Advanced Engineering Forum *Online: 2012-09-26*
ISSN: 2234-991X, Vols. 6-7, pp 410-415
doi:10.4028/www.scientific.net/AEF.6-7.410

A Real-time H.264 BP Decoder Based on ADSP-BF548

Chi-Wei Tung, Chou-Chen Wang, Wan-Ying Jhuang, Yi-Chieh Tsai

Department of Electronic Engineering, I-Shou University, Kaohsiung, Taiwan, R.O.C.

E-mail: chchwang@isu.edu.tw

Keywords: Video coding, H.264, Embedded system

Abstract. In this paper, we propose an embedded real-time H.264 Baseline Profile (BP) decoder based on ADSP-BF548 Blackfin processor. In order to achieve the real-time requirement of H.264 decoding, we modify and optimize the decoding modules and codes, respectively. Firstly, we analyze the number of operations for various modules using assembly code, and then use direct memory access (DMA) to carry out the parallelism between algorithm execution and data movement. Finally, we make use of two buffer groups (BG) as parallel decoding mechanism of broadcast transformation. Experimental results demonstrate that the play rate can reach above 25 fps as using QCIF video. According to some tests, a real-time H.264 BP decoding can be achieved with a 600 MHz DSP.

Introduction

Mobile multimedia communication typically involves the transfer of large amounts of data. Therefore, an efficient compression of video data is essential for a cost-efficient use of existing communication channels and storage media. Up-to-date video coding standards have been developed for different types of applications. H.264 is the recent ITU-T video coding standard, also defined by ISO as MPEG-4 part10 [1]. The H.264 standard is very suitable for multimedia applications including mobile video communications, video conferences and HDTV etc. In these applications, H.264 Baseline Profile (BP) is used mainly for mobile video communications. However, H.264 BP video decoding requires significant amount of computational power.

Non-programmable decoders offer enough computational power at a low cost but cannot cope with the quick evolution of the video decoding algorithms. On the other hand, the latest generation of digital signal processors (DSPs) [2-4] can support very flexible decoders like the multi-format one proposed in [5] at a relative low cost. As a result, optimization of applications to the maximum possible extent is necessary. With these figures, a real-time H.264 BP decoder based on DSP is very worthy of developing.

In this paper, the implementation of a real-time H.264 BP decoder based on ADSP-BF548 processor is described. In section 2, an overview of the ADSP-BF548 architecture is outlined. Section 3 explains the MPEG-4 decoder implementation architecture. Section 4 describes the tests results. Finally, section 5 is devoted to the conclusions.

The architecture of ADSP-BF548

A simplified block diagram of the DSP internal architecture is shown in Fig. 1. The CPU is a Blackin processor with a performance e of up to 4800@600 MHz. ADSP-BF548 [6] is a 16/32-bit processor and gives high performance for low power consumption, which makes it attractive for embedded applications. A 192 KB internal SRAM can be divided into 64 KB level-1 cache for program (L1P) and 64 KB for data (L1D). A leve-2 cache (L2) unified for data/program is 128 KB. A 64 MB external memory DDRRAM level-3 (L3) can be accessed through a dedicated interface using a 64-bit data interface. The other peripherals are a dynamic memory access (DMA) controller, video ports, an Ethernet port (EMAC), an output audio interface and several general-purpose I/O pins. The DMA controller allows moving data between memory and peripherals.

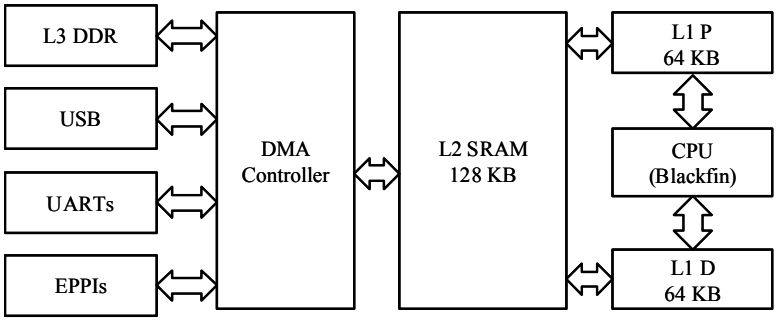

Fig. 1 The architecture of ADSP-BF548.

Fig. 2 The EZ-KIT Lite.

ADSP-BF548 is based on RISC principles, and the normalized instruction set and related decode mechanism are much simpler. This simplicity results in a high instruction throughput and impressive real-time interrupt response from a small and cost-effective chip. Pipelining is employed so that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory. The memory interface has been designed to allow the performance potential to be realized without incurring high costs in the memory system. Unlike traditional video embedded solutions that utilize two processor cores to provide video functionality, the ADSP-BF548 provides a convergent solution in a unified core architecture that allows voice and video signal processing concurrent with RISC MCU processing to handle network and user-interface demands. This unique ability to offer full functionality on a single convergent processor provides for a unified software development environment, faster system debugging and deployment, and lower overall system cost. The ADSP-BF548 is mostly used as a controller core rather than for data processing. But by intelligent usage of the features provided, it can be easily adapted for the video decoding process. The ADSP-BF548 EZ-KIT Lite as shown in Fig. 2 provides a cost-effective method for initial evaluation USB-based and PC-hosted tool set.

H.264 Decoder Implementation

The decoder implements BP of H.264 video coding standard [6-7]. The starting point was a standard compliant C decoder fully tested first in a PC environment and moved o the DSP environment afterwards. This initial code was optimized to increase the execution speed in about two orders of magnitude.

H.264 Decoding Process. H.264 decoding procedure is shown in Fig. 3. H.264 decoding procedure consists of bit stream parsing including network adaptation layers (NAL) unit, entropy decoding using CABAC or CAVLC [1], reoder, inverse quantization (IQ), inverse integer cosine transform (IICT), motion compensation (MC), intra prediction, deblocking filter (DF) and reconstruction of video. Table 1 gives the H.264 BP information of the decoding process. The most consuming processes of decoder include IICT, MC, entropy decoding and reconstruction of video as summarized in Table 1. It is evident from Table 1 that the IICT module occupies most time in decoder. Therefore, there will be considerable improvement in performance if the IICT can be optimized to the maximum possible extent.

IICT module is generally applied to a block of 8×8 pixels by using Chen's algorithm [7]. There is high spatial and temporal correlation when video sequence is applied to multimedia applications, this will result in very few significant coefficients (non-zero coefficients) after ICT while encoding.

Table 1 Gives the BP profile information of the decoding process.

Module Name	Time Consume in Module
Inverse Integer Cosine Transform +Motion Compensation	47.66%
Deblocking Filter	25.8%
Entropy Decoding	13.58%
others	12.96%

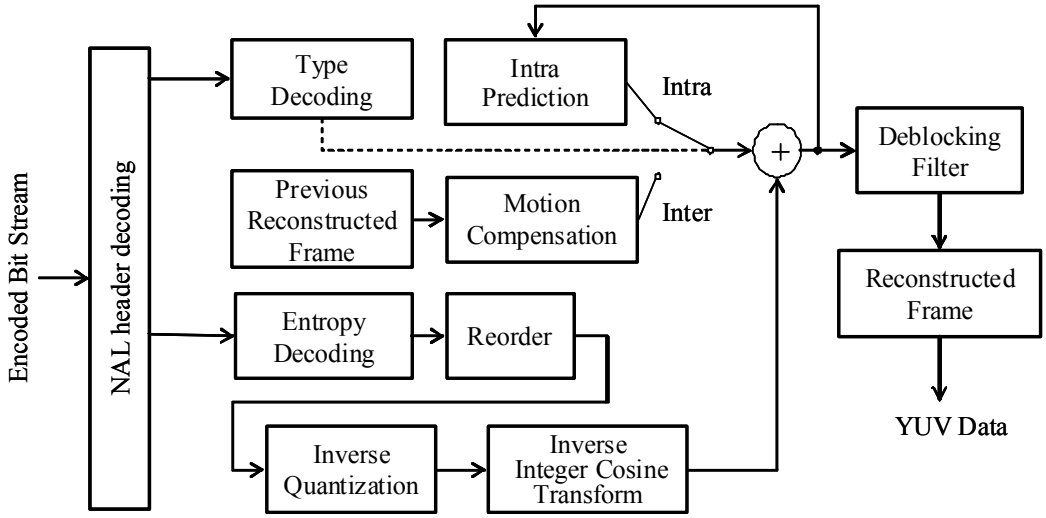

Fig. 3 H.264 decoding procedure.

The number of non-zero coefficients reduces further after quantization. Instead of generalized 8×8 IICT, depending on the number of non-zero coefficients, IICT can be applied to only the significant valued coefficients. This optimizes IICT to a great extent since default 8×8 IICT will involve unnecessary multiplications and additions for the zero valued coefficients. In order to further to increase the decoding performance, a novel deblocking filer technique which uses only the internal memory proposed in [10] is implemented in our study.

Use of Memory. The DSP memory usage in the decoder for processing data in ADSP-BF548 is optimally allocated according the proposed structure. The reference pictures and MB for deblocking filter are stored in internal memory L2 to speedup the decoding procedure. REF, ICT_COEFFS, IQ, DC/AC_COEFFS and REC buffers are located in internal memory L1D (internal SRAM is configured as 128 KB). The assigned addresses for L1 and L2 are from 0xffb00000 to 0xff800000 and from 0xfeb00000 to 0xfeb20000, respectively.

Parallelization. The conditional video playing process in the embedded system adopts the sequential playing model (SPM) as shown in Fig. 4. This model decodes bitstream and sequentially shows frames step by step. However, this will lead to frame delay when the H.264 decoded video is applied. To overcome the problem of frame delay to occur playing lag, we use a parallel architecture to move data and decoding operations. The ADSP-548 uses the parallel peripheral interface (PPI) to play the decoded video. In addition, we use the TV encoder to play the ITU-656 video format. To complete the playing process, we need to convert the YUV into ITU-656 since the decoded H.264 video is the YUV format.

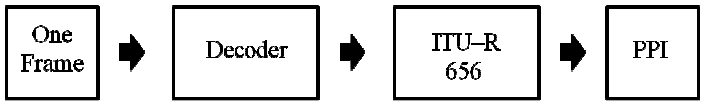

Fig. 4 The sequential playing model.

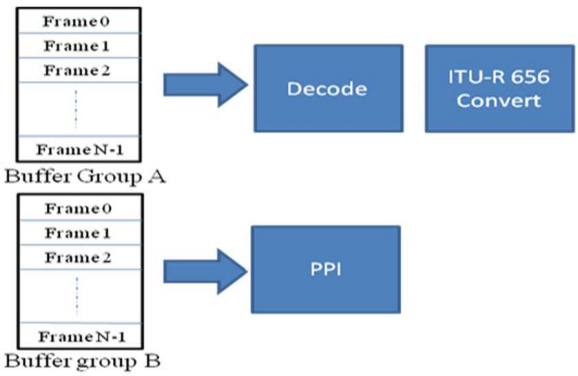

Fig. 5 The proposed two BG buffer structure.

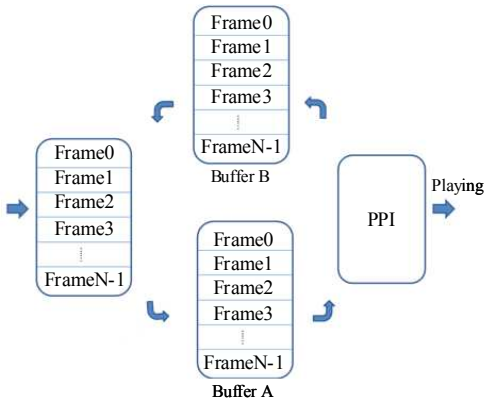

Fig. 6 The proposed parallel playing model.

In order to achieve a real-time decoding and playing, we further design two buffer group (BG) for parallel processing to carry out the playing machine. The BG consists of some frame buffer (FB) as shown in Fig. 5. Each FB stores a frame and each BG can store N-1 frames. The initial value of *N* is set to 16. The proposed two BG buffer structure is shown in Fig.56. The buffer group A firstly receives and stores the decoded frame, and then converts them into the ITU-656 frame. The buffer group B takes charge of displaying the decoded frame after transferred data for ITU-656 over PPI using DMA with descriptor list mode. It only needs one interrupt to finish the convert and PPI by DMA. Therefore, we can achieve the parallelization which the DSP implements the decoding procedure and DMA work on a ping-pong the convert and PPI process. Figure 6 shows the proposed parallel playing model (PPM) for Ping-Pong buffer structure.

Since the ADSP-BF548 only has 16 bits data bus, it is inefficient as increasing the numbers of BG. Therefore, we adopt two BG modules to improve the problem of sequential playing model. To achieve the real-time decoding for H.264 BP, we further use the SDRAM controller function in Blackfin to increase the executing efficiency of memory access.

Code Optimization. In order to reach real-time operation, additional optimization steps [9] have been carried out as follows:

1. The higher computational cost functions have been moved to internal memory.
2. Also frequently accessed data have been moved internal memory.
3. Frequent arithmetic operations have been coded using intrinsic (pseudo-assembler) instructions.
4. The core of the IQ, IICT, VLD and MC has been also coded intrinsic.

Test Results

Some tests have been carried out using test QCIF (176×144) video sequences including “Forman”, “Carphone” and “Test” as shown in Fig. 7. The “Test” video sequence is actually taken from the Digital Video (DV). The test stream is stored in a file and read at a picture basis and written in a

(a) Forman (b) Carphone (c) Test

Fig. 7 Test video sequences.

Table 2 Summarizes the average playing rate for SPM and proposed PPM using QCIF sequences

Video sequence	Playing Scheme	No. of FB	Average playing rate (fps)
Foreman (QCIF)	SPM	0	7 fps
	Proposed PPM	16	27 fps
Carphone (QCIF)	SPM	0	8 fps
	Proposed PPM	16	28 fps
Test (QCIF)	SPM	0	6 fps
	Proposed PPM	16	26 fps

stream buffer allocated in external memory. The decoder reads the stream from this memory decodes a picture and writes it in a picture buffer. The numbers of FB is set to $N = 16$.

A PC running Visual DSP++ v5.0 has been used to carry out the simulation profiles and implemented on the ADSP-BF548 EZ-KIT Lite. Table 2 summarizes the average playing rate for SPM and proposed PPM using QCIF sequences mentioned above. It is clear that the proposed PPM method can reach a real-time playing. There is a very distinct increasing of playing rate when using the proposed PPM when compared with general SPM method. For the CIF video sequence, it still needs to improve the playing rate due to the limitation of memory.

In addition, the related results are given in average core cycles per frame for the full decoder. The ADSP-BF548 is an embedded processor with 600 MHz core cycles. Figure 8 gives the MHz counts for the processes subjected to optimization in the video decoder and player. These results give the number of processor cycles taken by each of the processes for decoding one second of video comparisons of core cycles using the SPM and the proposed PPM. Fig. 8 shows the results of Test video sequence. We can find that the proposed PPM consumes less cycle to achieve a mostly real-time decoding and playing. It is worthy to sacrifice some core cycle as shown in the points in a multiple of $N = 16$ to wait the data transfer of BG after finishing the video playing. In addition, Fig. 8 also demonstrates that the proposed embedded scheme can decode a CIF video which reduces approximately 50.2 MHz core cycles.

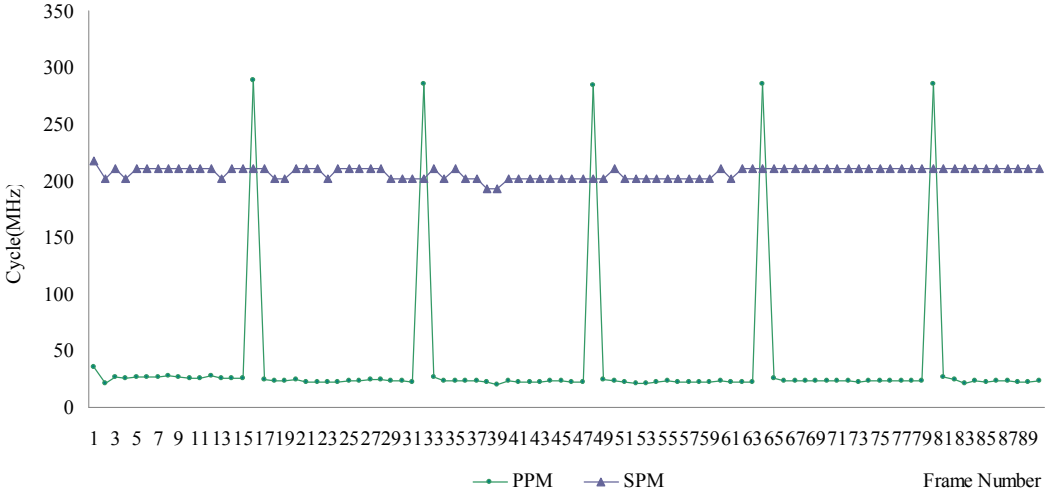

Fig. 8 The results of Test video sequence.

Conclusions

In this paper, we propose an implementation of optimized H.264 BP video decoder based on an ADSP-BF548 processor for real-time performance. Significant reduction in 50.2 MHz is obtained after implementing all the optimization techniques. The decoded QCIF frame playing rate can increase up to 25 fps when applied the PPI procedure. The playing rate can reach above 30 fps as using QCIF video so that the proposed method can achieve a real-time decoder and player.

Acknowledgement

This work was supported by the National Science Council, Taiwan, R.O.C., under Contract NSC 100-2221-E-214-048.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits and Systems for Video Technology, vol. 13 (2003), no. 7, p. 560

[2] Philips Semiconductors. Nexperia Media Processors. Available online at http://www.nxp.com/products/nexperia/home/products/mediaprocessors/index.html

[3] Analog Devices. Blackfin processors. Available online at: http://www.analog.com/processors/blackfin/index.html

[4] Texas Instruments. C6000 DSPs. Available online at: http://focus.ti.com/paramsearch/docs/parametricsearch.tsp?family=dsp&sectionId=2&tabId=1941&familyId=1398

[5] Y. S. Tung et al.: "DSP-Based Multi-Format Video Decoding Engine for Media Adapter Applications", IEEE Trans. on Consumer Electronics, vol. 51 (2005), p. 273

[6] "ADSP-BF548 EZ-KIT Lite Evaluation System Manual," Revision 1.3, October (2008).

[7] D. LeGall: "MPEG: A Video Compression Standard for Multimedia Applications," Communications of ACM, vol. 34 (1991), no. 4, p. 46

[8] D. Katz, C. Lam and R. Gentile: "Blackfin Processor's Parallel Peripheral Interface Simplifies LCD Connection in Portable Multimedia," Analog Dialogue 39-01, January (2005).

[9] F. Pescador et al.: "A DSP Based IP Set-Top Box for Home Entertainment". IEEE Transactions on Consumer Electronics, Vol. 52 (2006), p.254

[10] F. Pescador et al.: "A DSP Based H.264/AVC Decoder for Multimedia Terminal". IEEE Transactions on Consumer Electronics, Vol. 57 (2011), p.705