Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

An efficient hardware design for HDTV H.264/AVC encoder*

Liang WEI[†1,2], Dan-dan DING[1,2], Juan DU[1,2], Bin-bin YU[1,2], Lu YU[†‡1,2]

([1]*Institute of Information and Communication Engineering, Zhejiang University, Hangzhou 310027, China*)

([2]*Zhejiang Provincial Key Laboratory of Information Network Technology, Hangzhou 310027, China*)

[†]E-mail: weiliang33@126.com; yul@zju.edu.cn

Received June 16, 2010; Revision accepted Feb. 17, 2011; Crosschecked May 5, 2011

Abstract: This paper presents a hardware efficient high definition television (HDTV) encoder for H.264/AVC. We use a two-level mode decision (MD) mechanism to reduce the complexity and maintain the performance, and design a sharable architecture for normal mode fractional motion estimation (NFME), special mode fractional motion estimation (SFME), and luma motion compensation (LMC), to decrease the hardware cost. Based on these technologies, we adopt a four-stage macro-block pipeline scheme using an efficient memory management strategy for the system, which greatly reduces on-chip memory and bandwidth requirements. The proposed encoder uses about 1126k gates with an average Bjontegaard-Delta peak signal-to-noise ratio (BD-PSNR) decrease of 0.5 dB, compared with JM15.0. It can fully satisfy the real-time video encoding for 1080p@30 frames/s of H.264/AVC high profile.

Key words: H.264/AVC, High-definition television (HDTV), Hardware, Architecture, Encoder
doi:10.1631/jzus.C1000201 **Document code:** A **CLC number:** TN919.8

1 Introduction

In the past decade, high resolution video contents have widely prevailed from the Internet to broadcasting. Digital television (DTV) technologies with high-definition (HD) resolution not only provide broadcasting services but also try to expand their feasibility towards mobile market (Lee *et al.*, 2008). With the increase in multimedia market size, there is a greater demand for more efficient HD-oriented technologies for various video contents.

The latest video coding standard H.264/AVC (ITU-T, 2005) plays an important role in this area. It outperforms all the previous standards, with at least 2x compression improvement (Wiegand *et al.*, 2003), and is widely used in various multimedia systems. Similar to the previous H.26*x* and MPEG-*x* standards, block-based hybrid coding frameworks, including spatial and temporal prediction, residual transformation, quantization, and entropy encoding, are adopted in H.264/AVC to remove data redundancy.

To improve the coding efficiency, H.264/AVC adopts many new technologies, such as variable block size matching, motion vector prediction, and multiple frames for reference. Moreover, in high profile, adaptive block size transformation, weighted quantization, and context-based adaptive binary arithmetic coding are used (ITU-T, 2005). These new technologies, however, demand huge logic resources and high data dependence in hardware implementation.

It is well known that in the encoding process, motion estimation (ME) occupies most of the computational complexity and the reconstruction loop restricts the critical length of the pipeline. For HD realization, memory bandwidth becomes another bottleneck of the whole system because of a larger search window (Liu *et al.*, 2009). Many approaches have been proposed for HD applications (Yang *et al.*, 2006; Chang *et al.*, 2007; Chen *et al.*, 2008; Lin *et al.*, 2008a; Park and Lee, 2008) in which some were

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (No. 61076021) and the Program for New Century Excellent Talents in Universities, China

concerned only with optimal design of modules or part of the encoder, and some others were targeted at the encoder chip, which is unsuitable for 1080p implementation. In the architecture proposed by Lin *et al.* (2008b), fast algorithms are adopted for both integer and fractional motion estimation. However, the data flow of the system is so irregular that it can hardly be extended for future work, and the data requirement for off-chip memory is also increased. In contrast, Chen *et al.* (2008) targeted good performance and thus most definitely led to huge costs of logic resources and high memory requirements, which could not satisfy resource-constrained implementations.

In this paper, we propose an efficient hardware-oriented real-time encoder for 1080p H.264/AVC, which is highly suitable for resource-constrained realization. The system adopts a four-stage macroblock (MB) pipeline architecture. A two-level mode decision mechanism is employed to reduce the complexity with reasonable coding performance decrease compared with the reference software JM15.0. Furthermore, a resource-sharable architecture and a data reusable strategy with flexible bus tasks management are proposed for high reusability of logic resources and on-chip memory.

2 The proposed encoder system

2.1 Design constraints and optimization

Our highest specification is HDTV 1080p (1920×1080@30 frames/s) video encoder for H.264/AVC high profile, which has at least 4x higher complexity than 720p baseline (Lin *et al*., 2008b). Therefore, there exist many new challenges.

Bandwidth requirement: Since the throughput of MB processed is 8192 MBs/frame, which is more than 245k MBs/s, the memory bandwidth and hardware cost required will be greatly increased. Considering 266 MHz DDR SDRAM (64 bits in width) is used as external memory, the data access should be no more than 8.6η KB per MB in which η is the bus efficiency. In general, η cannot be 100% in practice; we assume η=70% optimistically. Hence, the data processed are limited to less than about 6 KB/MB, and an efficient data reuse strategy must be designed to meet the requirements (details will be illustrated in Section 3.3).

Computational complexity: There are abundant modes in H.264/AVC; that is, 17 different modes for intra-prediction while 259 kinds of partitions for inter-prediction (Chen *et al.*, 2006). To reduce the computational complexity, a fast algorithm should be introduced and reusability should be considered.

The crucial path and pipeline: The coding path of an MB is very long, including intra/inter prediction, reconstruction, entropy coding, and in-loop deblocking filter. Thus, this process should be partitioned into several stages working in pipeline to shorten the crucial path and enhance throughput. For example, in mode decision, the motion vector difference (MVD) cost of the current block depends on the motion vectors (MVs) of the neighboring blocks. This means that, the mode of the current block cannot be derived until we obtain the MVs of the neighboring blocks. This procedure will become a long coding path. Thus, we introduce a fast algorithm for mode decision to eliminate the data dependency. Details of the techniques to shorten the crucial path will be introduced in Section 3.1.

Considerations for future work: In this paper, we target a resource and memory bandwidth constrained H.264/AVC encoder. Extensibility and flexibility are considered in the design to support more formats like the Audio Video Standard (AVS) in the future. For example, the sub-pixel interpolation filters applied in fractional motion estimation (FME) are absolutely different between H.264 and AVS, leading to extra costs for supporting AVS. Other differences such as intra prediction, transform, and entropy coding (EC) should also be taken into account. Hence, more flexibility and extensibility are required.

2.2 The proposed encoder architecture and pipeline scheme

The proposed encoder system is shown in Fig. 1. Fig. 2 illustrates the four-stage pipeline schedule and Table 1 lists the cycle counts for different modules. Especially for EC (CABAC, context-adaptive binary arithmetic coding), the number of cycles required varies greatly according to the features of MBs, and the throughput of CABAC is about 1 bin/cycle, which is sufficient. Compared with other HD-oriented four-stage systems (Huang *et al.*, 2005; Chen *et al.*, 2006), we apply a two-level mode decision (MD) mechanism. Furthermore, we fully exploit the similarity among modules of normal mode fractional

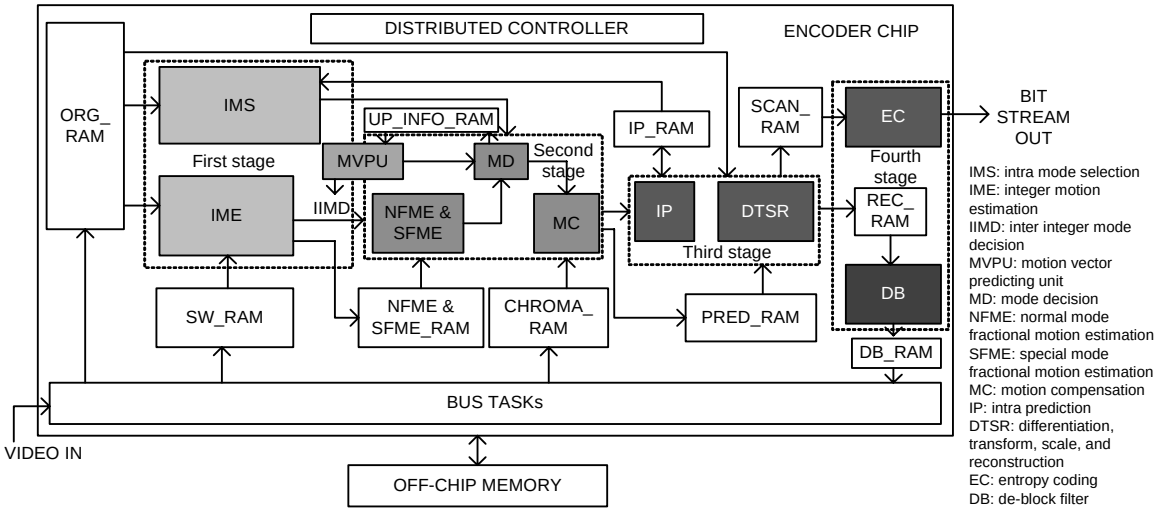

Fig. 1 The proposed hardware oriented H.264/AVC HDTV encoder system

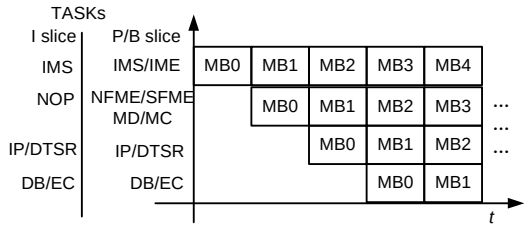

Fig. 2 The proposed macro-block pipeline schedule
The full names for the tasks are as described in Fig. 1

Table 1 Cycle counts for the different modules in Fig. 1

Module	Cycle count (cycle/MB)		
	Best case	Worst case	Average
IME	1040	1040	1040
IMS	960	960	960
NFME	88×2	160×2	<320
SFME	88×2	88×2	<176
MC	44×2	160×2+80×2	<480
IP/DTSR	900	1100	980
DB	670	900	<900

The full names for the modules are as described in Fig. 1

motion estimation (NFME), special mode fractional motion estimation (SFME), and motion compensation (MC), and arrange them in the same pipeline stage; this can reduce resource costs and share on-chip memory efficiently.

In the first stage, intra mode selection (IMS), which is aimed to select the best intra mode, and integer motion estimation (IME) with inter integer mode decision (IIMD), which is aimed to select the best inter mode of integer pixel accuracy, are processed in parallel to save cycles, since their architectures are difficult to share.

In the second stage, for I slice it is a NOP operation; for P/B slice, NFME, SFME for skip and direct modes, MD, and MC, including luma and chroma parts, are processed. Note that the motion vector predicting unit (MVPU) here is time-division multiplexing; it works in both the first and the second stages. In the first stage, it predicts MVs for IIMD to calculate the approximate MVD cost. In the second stage, it predicts the precise MVs to obtain MVDs transferred in bit stream.

In the third stage, intra prediction (IP) and DTSR, including differentiation, transform, scale, and reconstruction, share the whole clock cycle.

EC and de-block filter (DB) are parallel arranged in the fourth stage.

3 The proposed algorithm and architecture

3.1 Two-level mode decision mechanism

In H.264/AVC, the block size of motion estimation ranges from 4×4 to 16×16 (ITU-T, 2005), as shown in Fig. 3. Traditionally, there are 41 integer MVs output from IME, which means each partition has its best candidate MVs. Thus, NFME needs to obtain the block that most matches in the accuracy of quarter pixel for the 41 integer MVs, which will occupy either a great number of clock cycles or a large quantity of logic resources.

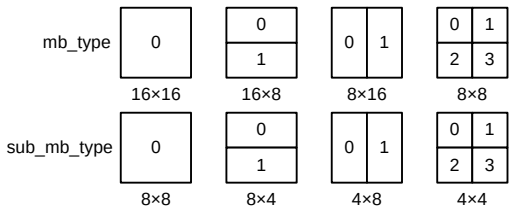

Fig. 3 Macro-block partitions in H.264/AVC

To obtain high coding efficiency, we need the block that most matches. Meanwhile, the computational complexity should be considered to avoid large hardware costs. In the proposed system, a two-level MD mechanism is designed (Fig. 4). At the first level, the best intra mode is selected by IMS, and IIMD selects the best inter mode of integer pixel accuracy, which determines the best MB partition and the best integer MVs. At the second level, first NFME refines the accuracy of the best integer MVs to quarter pixel based on the best MB partition, and then the MD unit compares the best intra, best inter, as well as skip/direct modes, and selects the final mode. Based on this algorithm, the FME is simplified and the pipeline structure is optimized.

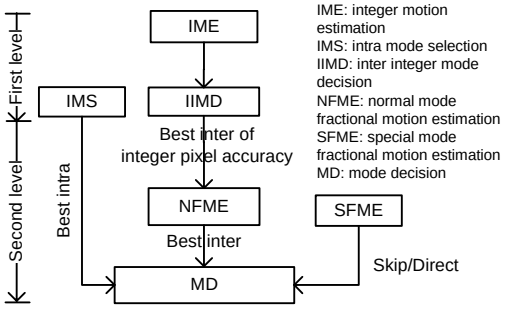

Fig. 4 The proposed two-level mode decision mechanism

Using the proposed method, the intra prediction process is divided into IMS and IP. The IP module requires fewer cycles since the best intra mode has already been selected by IMS and will not be started until the final mode is intra. Then, cycles of the third stage are almost consumed by the DTSR module, which acts the role of the reconstruction loop and restricts the pipeline length.

In H.264/AVC, the predicted motion vector (PMV) of the current block E is obtained by the neighboring blocks A, B, C, and D (Fig. 5a). Since we arrange the pipeline at MB level, and the final mode decision is made in the second stage, when IIMD starts, the final mode of the current MB has not yet been determined, which means that the neighboring blocks A, B, C, and D might be unavailable. Therefore, in the proposed architecture, we simplify the PMV algorithm to adapt to the MB pipelined system. As in Fig. 5b, we predict the MV of the current block E using blocks A′, B′, C′, and D′ in neighboring MBs instead of A, B, C, and D. Therefore, an approximated PMV is used to obtain the MVDs for IIMD. However, the MVDs that will be transferred in bit stream are accurately calculated after final mode decision is made, to make the encoder and decoder match.

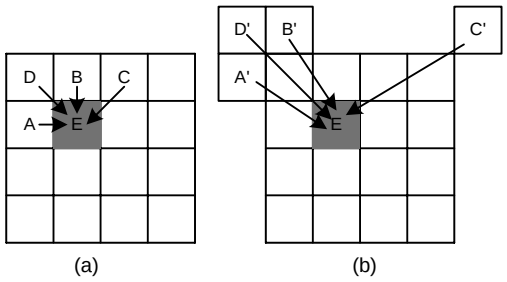

Fig. 5 Accurate motion vector prediction in H.264/AVC (a) and the proposed approximate prediction in inter integer mode decision (IIMD) (b)

Experiments were performed to show the performance of the proposed two-level mode decision mechanism. Four 1080p sequences were used, and we calculated the performance based on the quantization parameter (QP) of 22, 26, 30, and 34, and the group of pictures (GOP) structure of IBBP. For the inter part, a comparison is given in Table 2, and the average Bjontegaard-Delta peak signal-to-noise ratio (BD-PSNR) loss is about 0.2 dB, mainly due to the fast mode decision algorithm. Considering that motion vector prediction (MVP) is used as the search origin in reference software, the loss will be some larger for fast motion sequences, but acceptable (Table 2). Therefore, considering memory access, we used (0, 0) as the search origin (details are given in Section 3.3). Furthermore, Table 3 shows the performance deterioration by applying the proposed two-level mode decision, compared with the reference software JM15.0. Then we can derive the performance decrease caused by the intra part, which is about 0.3 dB since original pixels are used as reference samples in IMS.

Table 2 Performance loss for the inter part

Sequence	BD-PSNR (dB)	
	Compared with JM with search origin MVP	Compared with JM with search origin (0, 0)
Vintage_car	0.15	0.15
Kimono	0.21	0.15
Walking_couple	0.23	0.22
BasketballDrive	0.32	0.12

Table 3 Performance loss for two-level mode decision

Sequence	BD-PSNR (dB)	
	Compared with JM with search origin MVP	Compared with JM with search origin (0, 0)
Vintage_car	0.41	0.41
Kimono	0.58	0.51
Walking_couple	0.53	0.53
BasketballDrive	0.61	0.45

3.2 Sharable architecture of NFME, SFME, and LMC

Traditionally, FME (including NFME and SFME) and MC are arranged in different pipeline stages (Huang *et al.*, 2005; Chen *et al.*, 2006; Lin *et al.*, 2008b); that is, MC is in the third stage and FME in the second. Thus, in these architectures, modules in stages 1–3 should access the off-chip memory sequentially to fetch the reference data, which will add the burden of memory bandwidth. Actually, operations of NFME, SFME, and LMC are quite similar, all containing luma sub-pixel interpolation and sum of absolute difference (SAD) calculation, so their architectures can be shared to a large extent.

Our proposed architecture (Fig. 6) takes full account of the similarity in architectures and the requirement of memory bandwidth among NFME, SFME, and LMC. The architecture is divided into two courses: (1) interpolation; (2) SAD calculation and comparison. The two courses are pipelined, and between them there is a select/control unit to distinguish the three tasks. A time-division multiplexing strategy is performed for NFME, SFME, and LMC to share the architecture.

Specifically, in SFME processes, for skip or direct mode, the output of interpolation is input to both the SAD process element (PE) array and PRED RAM. The advantage is that, once the final mode is skip or direct, it is unnecessary to interpolate the LMC module again, which greatly saves pipeline cycles. Otherwise, the LMC module will refresh the PRED RAM. However, we do not store the NFME interpolation results into PRED RAM, since there are too many candidate sub-pixel positions, which will largely occupy on-chip memory.

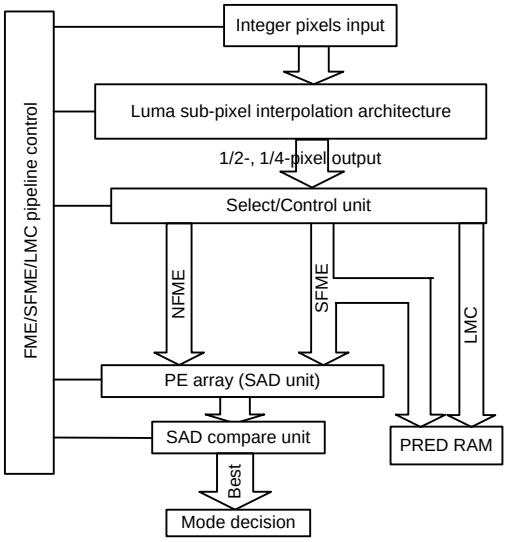

Fig. 6 The proposed architecture of normal mode fractional motion estimation (NFME), special mode fractional motion estimation (SFME), and luma motion compensation (LMC)

The luma sub-pixel interpolation architecture (Fig. 7) adopts a 2D PU (process unit, the interpolation filters for sub-pixels) array and processes the rows and columns synchronously. It processes the candidate MB in the unit of $4\times\delta$ blocks, where δ denotes the height of a candidate block and can be 4, 8, or 16 for different MB partitions. The interpolation process is divided into three steps. First, half pixels b and h are obtained from B PUs and H PUs. Then, using b, h and integer pixels, we can obtain half pixels j, and quarter pixels n/d, c/a, r/p/e/g through J, N/D, C/A and R/P/E/G PUs, respectively. Finally, quarter pixels k/i and q/f are obtained from K/I and Q/F PUs by using h, b, and j. Therefore, in our architecture, all 1/2 and 1/4 sub-pixels are pipeline interpolated and output simultaneously, which greatly increases the throughput.

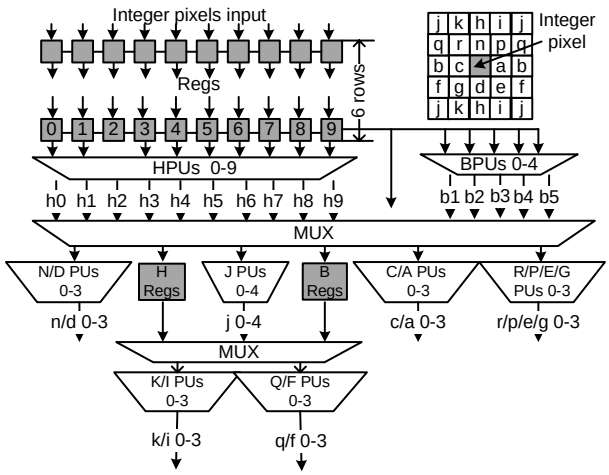

Fig. 7 The proposed luma sub-pixel interpolation architecture and the diagram of all sub-pixels in different positions

3.3 Management of data reuses and bus tasks

Compared to standard-definition (SD, 720×576, 25 frames/s) video coding, the original video data that need to be processed in HD (1920×1080, 30 frames/s) video coding are increased by about six times. Memory bandwidth becomes a major bottleneck because of the larger frame size and search range (Liu *et al.*, 2009). Thus, in the proposed architecture, it is a great challenge to arrange the bus tasks within constrained bandwidth.

In the proposed encoder, the search range (SR) is ±32, and the corresponding search window (SW) size of IME is 80×80. Benefiting from the two-level mode decision mechanism, reference data used in NFME cover the reference data of IME. Furthermore, as our search origin of motion estimation is (0, 0), reference data required for SFME are also included in the reference data used for NFME. Therefore, we extend the SW to 86×86 so that NFME, SFME, and LMC can share the SW data, which greatly reduces the memory access. Two data storage format conversion (DSFC) modules are used here to rearrange the reference data: DSFC1 is used for NFME and LMC, and DSFC2 for SFME (Fig. 8). Additionally, since our search origin is set at (0, 0), most of the SW data of current MB can be reused by the next MB; that is, only 16×86 pixels need to be refreshed, except the first MB in a row.

With the aforementioned approach, we obtain the on-chip memory used in the proposed encoder (Table 4). A comparison of data requirements on the bus (Table 5) clearly shows that the requirement of memory bandwidth is reduced greatly compared with those without data reuse.

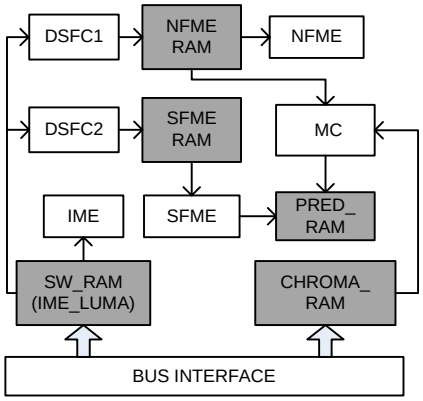

Fig. 8 Data reuse of on-chip memory

Table 4 On-chip memory used in the encoder

RAM	Value (KB)
ORG_RAM	2.560
SW_RAM*	14.792
UP_INFO_RAM	1.425
NFME_RAM	3.200
SFME_RAM	1.568
CHROMA_RAM	1.152
IP_RAM	0.256
PRED_RAM	1.536
SCAN_RAM	0.768
REC_RAM	1.536
DB_RAM	0.512
Other RAMs and FIFOs	1.024
Total	30.329

* Two reference frames

Table 5 Comparison of data requirements on the bus

Module	Data requirement (KB)			
	Predicted search origin		Search origin (0, 0)	
	Without data reuse	With data reuse	Without data reuse	With data reuse (proposed)
IME	12.800	14.792	2.560	2.752
NFME	3.200	0	3.200	0
SFME	1.568	1.568	1.568	0
MC	3.776	0.576	3.776	0.576
Total	21.344	16.936	11.104	3.328

Fig. 9 explains the bus tasks arrangement with the data reuse strategy. There are three types of tasks: (1) ORG_IN and ORG_OUT manage the original data. ORG_IN writes original video input data into off-chip memory, and it can be started flexibly when the bus is idle. ORG_OUT should be finished before IME and IMS because it fetches the original video data from off-chip memory and writes them into ORG_RAM. (2) Search window data of ME and chroma data of MC are prepared by SW_LUMA and MC_CHROMA, respectively. (3) Finally, WB writes reconstructed data of I or P slice back into off-chip memory as the reference data.

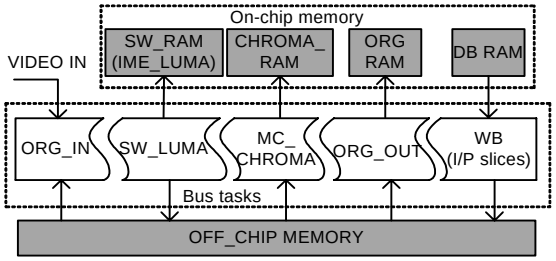

Fig. 9 Bus tasks after data reuse

4 Experimental results

The rate-distortion (RD) curves of the proposed encoder and reference software JM15.0 without RD optimization are shown in Fig. 10. The search range is [−32, +32] for both. Compared with JM15.0, the proposed encoder has an average BD-PSNR decrease of about 0.5 dB for 1080p sequences. The proposed architecture is synthesized in TSMC 0.13 μm technology and uses about 1126k gates under the frequency constraint of 270 MHz. Table 6 shows that it has a very low requirement of memory bandwidth and on-chip memory size. This design is functionally verified based on a field-programmable gate array (FPGA) platform with Xilinx Virtex-V. Simplifications such as decreasing the search range and decreasing the working frequency have been conducted due to the constraint of FPGA. As a result, the proposed encoder design can satisfy real-time video encoding of 1920×1080@30 frames/s for H.264/AVC at the operating frequency of 266 MHz.

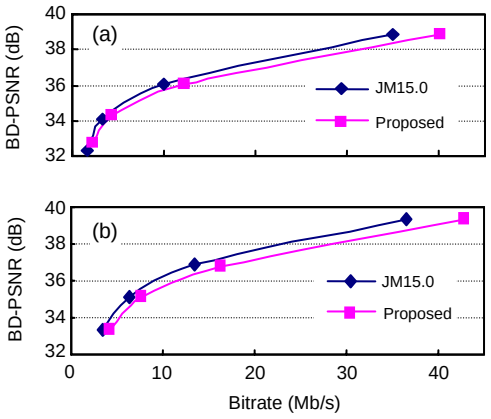

Fig. 10 Rate-distortion curves of the proposed encoder and the reference software JM15.0
(a) Vintage_car; (b) Walking_couple

Table 6 Comparison of different architectures

Parameter	Value/Description			
	Proposed	Liu *et al.* (2009)	Lin *et al.* (2008b)	Chen *et al.* (2006)
Profile	H.264 high	H.264 baseline	H.264 high	H.264 baseline
Maximum resolution	1080p@30 frames/s	1080p@30 frames/s	1080p@30 frames/s	720p@30 frames/s
Process (μm)	0.13	0.18	0.13	0.18
Logic resource (gate)	1126k	1140k	593k	922.8k
Memory bandwidth requirement (KB/MB)*	Total: 4.224; ME: 2.752	–	Total: –; ME: 4.282	–
Operating frequency	266 MHz@1080p	200 MHz@1080p	145 MHz@1080p	108 MHz@720p
Performance loss	0.5 dB with JM15.0	0.2 dB with JM8.1a	0.1 dB with JM9.0**	–
Inter mode	All	8×8 to 16×16	Skip/Direct mode not included	All
On-chip memory size (KB)	30.329	108.3	22	34.72

* Two reference frames; ** 720p IPPP sequences. ME: motion estimation

5 Conclusions

This paper presents an efficient hardware implementation for an H.264/AVC high profile HD encoder. A two-level mode decision mechanism is employed to reduce the complexity and maintain the performance. The sharable architecture is designed for NFME, SFME, and LMC to decrease the hardware costs and to improve the implementation efficiency. Correspondingly, a flexible data reuse strategy is proposed to cut down the requirement of on-chip memory and bandwidth. The synthesized results show that this encoder uses about 1126k gates with an average BD-PSNR decrease of 0.5 dB, compared with JM15.0. It is highly suitable for resource and memory bandwidth constrained high-definition real-time video encoding applications.

References

Chang, T.C., Chen, J.W., Su, C.L., Yang, Y.C., Li, Y., Chang, C.H., Chen, Z.M., Yang, W.S., Lin, C.C., Chen, C.W., *et al.*, 2007. A 7mW-to-183mW Dynamic Quality-Scalable H.264 Video Encoder Chip. IEEE Int. Solid-State Circuits Conf., p.280-281. [doi:10.1109/ISSCC.2007.373403]

Chen, T.C., Chien, S.Y., Huang, Y.W., Tsai, C.H., Chen, C.Y., Chen, T.W., Chen, L.G., 2006. Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder. *IEEE Trans. Circ. Syst. Video Technol.*, **16**(6):673-688. [doi:10.1109/TCSVT.2006.873163]

Chen, Y.H., Chuang, T.D., Chen, Y.J., Li, C.T., Hsu, C.J., Chien, S.Y., Chen, L.G., 2008. An H.264/AVC Scalable Extension and High Profile HDTV 1080p Encoder Chip. IEEE Symp. on VLSI Circuits, p.104-105. [doi:10.1109/VLSIC.2008.4585969]

Huang, Y.W., Chen, T.C., Tsai, C.H., Chen, C.Y., Chen, T.W., Chen, C.S., Shen, C.F., Ma, S.Y., Wang, T.C., Hsieh, B.Y., *et al.*, 2005. A 1.3TOPS H.264/AVC Single-Chip Encoder for HDTV Applications. IEEE Int. Solid-State Circuits Conf., p.128-133. [doi:10.1109/ISSCC.2005.1493902]

ITU-T, 2005. ITU-T Recommendation and International Standard of Joint Video Specification. H.264/ISO/IEC 14496-10 AVC.

Lee, K.H., Alshina, E., Park, J.H., Han, W.J., Min, J.H., 2008. Technical Considerations for Ad Hoc Group on New Challenges in Video Coding Standardization. MPEG Doc. M15580. Hannover, Germany.

Lin, Y.K., Lin, C.C., Kuo, T.Y., Chang, T.S., 2008a. A hardware-efficient H.264/AVC motion-estimation design for high-definition video. *IEEE Trans. Circ. Syst. I*, **55**(6):1526-1535. [doi:10.1109/TCSI.2008.916681]

Lin, Y.K., Li, D.W., Lin, C.C., Kuo, T.Y., Wu, S.J., Tai, W.C., Chang, W.C., Chang, T.S., 2008b. A 242mW, 10mm^2 1080p H.264/AVC High Profile Encoder Chip. 45th ACM/IEEE Design Automation Conf., p.78-83.

Liu, Z.Y., Song, Y., Shao, M., Li, S., Li, L.Y., Ishiwata, S., Nakagawa, M., Goto, S., Ikenaga, T., 2009. HDTV1080p H.264/AVC encoder chip design and performance analysis. *IEEE J. Sol.-State Circ.*, **44**(2):594-608. [doi:10.1109/JSSC.2008.2010797]

Park, S.H., Lee, J.H., 2008. Hardware Architecture for Real-Time HD(1920×1080@60fps) H.264/AVC Intra Prediction. IEEE Int. Symp. on Consumer Electronics, p.1. [doi:10.1109/ISCE.2008.4559505]

Wiegand, T., Schwarz, H., Joch, A., Kossentini, F., Sullivan, G.J., 2003. Rate-constrained coder control and comparison of video coding standards. *IEEE Trans. Circ. Syst. Video Technol.*, **13**(7):688-703. [doi:10.1109/TCSVT.2003.815168]

Yang, C.Q., Goto, S., Ikenaga, T., 2006. High Performance VLSI Architecture of Fractional Motion Estimation in H.264 for HDTV. Proc. IEEE Int. Symp. on Circuits and Systems, p.1-4. [doi:10.1109/ISCAS.2006.1693157]