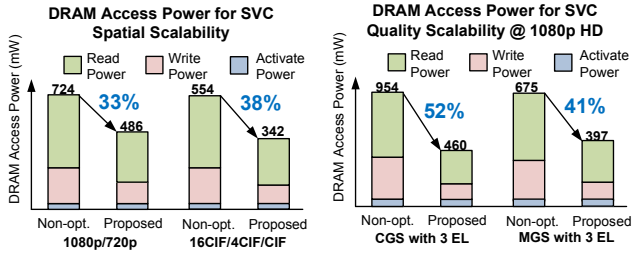

Figure 15. DRAM access power for SVC decoding.

missing intra-signals of the input of ILIP filter through border extension, only the intra-coded block boundary pixels are used as padding inputs. Therefore, the content of the twelve pixel lines depends on the adjacent MBs' modes. For example, if the L-MB is in intra-coded while C-MB is not, the line-L in Fig. 13 (b) is stored in V4. V0 is always loaded from the external memory because the deblocking engine does not have these four pixels. The detail content assignment of the twelve 4x1 pixel lines is list in Table 1. In our proposed padding architecture, only V4 needs to be stored in the external memory as the V0 of the bottom MB. The padding filter acts like intra prediction in H.264/AVC. The padding engine generates the required pixels of the upper-left four 8x8 blocks (B0, B1, B2 and B3). V0, V1, H0, and H1 are the padding inputs of block B0. V0, V1, H2, and H3 are for B1, and so on. All the padded pixels are stored in the external memory as inputs for ILIP upsampling.

D. Hardware Architecture for Quality Scalability

Unlike spatial scalability decoding, no new coding tool is needed in the proposed layer-interleaving decoding scheme. However, other design issues arise in CABAD decoding and on-chip memory usage. The CABAD has to interleaved decode several slices of bit-stream in an interleaved manner for SVC quality scalability decoding in our proposed scheme. Therefore, extra context model cache, context model SRAM and side information buffers for EL are required. Furthermore, the MB-level decoding schedule in the proposed scheme is different from that in Fig. 6. Figure 14 shows the decoding schedule of the first and the second MB-pipeline stage with only one quality EL. The decoder interleaved decodes BL and EL data in each stage. In the proposed system architecture, the pipeline buffers of Residual SRAM, UP Data SRAM, and REC Data SRAM are reconfigurable. They can be configured as pipeline buffers that connect two stages or as the internal buffers in one MB pipeline, as shown in Fig.14. Therefore, no additional buffer is required to store ILP data in the proposed architecture.

V. Implementation Result

Table 2 shows the hardware implementation results. The proposed architecture is designed with Verilog-HDL and synthesized with UMC 90-nm technology. The total gate count is 36.6K with 1.8KB SRAM. The proposed architecture has been integrated into a three MB-pipeline SVC decoder as shown in Fig. 10. The maximum frequency of the SVC decoder is 210MHz. It is capable of decoding a 1080p HD video @ 30 fps with 3 CGS/MGS quality EL under 120 MHz, or a 1080p/720p HD video @ 30fps with spatial scalability under 98 MHz. The proposed architecture can support extended spatial scalability (ESS). As discussed in the previous sections, the proposed decoding schemes and architecture reduce most part of the ILP bandwidth. The benefit of the proposed low bandwidth decoder framework is reflected in low DRAM access power consumption. In our simulation environment, the Micron MT48LC4M32B2 SDRAMs [6] are used as external memory. DRAM access power for SVC decoding is modeled by using [7]. From Fig. 15, 212~494mW DRAM access power is saved. 33~52% of the DRAM access power reduction is achieved (DRAM background power is not included).

VI. Conclusion

In this paper, a low bandwidth decoder framework for SVC is proposed. After analysis of SVC decoding bandwidth overhead, two main decoding schemes are proposed to reduce the decoding bandwidth. MB-level on-the-fly padding and on-line upsampling is proposed to reduce 33~37% of decoding bandwidth and 34% of processing cycles for SVC spatial scalability. For SVC quality scalability, layer-interleaving decoding scheme is proposed to eliminate all ILP bandwidth which is 41~51% of the whole decoding bandwidth. The corresponding hardware architectures of these two decoding schemes are also provided. This low BW framework can reduce 33~52% DRAM access power for SVC decoding.

References

[1] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," IEEE Trans. Circuits Syst. Video Technol., vol. 17, no. 9, pp. 1103–1120, Sep. 2007.

[2] Advanced Video Coding for Generic Audiovisual Services, ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), ITU-T and ISO/IEC JTC 1, Version 8 (including SVC extension): Consented in July 2007.

[3] T.D Chuang, et al., "Bandwidth-efficient cache-based motion compensation architecture with DRAM-friendly data access control," ICASSP 2009, pp. 2009 - 2012, May 2009.

[4] J. Zhu, et al., "An SDRAM controller optimized for high definition video coding application," ISCAS 2008, pp. 3518 - 3521, May 2009.

[5] D. Zhou, et al., "A 1080p@60fps multi-standard video decoder chip designed for power and cost efficiency in a system perspective," VLSI Circuit Symposium, pp. 262–263, Jun. 2009.

[6] Micron SDRAM MT48LC4M32B2 datasheet, www.micron.com

[7] Micron Technology, Inc., SDRAM System-Power Calculator, http://www.micron.com/support/part_info/powercalc