TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol.12, No.6, June 2014, pp. 4393 ~ 4399
DOI: 10.11591/telkomnika.v12i6.5473

Real-time Implementation of a xAVS Video Decoder

Qing Chang, Xin Liu, YaoLi Wang*
College of Information Engineering, Taiyuan University of Technology,
Taiyuan, 030024, China, Ph./Fax: +0086-3516014052/3516014052
*Corresponding author, e-mail: willingwan@gmail.com

Abstract

AVS video decoder consumes a huge number of computation, so real-time implementation of an AVS decoder has some challenging on x86 computing platform. This article describes a xAVS open source project to solve this problem. First of all, the reason for the low efficiency of the code of the existing AVS video decoder open source reference software RM52J_r1 is analyzed according to the description of AVS key technologies and decoding principles in the official documentation. Then according to the main problems of the reference software, re-design the optimized xAVS decoder architecture, and real-time property significantly improved with C code. Finally, use the x86 platform multimedia instruction sets to further optimize xAVS semantic processor. The experimental results show that, under the precondition of ensuring the quality of decoding, the decoding speed of the xAVS decoder for D1 has increased more than 14 times, to fully meet the needs of real-time decoding.

Keywords: *xAVS, decoder, real-time decoding*

1. Introduction

AVS is independently developed by Chinese as the second generation source coding standard. AVS video codec performance is 2 to 3 times higher than MPEG-2, and quite well with H.264 [1-4], but the implementation complexity is lower than the H.264. In order to verify the correctness of Algorithm, AVS working group submitted the reference software RM52J_r1, but RM52J_r1 was not optimized for a platform. The decoding speed of RM52J_r1 is less than satisfactory, and the decoding rate is only 3 to 5 per second in the current high-end x86 kernel, so there is a big gap for practical applications. The task of implementing AVS algorithms on the x86 platform was proposed by AVS Workgroup, that is xAVS open source project. Our task force assumed the optimization of xAVS decoder.

This paper first introduces the structure and characteristics of AVS, then according to the deficiencies of AVS reference software, proposes the solution of xAVS decoder, which designs a new decoding process and program structure, and achieves the goal with C code. By optimization of the multimedia instruction set, the D1 decoding speed is more than 50 frames per second, so it meets the real-time requirements.

2. Decoding Principles of AVS

AVS decoding algorithm structure is shown in Figure 1.

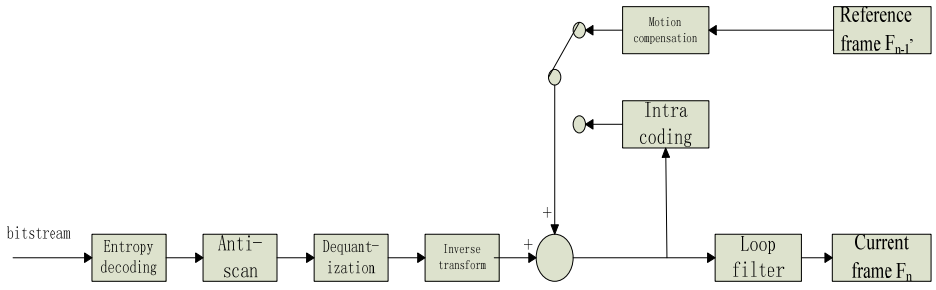

Figure 1. Structure of AVS Decoder

Received December 21, 2013; Revised February 10, 2014; Accepted February 22, 2014

The decoding process is: the received bit stream which is compressively encoded enters the input terminal of the decoder, and according to the entropy decoding it can obtain a series of parameters. On the one hand, the quantized coefficients can be obtained through anti-scanning the parameters, after de-quantization and inverse transformation it can obtain the residual coefficients.

On the other hand, according to the received prediction mode and data the decoder obtains the predicted values of intra prediction or inter prediction, the reconstructed values are the sum of these predicted values and the residual coefficient. Finally, though loop filter these reconstructed values can get the final image values.

3. The Design of xAVS Decoder

The decoding reference software RM52J_r1 which was released by the official have already realized.

All syntax elements of the AVS standard and basic semantics, but the efficiency of the code is still very low. Analysis of the reasons are [5-7]: the structure of reference code is irrational, so there is a large number of unnecessary criteria and jump; the reference code exists complex multi-cycle, and there are multi-level nested calls between function; variable definitions are not uniform, too many duplicate definition; the dynamic memory allocation was used repeatedly, the CPU resources was consumed unreasonably, so the efficiency of the memory is low. For the main problem of RM52J_r1, this paper presents a new design of xAVS decoder.

Table 1. DTC-CSF System

Parameter	value
Proportional gain, Kp	36
Integral gain, Ki	12643
Flux hysteresis band	0.01 Wb
Sampling frequency	40kHz
Switching frequency	4kHz

Table 2. Induction Machine Parameters

Parameter	value
Stator resistance	5.5 Ω
Rotor resistance	4.51 Ω
Stator self inductance	306.5 mH
Rotor self inductance	306.5 mH
Mutual inductance	291.9 mH
Momen of inertia	0.01 kg.m^2
Number of poles	4
Rated speed	1410 rpm
DC-link voltage	565 V

3.1. xAVS Process

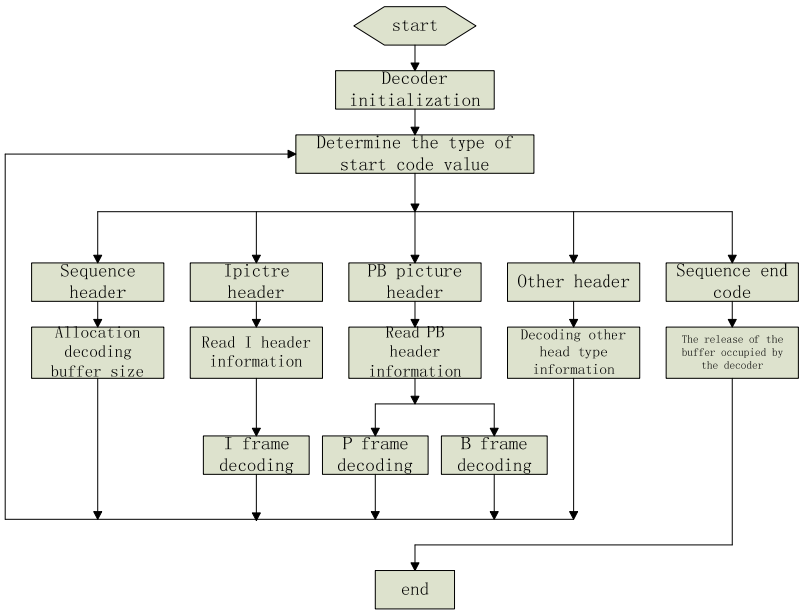

Figure 2. The Decoding Process of xAVS

The decoding process of xAVS is shown in Figure 2. The image data is divided into some frames. One frame is a slice. Every slice has a different start code. First the start code of image header information is analysed, the slice type can be got by the start code. Then the decoder chooses different decoding process by the slice type. When parsing out the start code is I, P, or B picture header, we decode them respectively.

3.2. Module Optimization

AVS uses block-based hybrid coding framework. Its core technologies include [8-10] entropy decoding, inverse quantization and inverse transformation, intra prediction, inter prediction, and loop filter.

1) Entropy decoding

According to the syntax element [11-14], entropy decoding can generate Level array and Run array. Level array contains the amplitude of the non-zero quantized coefficients, run array contains the number of consecutive zero before the current non-zero quantized coefficients. The process of entropy decoding is that, First determine if the encoded data of the current 8x8 block exists or not, If it exists, initialize the mapping table and jump threshold, otherwise exit. Then trans_coefficient is parsed, if trans_coefficient is less than 59, quantized coefficients and run can be get by lookup CurrentVLCTable, So Level array and Run array can be got .If trans_coefficient is bigger than or equal to 59, escape_level_diff should be parsed out, Level arrays and Run array can be calculated by escape_level_diff. If trans_coefficient equal to EOB, then the end of the block coefficient decoding. Finally, according to absLevel to update the number of the next stopwatch.

From the process of entropy decoding, if trans_coefficient less than 59, Level and Run are used in the CurrentVLCTable and escape_level_diff isn't used. If trans_coefficient is bigger than or equal to 59, only escape_level_diff is used. The case of less than 59 is used in the majority of the entropy decoded; in most cases escape_level_diff of CurrentVLCTable is superfluous. Thus two structures are defined when we design code table in xAVS decoder. Structure contains Level and Run will be used in the case of less than 59,another structure which only contains escape_level_diff will be used in other cases. Two tables are used for storing a VLC code table in xAVS decoder.

When according absLevel value to update the serial number of a stopwatch, RM52J_r1 uses a series of judgment statement, so increasing the running time of the code. xAVS decoder uses the method of looking-up table instead of conditional judgment. The switching rules of code table are shown in Table 3 to Table 5.

Table 3. xAVS Intra Luma Table Switching Rules

absLevel	0	1	2	3	4	5	6	7	8	9	10	>10
table_num	0	1	2	3	3	4	4	4	5	5	5	6

Table 4. xAVS Inter Luma Table Switching Rules

absLevel	0	1	2	3	4	5	6	7	8	9	>9
table_num	0	1	2	3	4	4	4	5	5	5	6

Table 5. xAVS Intra Luma Table Switching Rules

absLevel	0	1	2	3	4	>4
table_num	0	1	2	3	3	4

2) Inverse quantization and Inverse transformation

The integer inverse quantization and inverse transformation of 8x8 block are used in AVS. Dimensional integer inverse transformation can be decomposed into horizontal and vertical one-dimensional integer inverse transformation. xAVS and RM52J_r1 are basically the same, but xAVS has all zero block judgment before the inverse transformation.When the current 8x8 sub-block is the all-zero block, this block doesn't exist inverse transformation, because after inverse transformation these block are still the all-zero block.

3) Intra prediction

Intra prediction includes determining the prediction mode of every 8x8 sub-block, getting reference sample value and calculating predicted values. In RM52J_r1, It firstly obtains and saves the reference pixel value, and then determines a prediction mode to predict. The processes of xAVS is different from RM52J_r1. First it determines the prediction mode of each 8x8 sub-block, then reads the corresponding pixel values according to the prediction model, so many unnecessary reference pixel values can be avoided.

4) Inter prediction

In AVS, P or B frame has two reference frames at most, P frame can refer to the most forward nearest decoded I frame or P frame ,B frame can refer to a front and rear of the most nearest decoded I frame or P frame. In xAVS the macroblock decoding of P and B frame are similar to I-frame, also according to the type of the macroblock to decode. First, obtain the corresponding reference index of 8x8 sub-block. Next, based on the motion vector of the adjacent block to obtain a prediction motion vector of the current block. reconstruct them to get motion vector of the current block. The corresponding reference samples can be got through the motion vector and the reference index. If the reference sample are not in the integer pixel location, so these reference samples should be interpolated to obtain the predicted values.

In order to optimize the efficiency of code execution, the inter prediction of xAVS has a different design ideas from RM52J_r1.

Firstly, in RM52J_r1, calloc function is called at the start of an image decoding to dynamically allocate memory. After the end of an image decoding release the memory, so the efficiency of this memory is very low. In xAVS decoder, the dynamic memory allocation of reference frame are put before decoding the entire video sequence, and these memory are released after the end of the entire video sequence. Secondly, in the decoding process of each frame, residual data, the predictive value and the reconstructed value are stored by a one-dimensional array, the dimension of the array are reduced. Third, the P frame or B frame can have up to two reference frames, therefore xAVS assigns three frame buffers to store the information of the currently decoded frame as well as the two reference frames. And expands boundary of frame buffer, thus it's better for interpolation optimization.

During the interpolation process, if the reference integer sample is out of the reference image, instead it with the nearest integer sample (edge or corner of the sample) which is nearest to the reference sample that is motion vector can point to samples which are out of a reference image [15-22]. So during the interpolation process, every reference pixels should be judged whether is outside the boundary of the reference image. The max and min function are used in reference software, so that each 8x8 sub-block needs 64 judgments, the efficiency of the program is reduced. There is boundary extension of frame buffer space in xAVS decoder, the nearest image boundary pixel will instead of the pixel which is beyond the image boundary. So during the interpolation process, the reference pixel doesn't judge whether it is out of bounds or not.

5) loop filter

There are three filtering modes: strong filtering (BS=2), standard filtering (BS=1) and non-filter (BS=0). In this article, the design idea of xAVS decoder is that, after the end of one image decoding, all the macroblocks of one frame are filtered with a raster scanning method cycle. The BS of every 8x8 macroblock should be got firstly, and then filter the luma and chroa boundary. If the current decoding images is I frame, then BS is equal to 2,so the calculation of the filter function is omitted. For P and B frame, there is no vertical boundary filtering of 16x16 and 16x8 macroblocks, and no horizontal boundary filtering of 16x16 and 8x16 macroblocks. When these conditions are met, the corresponding BS don't need to calculate. For filtered pixels which chroma block needs is less than luma block, so the chroma filter function and chroma filtering function was divided in xAVS decoder. After such optimization, the loop filter module saves a lot of calculation in xAVS decoder, so the decoding speed is improved.

3.3. SIMD Optimization

SIMD is a CPU execution mode which is single-instruction and execute multi-channel data for x86 platform, the plurality of elements to be processed in a parallel manner, to improve the speed of the program. In this paper, xAVS decoder was optimized with MMX and SSE2.mainly optimize for inverse transformation, interpolation, and loop filter module which include a large amount of calculation.

Horizontal inverse transformation of inverse transformation can be optimized with group sum, group subtraction as well as group shift of MMX, the matrix transposition is a critical part [23-25]. MMX instructions can only handle four 16-bit or two 32-bit data, so the inverse transformation module was achieved with SSE2 [6]. SSE2 is based on 128-bit registers, it's easier to implement inverse transformation and matrix transposition.

For example, in order to obtain one-half of the b example, this paper descript how to use the MMX to achieve the optimization of the interpolation module. By filtering around the horizontal direction on the four integer point One-half of the example b obtains an intermediate valueb1=(-C+5D+5E-F), predictive value b=max(0,min(255,(b1+4)>>3)) is obtained by the intermediate valueb1.The main instructions are movd, pshufw, punpcklbw, jnz, paddw, packuswb. The function of packuswb is limiting calculated predictive value from 0 to 255.The realization of the process is shown in Figure 3.

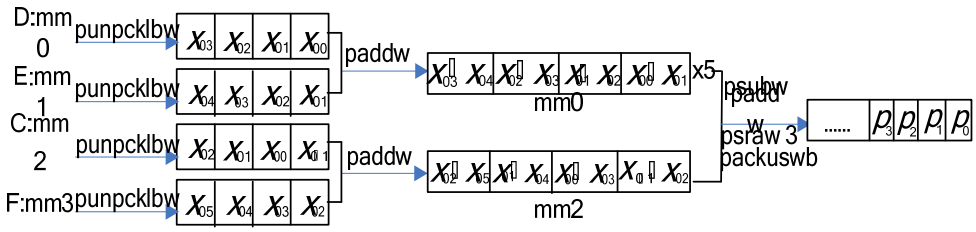

Figure 3. Schematic Diagram of Calculating the Half Interpolation

The optimization of the loop filter is achieved by SSE2.For different filer strength, the horizontal filter and vertical filtering are written separately. The array which is read into the register 128 in the vertical filtering is shown in Figure 4.In the operation of the registers, the data of the registers are right of Figure 4 .Therefore, in the vertical filter matrix transpose should be operated firstly, in order to achieve a parallel data processing.

The system is fully optimized in accordance with the above methods, and the effect is desirable.

xmm0: a0 a1 a2 a3		a0 b0 c0 d0
xmm1: b0 b1 b2 b3	→	a1 b1 c1 d1
xmm2: c0 c1 c2 c3		a2 b2 c2 d2
xmm3: d0 d1 d2 d3		a3 b3 c3 d3

Figure 4. Register Array of Vertical Filter

4. Experimental Results

The test environment of xAVS decoder is Windows7, Pentium dual-core processor, 1.8GHz, compiled and debugging in Visual Studio 2008.The frame order is IBBPBBP. AVS test stream of 30000 frame including the type of QCIF (176x144), CIF (352x288) and D1 (720x576) are test, and compared with RM52J_r1 decoder. The test results of frame rate is shown in Table 6, and the test results of PSNR is shown in Table 7. From Table 6 and Table 7, xAVS decoder and RM52J_r1 decoder have same PSNR, the decoding speed of xAVS decoder is greater than RM52J_r1 decoder. The decoding speed of QCIF, CIF and D1 is respectively 22.14 times, 16.26 times and 14.29 times than RM52J_r1 decoder.

Table 6. Frame Rate Test Results Contrast of xAVS Decoder and RM52J_r1 Decoder

Test Sequence	Frames	Decoding speed(fps)	
		RM52J_r1	xAVS
akiyo. qcif	30000	57.91	1282.05
foreman. cif	30000	13.58	220.75
SOCCER.D1	30000	3.54	50.59

Table 7. PSNR Test Results Contrast of xAVS Decoder and RM52J_r1 Decoder

Test Sequence	RM52J_r1			xAVS		
	PSNR_Y	PSNR_U	PSNR_V	PSNR_Y	PSNR_U	PSNR_V
akiyo. qcif	38.02	42.24	43.21	38.02	42.24	43.21
foreman. cif	34.75	41.30	42.93	34.75	41.30	42.93
SOCCER.D1	34.53	42.21	44.03	34.53	42.21	44.03

5. Conclusion

In the foundation of analyzing and researching RM52J_r1, this paper puts forward optimized improvement programs and realizes a new xAVS decoder. In the premise of maintaining the original image quality, decoding speed has been greatly improved. xAVS decoder meets the requirements of real-time decoding.

Acknowledgements

It is a project supported by Shanxi National Natural Science Foundations (No. 2013011015-1).

References

[1] Gao Wen, Wang Qiang. Digital Audio Video Coding Standard of AVS. *ZTE Communications*. 2006; 12(3): 6-9,13.
[2] Gong Xiaoxia, Liu Xingwang. An achievement of the full-I-frame real- time decoding on PC platform. *Sciencepaper Online*. http://www.paper.edu.cn /index.php/default/ releasepaper/content/200911-162.
[3] Li Yan, Liu Xiulan. Bottleneck analysis and optimization of AVS software decoder. *Electronic Measurement Technology.* 2010; 33(4): 128-130.
[4] Audio Video coding Standard Workgroup of China.GB/T20090.2-2006. *Information technology-Advanced coding of audio and video-Part2: Video*. 2006.
[5] H Malvar, A Hallapuro, M Karczewicz, L Kerofsky. Low-Complexity transform and quantization in H.264/AVC. *IEEE Trans.Circuits Syst.Video Technol*. 2003; 13: 598-603.
[6] Wei Fang, Li Xueming. SIMD Optimization of Transform and Ouantization in H.264. *Computer Engineering and Applications*. 2004; 17: 24-27.
[7] Martuza Muhammad, Wahid Khan A. Implementation of a cost-shared transform architecture for multiple video coder. *Journal of Real-Time Image Processing*. 2012; 1-12.
[8] Lv Qian, Cui Ligong. FPGA verification for the OR1200 subsystem in AVS-SoC. *Advanced Materials Research(ICEM 2012)*. 2013; 651: 807-811.
[9] Hu Qian, Zhang Ke, Yu Lu. Decoder architecture and hardware implementation for AVS-video. *Journal of Zhejiang University (Engineering Science)*. 2006; 40(12): 2139-2143.
[10] Wang Fei, Li Yuan, Jia Huizhu, Xie Xiaodong, Gao Wen. *An efficient fractional motion estimation architecture for AVS real-time full HD video encoder*. Proceedings of IEEE International Conference on Imaging Systems and Techniques(IST 2012 - 2012). 2012; 279-284.
[11] Ding Dan-dan. Research on Reconfigurable Video Coding. *Journal of Zhejiang University (Engineering Science)*, 2011.
[12] Dandan Ding, Honggang Qi, Lu Yu, Wen Gao. Reconfigurable Video Coding Framwork and Decoder Reconfigurable Instantiation of AVS. *Signal Processing: Image Commun*. 2009; 24(4): 287-299.
[13] DING Dan-dan, Yu Lu. Overview of MPEG Reconfigurable Video Coding. *Video Engineering*. 2009; 33(7): 12-15.
[14] WANG Wen-xiang, SHEN Hai-hua. A Reconfigurable Sub-Pixel Interpolation Architecture Design for Multi-standard Video Decoding. *Journal of Computer-Aided Design & Computer Graphics*. 2011; 23(9): 1603-1613.
[15] DU Juan, DING Dan-dan, YU Lu. Design methodology of FPGA based reconfigurable video encoder. *Journal of Zhejiang University (Engineering Science)*. 2012; 46(5): 905-911.
[16] ZHAO Jing, ZHOU Li, YU Qing-dong, et al. Research of audio video standard (AVS) HD decoding based on a reconfigurable processor. *Journal of Harbin Engineering University*. 2012; 33(2): 226-233.
[17] Wang Jin, Yuan Bin, Zhang Gang. The design of I frame based on the x264 architecture. *Taiyuan University of Technology*. 2010; 41(2): 139-141, 146.
[18] Bai Yuting. Hardware implement of the key modules in AVS encoder. *Taiyuan University of Technology*. 2012.
[19] Xiang Hongli. FPGA implementation of prediction module for AVS encoder. *Taiyuan University of Technology*. 2012.
[20] LI Fujiang. The research of key algorithms for real time avs encode and the ASIC design based on cloud on chip. *Taiyuan University of Technology*. 2011.

[21] LI Fujiang, Zhang Gang. Quick AVS intra prediction mode selection algorithm. *Computer Engineering and Applications*. 2010; 46(2): 1-3.
[22] Jiang Tao, Zhou Peihai. MIN of Bahadur KC. Intra prediction in H.264 and AVS dual-mode video decoder hardware design and implementation. *Application of Electronic Technique*. 2007; (9): 41-44.
[23] Zhao Longhui, Chen Xinhua, Ren Huai Lu. AVS-based entropy decoder design. *ITS APPLICATIONS*. 2010; (20): 53-55 ,58.
[24] Yi Bo-nian. Turbo Code Design and Implementation of High-Speed Parallel Decoder. *TELKOMNIKA Indonesia Journal of Electrical Engineering*. 2013; 11(4): 2116-2123.
[25] Yu Fan, Yang Huijing, Li Gang. A High Performance Sigma-Delta ADC for Audio Decoder Chip. *TELKOMNIKA Indonesia Journal of Electrical Engineering*. 2013; 11(11): 6570-6576.