# Multiview Video Coding Optimization using SIMD on Portable Devices

John Mduduzi Mudumbe
Electronics, Information and Communication Engineering
Konkuk University
South Korea, Seoul

James Ntaganda
Electronics, Information and Communication Engineering
Konkuk University
South Korea, Seoul

Cho Yong Beom*
Electronics, Information and Communication Engineering
Konkuk University
South Korea, Seoul

## ABSTRACT

Multimedia support on hand-held devices is growing rapidly. The introduction of 3D videos in cinemas and on home entertainment systems has made it appealing on mobile environment devices. MVC (Multiview Video Coding) provides a compressed representation of multiple views in a single scene. However, handheld devices have hardware constraints that makes it challenging to support high-definition content and increases the complexity of the algorithms used. Due to this, an advanced processor with the capabilities of low power consumption and high instruction computation is needed. Compression standards such as H.264/MPEG-4 and its amendment, AVC (Advanced Video Coding), have had great success in video compression. However, more complex algorithms and significantly higher processing power are necessary. As an extension to this standard, MVC was introduced for 3D stereo video support. Conventional processors contain coprocessors that support Single Instruction and Multiple Data technology. Some arithmetic operations are also supported on the main processor. Therefore these features can reduce the decoding time of the MVC decoding process.

## General Terms

Multiview video coding, Video codec, SIMD, H.264 codec, Decoder, Standards, Optimization

## 1. INTRODUCTION

Advancement of digital media has brought about many changes in our society. The digital information content can be stored, processed and transmitted in many digital file formats using digital systems. Movies can be watched from our mobile devices, surf the Internet and download content with different formats. Recently, there has been a rapidly growing demand for 3D feature films, especially in cinemas and for home entertainment units such as 3D HDTV.

Information formats for audio, text, images, and video can be integrated into a single file or a “stream format”. This stream format enables the user to tailor it to a specific target such as compression, analysis, and decompression. This is made possible with standards from organizations such as ISO and MPEG.

Multiview and 3D videos have high image content that needs to be intensely processed during the compression or decompression of video frames. Processing and transmitting high video content requires extensive processing power and resources. Mobile devices have both memory and battery limitations. Some processors have low power consumption, making them suitable for running complex mobile applications or streaming video [1]. Nevertheless, powerful processers are necessary for real-time decoding in mobile environments as well as good-quality displays.

The new video standards have similar coding algorithms and processes for achieving higher compression rates and processing from hardware compared to previous standards. Considering the need for such processing by systems, the support for real-time video decoding is lacking especially for handheld devices.

The aim of this paper is to incorporate SIMD (Single Instruction, Multiple Data) technology into the existing architecture of processors to optimize the multiview coding decoder. This work mainly focuses on the decoding of MBs (IQ - Inverse Quantization) based on the chroma prediction method, including the quarter-pel filter (on MVC) using SIMD. Section 2 briefly discusses SIMD technology used on coprocessors, Section 3 describes the testing platform constraints, Section 4 elaborates MB decoding, and results of the profiled video are presented on the last section.

## 2. SIMD

Image processing applications consume a lot of memory and the computation requires enormous power. However, most mobile devices are limited in terms of battery duration, processing speed, and memory. ASIC can be seen as a superior option in terms of hardware acceleration and optimization of video programs [2], but it has drawbacks. ASIC is program-specific, and any modification to the program requires the integrated chip to be changed. Additionally, because it is necessary to change the resolution to fit most handheld devices, ASIC designs are rather costly and inefficient.

Video codecs operate on large amounts of data. Generally, 8 bits of data are used. When using a 32-bit microprocessor, during computation, some units are not utilized and these consume power. General processors take three clock cycles to execute a single instruction set (fetching, decoding, and execution) with an exception of processors that makes use of the MIPS (Million Instructions Per Second).

SIMD-architecture-based processors are energy efficient for applications that support easy data parallelization during computation [3]. The design shown in Figure 1 shows the architecture of an ARM A8 coprocessor that supports an SIMD instruction set. The registers are treated as vectors for different data types ranging from 8 × 8-bit-wide vectors to 4 × 16-bit-wide vectors and 32 × 4-bit-wide vectors. Arithmetic operations such as addition, subtraction, bit shifting, multiplication, and division instructions [4] are supported by the coprocessor. SIMD instructions are included in modern processors as they enhance instruction-level parallelism.

SIMD accelerates detection algorithms [5] by scaling down the overall performance of the algorithm. This is achieved by computing matrices in parallel. Using SIMD calls within interpreted languages provides the developer with the freedom to manipulate the internal data without much hardware monitoring even though some compilers allow automated SIMD vectorization.

Most speech recognition systems rely on statistical models such as HMMs (Hidden Markov Models) to "perform acoustic modeling of speech recognition" [6]. To enhance the computation of intensive likelihood-based statistics, SIMD technology is used, and approximately 27% of the total duration is compensated on the LVCSR (Large Vocabulary Continuous Speech Recognition) systems. This is a fair advantage for systems that implement real-time speech recognition, because the power consumption is not exhausted.

## 3. TESTING PLATFORM

The tests discussed in the results section presented further were performed using a Cortex-A8 processor with a 64- or 128-bit configurable bus architecture and a dedicated pipeline to execute SIMD instruction sets. The L1 cache is configurable between 16 KB or 32 KB, and the L2 cache can be configured between 0 KB and 1 MB. The L2 is efficient because it eliminates latency between direct memory access and the ALU (Arithmetic Logic Unit) during the data-fetch phase. The processor also contains a VFP (Vector Floating Point) coprocessor that supports single-precision add, multiply, divide, and square root operations. (More details are provided in the reference manual [7].) The SIMD technology provided on this processor architecture has 16 × 128-bit quad-word registers and 32 × 64-bit double-word registers. It supports 8-, 16-, 32, and 64-bit signed and unsigned integer data types. The architecture includes 32-bit single-precision floating-point elements. Our target system operates at 800 MHz, 512 MB, running a Linux OS release 2.6.32.9. The size of the compiled decoder is 6,546 KB.

The source code for the decoder we used is JMVC software written in C++ and compiled using the ARM tool chain to run on our target system. The software is written for high computing machines. To run the decoder on the target board, a few adjustments had to be made to the source code. The software was optimized by the compiler disregarding SIMD optimization to increase decoding speed and the efficiency of the decoder. The next section discusses the video sequences used and profiles results of the sequences.

## 4. PROPOSED WORK

Real-time encoding/decoding is of paramount importance because the temporal and spatial resolutions of the views increase. Most researchers working on optimization of the codecs based on H.264 have shown that the motion-compensation block consumes more than half the total required time for total decoding [8] [9] [10]. Hence, research being conducted is based on finding techniques and algorithms to reduce data computation regarding the MC (motion compensation) on the H.264/AVC standard [9] [11], including MP (Motion Prediction) and ME (Motion Estimation).

When optimizing software, profiling needs to be performed. Because MVC is backward compatible with H.264/MPEG-4 AVC, decoding singleview videos is carried out first during profiling. Figure 2 depicts the payloads of modules used by the decoder and their total contribution in MCPS (Million of Cycles Per Second). The testing sequences are in singleview form. The MC module (Motion Compensation) consumes approximately 34–36%; the DB (Deblocking) filter consumes an average of 9–16%; and Entropy (uVLC-Universal Variable Length Coding) consume 6–8% the fewer contributing modules are inverse transform and Reconstruction of frames. MC profiling was conducted on the granule level and Table 1 shows the innermost modules that are also costly in terms of processing.

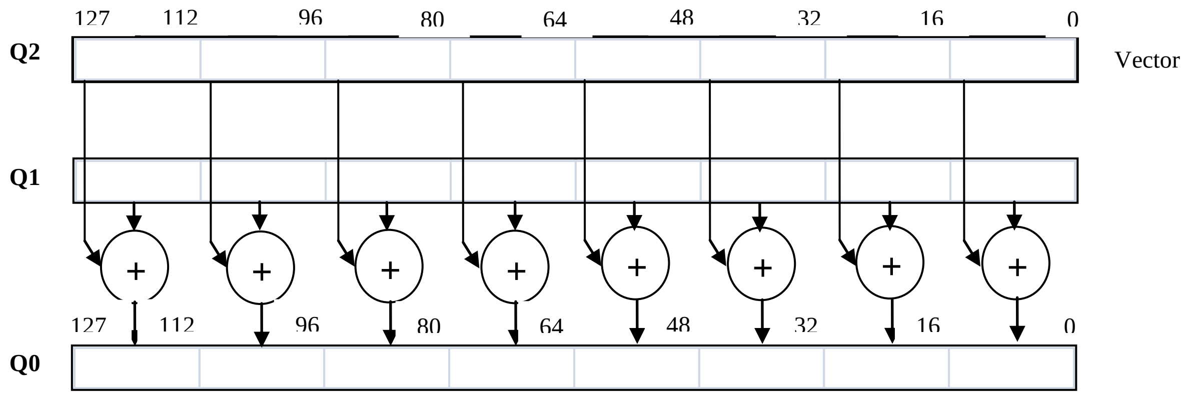

**Figure 1: Two 128-bit-long vectors Q1 and Q2**

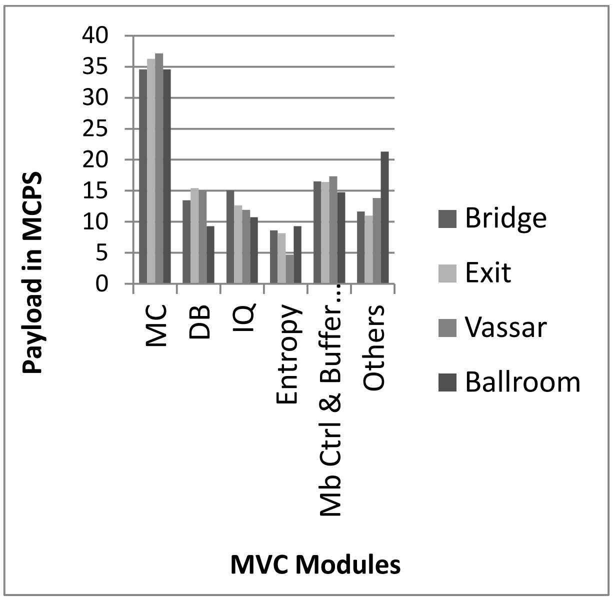

**Figure 2: Profile of singleview video sequences**

**Table 1: Motion Compensation sub-module performance (MCPS)**

| | Chroma and Luma Prediction | Quarter-pel filter |
|---|---|---|
| Bridge | 9.76 | 9.6 |
| Exit | 13.08 | 9.33 |
| Vassar | 11.75 | 6.05 |
| ballroom | 10.34 | 7.75 |

Because of monotonousness routine executed on each MB (either on 16 × 16, 16 × 8, 8 × 16, also known as sub macro blocks 8 × 8, 8 × 4, 4 × 8 and 4 × 4) modes. The computation is redundant until the entire frame is reconstructed. By applying SIMD intrinsic built-ins to modules that significantly contribute to the MC payload, the MC (Intraprediction and Quarter-pel filter), Deblocking filter and the IQ payload can be reduced. In this article, we propose the use of SIMD technology to reduce the total time necessary to compute chroma intra-prediction during decoding of frames on multi-view videos.

## 5. MOTION COMPENSATION (MVC)

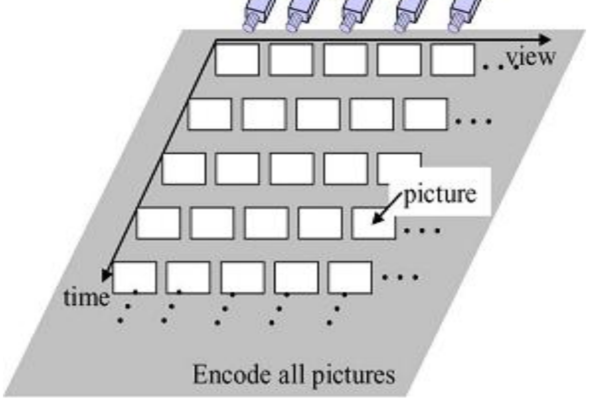

**Figure 3: Multiview videos captured using five cameras**

Two or more cameras capturing the same scene would result in a large amount of redundant data from common frames from adjacent cameras or frames within the same view. The MVC scheme exploits such temporal redundancies between successive images in each video, whereas the inter-view exploits the similarities between adjacent camera views [12]. Although significant improvements have been made, more research needs to be conducted because the goal of MVC is to support 3D video applications such as teleconferencing and enable viewers with free-viewpoint videos. The aim of MVC was to enable viewers to use the free-viewpoint option in real time [13], where the viewpoint and view direction can be changed interactively.

Recent solutions such as fast search algorithms within frames were introduced [14], including the fast mode search during the motion estimation execution to improve the prediction precision and the efficiency of coding base on multi-block and multi-reference frames [15] by provided great enhancement to the standard. To calculate vector prediction, the MBs of the inter-modes or intra-modes are assigned weights according to the cost incurred during motion estimation; the greater the cost, the more accurate the block is considered to be, and the more likely it is for it to be usable for estimation of the next sequence frame.

The decoder can be configured as a main or baseline profile for achieving a coding gain at the cost of computational power. The encoder configuration during the encoding of the sequences was set to three reference frames. Because MC is performed on inter-block coding, MBs are divided into squares with a minimum size of 4 × 4 pixels. To obtain the random access point of the views when decoded, the pictures contain intra-coded frames that are not encoded with prediction from other frames. Regardless of the type of the s*lice (I, P,* or *B),* the MBs in the slice have to be decoded with data on each header of the slice. This paper focuses on computing data in parallel on the computation of chroma and luma prediction during the reconstruction of the MB [16] IQ (Inverse Quantization). The two components are calculated separately because the chroma and luma predictions are independent of each other in intra-frame prediction. Each component is based on the mode selected, and the component can be decoded for horizontal and vertical prediction.

**Table 2: Chroma and Luma Predictions and Quarter-pel Filter (MCPS)**

| | Chroma and Luma Prediction | Quarter-pel filter |
|---|---|---|
| Bridge (3 views) | 12.27 | 6.63 |
| Exit (3 views) | 12.38 | 10.5 |
| Vassar (8 views) | 12.48 | 8.29 |
| Ballroom (8 views) | 10.51 | 10.35 |

The rescaling of MBs with data from the slice header during the reconstruction of the frame is redundant on all MBs. A typical 4 × 4 MB reconstruction would loop four times. However, using the SIMD given the data type, be it 16 or 32 bits, the computation can be executed without looping. The same applies for the 8 × 8 or 16 × 16 MB scaling, which can be achieved efficiently using a NEON instruction set. IQ reconstructs the MB using the already mapped coefficients during the encoding period.

$$Pred_C[x,y] = \alpha \cdot Rec_L'[x,y] + \beta$$

Figure 4 shows part of our implementation of the SIMD instruction set (VMUL, VPADD, and VSLHQ). The execution time was greatly reduced on the chroma prediction module because the sub modules have 8- and 16-bit-long data. This implies that we can multiply a vector of unsigned 16 × 4 bits by a 16-bit scalar, and the result would be stored in a 32 × 4-bit-long vector.

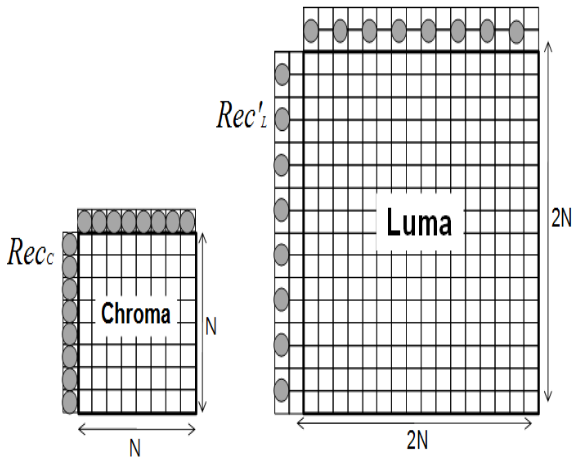

**Figure 4: Chroma prediction, reconstruction of MB**

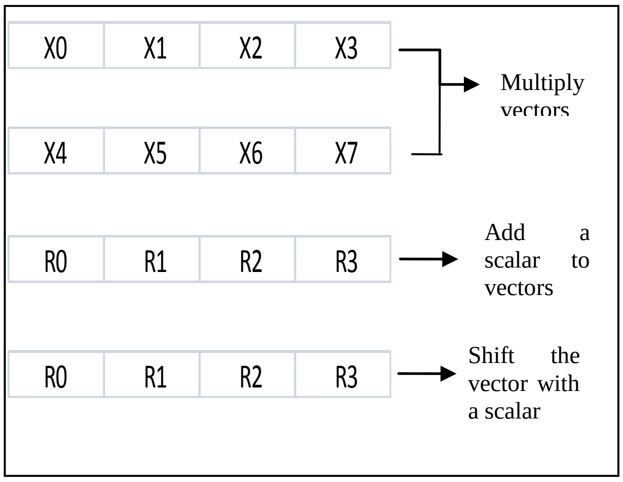

**Figure 5: Implementation of Rescaling modules**

## 6. RESULTS

We used the JMVC 8.5 (open-source code) decoder, optimized for C++ on the target system mentioned in Section 3. An ARM tool chain supplied along with d-stream as a cross compiling tool for the decoder was used. To enable optimization for using the SIMD instruction set, the ARM tool chain provides the necessary libraries needed for using intrinsic functions.

Figure 6 displays the profiled video sequences using only three views; Figure 7 displays the profiled video sequences using eight different views. The difference between the two figures is the Recon module and DB filter payloads. The three video sequences (Bridge, Exit, Vassar, and Ballroom) are found in the JVT documentation and on the Merl Website [17] [14]. The video sizes are 176 × 144 (~QCIF), and each of the video was encoded at a rate of 30 fps. The number of reference pictures was set to three. Fast search mode, bi-prediction iteration with four iterations for all the video sequences, and Context-adaptive Variable Length Coding (CAVLC) were used for entropy coding. The number of encoded frames was 280 per video sequence, and the GOP (Group Size of Pictures) was set as three. Figure 2 shows the profiling results for single view videos.

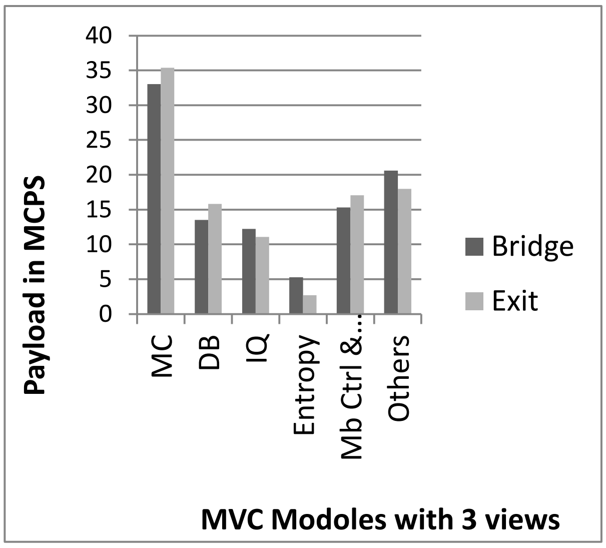

**Figure 6: Videos of Bridge and Exit sequences using three views**

Table 3 summarizes the overall performance achieved using SIMD operations. The chroma prediction and Quarter-pel filter summed up on MC submodule was reduces nearly half of the initial overall payload when optimization was implemented.

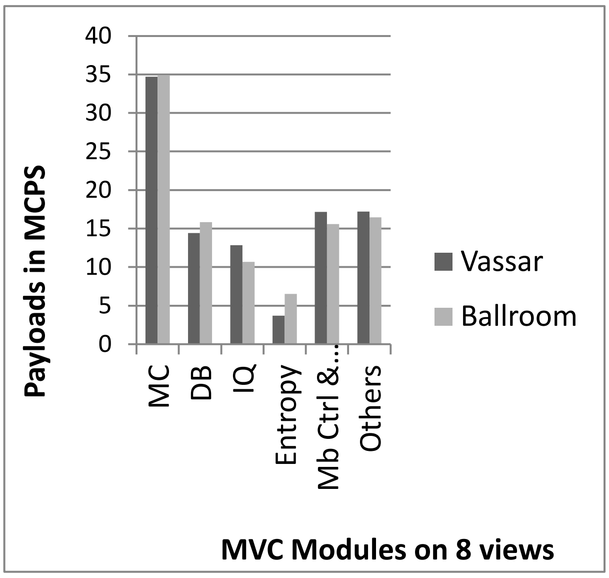

**Figure 7: Videos of Vassar and Ballroom sequences using eight views**

Vectorization is necessary in order to perform much optimization using SIMD instruction set. In future research, we will focus on the power consumption when the MVC decoder is running and the differences when the optimized decoder is executed as well.

It is notable that more complex algorithms that can be easily broken into vectors can be processed by processors with less computational power, as they can be processed without much hardware constraints. This is because the coprocessor uses the same hardware features as the main processor and has precedence over the L2 cache memory hence does not depend on the main processor. Macro block control and Buffer load modules are pretty expensive, but these modules are directly linked to memory access for acquiring buffer during decoding time, initializing the bitstream, controlling the reference frames, managing the POC order and managing the number of slices in a group. The systems memory delay time, affects this module directly. These modules payloads can be noted on figure 2,6 and 7.

**Table 3: Module Performance achieved using SIMD**

| Video Sequences | MC – Submodules | IQ | Deblocking filter |
|---|---|---|---|
| Bridge (3 views) | 6.41 | 9.72 | 6.74 |
| Exit (3 views) | 8.48 | 10.64 | 8.48 |
| Vassar (8 views) | 6.7 | 9.07 | 6.7 |
| Ballroom (8 views) | 7.66 | 4.68 | 7.66 |

## 7. CONCLUSION

Simultaneous parallel execution of data is an advantage to image processing or video compression/decompression using MVC because a large amount of data can be processed without delay or much processing power.

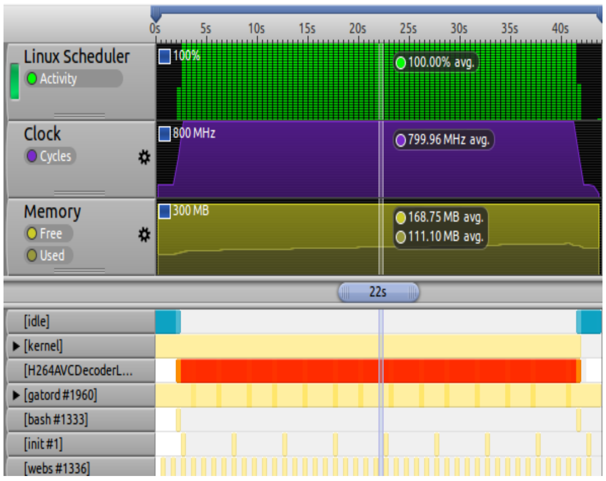

**Figure 8: MVC (not optimized) streamline analysis**

Fugures 6 and 7, depicts the IQ module consuming about 12 -15 % of the total payload on the decoder. This was reduced by 30-35 %. A conclusion can be made from these results, that co-processors are efficient and offer acceleration to multimedia applications running portable devices. The implementation is made on the source code and suitable features required on the processor should be enabled during run time.

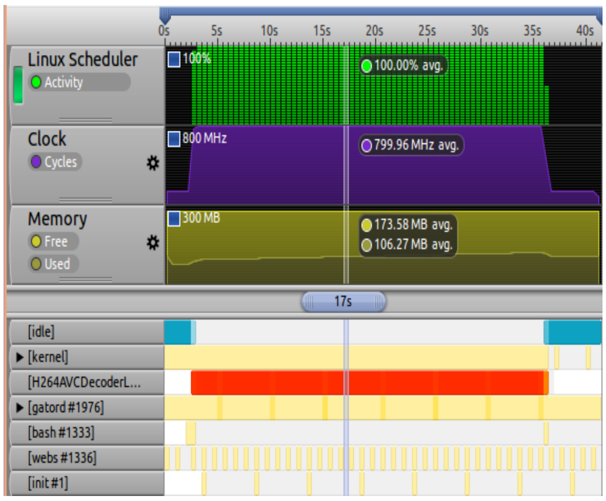

**Figure 9: MVC (optimized) streamline analysis**

Figures 8 and 9 show the profiled results of the decoder on the target system in terms of memory, system clock cycles, and scheduler. The system does not require or use additional resources compared to when only the main processor is running. Thus, the amount of memory and the frequency remain the same throughout the duration of decoding.

## REFERENCES

[1] Joon-Ho Song; Won Chang Lee; Doo Hyun Kim; Do-Hyung Kim; Shihwa Lee, "Low-power video decoding system using a reconfigurable processor," *Consumer Electronics (ICCE), 2012 IEEE International Conference on* , vol., no., pp.532,533, 13-16 Jan. 2012

[2] Liwei Chen, Ming Cong, Jing Huang, Ling Li, Hongwei Liu, Cheng Qian, "A Novel HW/SW Partitioning with SIMD Instructions for AVS Video Decoder," *nas, pp.273-277, 2012 IEEE Seventh International Conference on Networking, Architecture, and Storage*, 2012

[3] Sangwon Seo; Dreslinski, R.G.; Woh, M.; Yongjun Park; Charkrabari, C.; Mahlke, S.; Blaauw, D; Mudge, T., "Process variation in near-threshold wide SIMD architectures," *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE* , vol., no., pp.980,987, 3-7 June 2012

[4] ARM 2009 .introducing NEON - Development Article. ARM Ltd.

[5] Yen-Kuang Chen; Li, E.; Xiaofeng Tong, "Parallelization of AdaBoost algorithm on multi-core processors," *Signal Processing Systems, 2008. SiPS 2008. IEEE Workshop on* , vol., no., pp.275,280, 8-10 Oct. 2008

[6] J. Ou1, J. Cai1 and Q. Lin, “Using SIMD technology to speed up the likelihood computation in HMM-based speech recognition systems”, *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on* , vol., no., pp.123,127, 2008.

[7] ARM Cortex-A8 technical reference manual. [online]. Available: http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0473e/Bcfjicfj.html

[8] Pujara, C.; Modi, A.; Sandeep, G.; Inamdar, S.; Kolavil, D.; Tholath, V., "H.264 Video Decoder Optimization on ARM Cortex-A8 with NEON," *India Conference (INDICON), 2009 Annual IEEE* , vol., no., pp.1,4, 18-20 Dec. 2009 J. Ahn, S. Song and K. Kim, “Implementation of h.264 fractional motion estimation using full search algorithm”, IEEE, 2009.

[9] Jingyu Ahn; Sehyun Song; Kichul Kim, "Implementation of H.264 Fractional Motion Estimation using full search algorithm," *SoC Design Conference (ISOCC), 2009 International* , vol., no., pp.357,360, 22-24 Nov. 2009

[10] Kondo, S.; Sasai, H., "A motion compensation technique using sliced blocks in hybrid video coding," *Image Processing, 2005. ICIP 2005. IEEE International Conference on* , vol.2, no., pp.II,305-8, 11-14 Sept. 2005.

[11] A. Barjatya, “Block matching algorithms for motion estimation”, DIP 6620 Spring 2004 Final Project Paper , 2004.

[12] Vetro, A.; Wiegand, T.; Sullivan, G.J., "Overview of the Stereo and Multiview Video Coding Extensions of the H.264/MPEG-4 AVC Standard," *Proceedings of the IEEE* , vol.99, no.4, pp.626,642, April 2011.

[13] A. Smolic and P. Kauff, "Interactive 3-D video representation and coding technologies", *Proceedings of the IEEE*, vol. 93, no. 1, pp 98-110, Jan. 2005.

[14] Peng Yin; Tourapis, H.-Y.C.; Tourapis, A.M.; Boyce, J., "Fast mode decision and motion estimation for JVT/H.264," *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on* , vol.3, no., pp.III,853-6 vol.2, 14-17 Sept. 2003.

[15] Yansong Cui; Zhongliang Deng; Weizheng Ren, "A Fast Search Algorithm with Multi-Block Mode and Multi-Reference Frame," *Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on* , vol., no., pp.1,4, 24-26 Sept. 2009.

[16] Merkle, P.; Mǘller, K.; Smolic, A.; Wiegand, T., "Efficient Compression of Multi-View Video Exploiting Inter-View Dependencies Based on H.264/MPEG4-AVC," *Multimedia and Expo, 2006 IEEE International Conference on*, vol., no., pp.1717,1720, 9-12 July 2006.

[17] Y. Su, A. Vetro and A. Smolic, Common test conditions for multiview video coding, ITU-T JVTU211, 2007.