A Cache-based Bandwidth Optimized Motion Compensation Architecture for Video Decoder

Meng Li[1,2], Huizhu Jia[1*], Xiaodong Xie[1], Jason Cong[2], Wen Gao[1]
[1]National Engineering Laboratory for Video Technology, Peking University, Beijing, China
[2]UCLA/PKU Joint Research Institute in Science and Engineering
{mmli, hzjia, donxie, wgao}@pku.edu.cn, cong@cs.ucla.edu

***Abstract*—In video decoder applications, motion compensation (MC) is bandwidth consuming because of the non-regular memory access. Especially with the popularity of UHD video and the development of new coding standard (HEVC), external memory bandwidth becomes a crucial bottleneck. In this paper, we propose an area efficiency cache-based bandwidth optimization strategy to minimize the memory bandwidth. First a four-way parallel cache architecture is described. Then partially replacement strategy is proposed to further reduce memory bandwidth and power consumption. At last a column based storage scheme is provided to reduce the precharge/active frequency. We realize this idea using high level synthesis, which allow multiple iterations with quick turnaround time for micro architecture changes, and the results show that the averagely bandwidth reduction is up to 79.9% with moderate resource utilization, which outperforms the state-of-the-art works.**

***Index Terms*—bandwidth, MC, cache, decoder, HLS**

I. INTRODUCTION

In video sequences, there is a strong correlation between adjacent frames because of the successive movement of the objects, and inter-frame prediction is used to eliminate the temporal redundancy. One or more decoded frames are used as reference frames, which typically stores in an external memory such as double data rate synchronous dynamic random access memory (DDR-SDRAM). Accessing to these reference pixels for motion compensation (MC) results in a great memory bandwidth and power consumption. Based on the simulation, the MC will take up to 75~83% memory bandwidth in H.264 without applying any MC bandwidth reduction technique [1].

Several techniques have been proposed to reduce MC bandwidth requirement in video decoder system. Tsai *et al.* [2] proposed an interpolation window reuse (IWR) scheme which load reference data according to macro-block (MB) type. Wang *et al.* [3] proposed four motion compensation memory access optimization strategies for H.264/AVC decoder. Li *et al.* [4] proposed a classification scheme for MB-level processing pipeline. These data reuse schemes can reuse overlapped reference data by storing certain data blocks in a register file and re-used by next blocks. However, the performance is limited when the motion vectors of neighboring blocks are different. Chuang *et al*. [1] further proposed a cache-based MC architecture to reduce both the reference data loading bandwidth and the equivalent bandwidth from DRAM access overhead latency. In recent years, as the successor to H.264, High Efficiency Video Coding (HEVC) achieves better performance but adds extra burden on complexity for hardware implementation because of the new coding tools. Coupled with the specification of Ultra High Definition Television (UHDTV), memory bandwidth problem becomes more severe. In [5], Wang *et al*. proposed a four-bank parallel 2D cache organization and a pipelined Write-Through mechanism (WTM) to achieve conflict-free performance. Tikekar *et al*. [6] proposed a high-throughput read-only cache combined with DRAM-latency-aware memory mapping to reduce DRAM bandwidth. Although these methods achieve better performance than the previous, the area efficiency and complexity still have large room to improve for reducing power consumption.

In this paper, we propose a cache-based bandwidth optimized motion compensation architecture for the latest video standard. We first describe the potential of cache-based MC architecture and how the algorithm guiding the cache design, including the cache size and refresh strategy. Then a four-way parallel 2D cache architecture is proposed to reuse the locality between references. The VLSI architecture is implemented using high level synthesis [7] tool which raises the level of abstraction beyond register transfer level (RTL) and gives us better control over the optimizations of the system architecture. By which we explore a large design space and the experiments show that the proposed architecture can achieve 79.9% bandwidth reduction with moderate resource utilization.

II. BACKGROUND AND MOTIVATION

Inter frame prediction is known as searching a block that is similar to the current one in a reference frame, and the motion vector (MV) points to the position of the matching block at the reference frame. In decoding process, first we need to access the reference data for MC according to the MV position, and then the reference data need to be interpolated based on the MV type (integer-pixel or sub-pixel). Our architecture design is based on the following two important observations derived from an analysis of the motion between adjacent frames.

Observation 1: Enormous data overlap but irregular block size and directing position

*the corresponding author, Huizhu Jia is with Peking University, also with Cooperative Medianet Innovation Center and Beida(Binhai) Information Research.

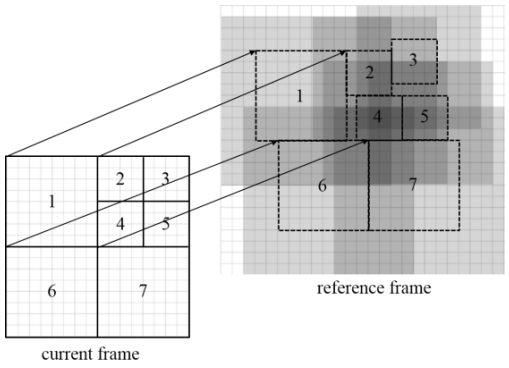

Fig 1. Data overlap of 7 sub-blocks within a 16x16 block

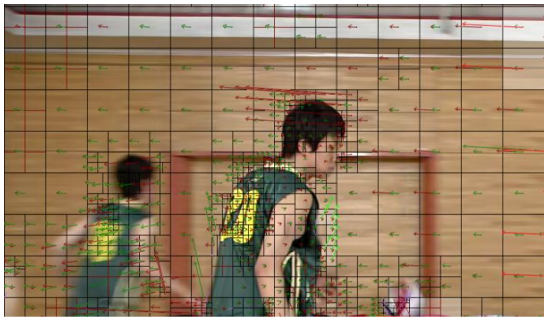

Fig 2. Motion correlation of sequence *'BasketballDrill'*

In the latest coding standard (such as HEVC, AVS2), it adopts a flexible hierarchical structure for Coding Unit (CU) and Prediction Unit (PU), where CU ranging from 8 to 64 while each CU can be further partitioned into symmetric or asymmetric PUs. In addition, it supports a longer 8-tap filter to realize more accurate interpolation. Taking 8x8 PU for example, 15x15 reference pixels are needed, around 72% data overhead are added on external memory bandwidth.

As shown in Fig. 1, a 16x16 block in inter coded frame is divided into 7 blocks, and the corresponding mating blocks in the reference frame are shown by dotted line. The shaded area in reference frame illustrates the required pixels by each block. We can see that there are enormous data overlap exist, and the darker area means more frequently it is to be used in the reference frame. Although there is a great potential to reuse these overlap data, it is hard to realize efficiently simply by a data reuse buffer because of the uncertainty of block size and the directing position in reference frame. In the following sub-section, we explore the statistical property in video sequence, which guides the design of the proposed cache-based motion compensation architecture.

Observation 2: Motion correlation of adjacent blocks

Fig. 2 shows an example about the motion correlation for sequence *'BasketballDrill'*, the red and green arrows represent two motion vectors for each PU. We can see that neighbouring PUs have the tendency of similar motion directions. If so, when the current block accesses a certain chunk of memory, its neighbouring blocks are likely to access that location near it. We know that when the cache size is too small, frequently data refreshing will occur as the low hit rate. On the contrary, if the cache size is too large, there will be long loading latency and it is a huge resource consumption. Following the experimental results in [5], the hit rate of cache increases with the cache size, but when the cache size reaches 64x64 pixels, the increasing tends to flatten out. We suppose the reason is that the correlation within a LCU is strong and the overlap region takes a great proportion.

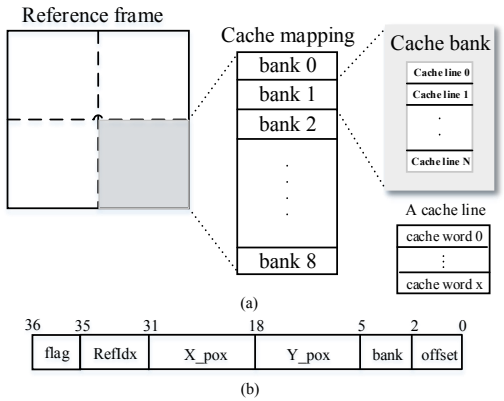

Fig 3. (a) Data mapping in proposed cache organization (b) Tag contents of cache block

Unlike in the software applications, where there are many looping and branching constructs, in real decoding applications, the decoding sequence is raster scan order, which is top to bottom and left to right. And from the statistical results, in most cases, the motion vectors of adjacent two LCU will not differ greatly (we marked as a distance of 64). So in general, the reference area is also correspondingly sliding from top to bottom and left to right as raster scan order. It is most likely that the first block in cache is the left and top reference data, which is the farthest from the current block and the least likely to be referred. So FIFO replacement strategy is considered and its validity has also been confirmed by experiment.

III. Architecture Design

To alleviate the MC bandwidth requirement, we propose a cache-based motion compensation architecture with friendly DRAM access control. Our architecture design is targeted on FPGA and described in C code, subsequently converted into RTL automatically by High Level Synthesis (HLS) tools, which raises the level of abstraction beyond register transfer level and gives us efficiency in exploring the optimizations of the hardware architecture.

A. Data Mapping Design

In our proposed cache architecture, we adopt a 2-D data mapping strategy as shown in Fig. 3(a). To improve the utilization of cache block, any block of data in reference frame could map to a cache block, which has been partitioned into 8 banks. A cache bank consists of *N* cache line and a cache line consists of *X* cache word. The size of a cache word is equivalent to the external memory bus width, which is 64-bits in our design (using Xilinx Kintex-7 FPGA KC705). Based on the observation in section 2, we choose 64x64 as the size of one way cache block. Each cache line represents 64 pixels in a line of image. Thus, a

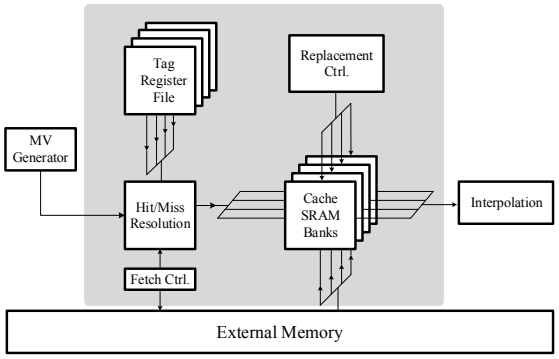

Fig 4. System architecture of proposed 4-way parallel cache

cache line composes of 8 cache words and each cache line can be mapped to a unique cache bank. From the above, we can see that each cache bank has 8 cache lines. To increase the parallelism of BRAM access in FPGA, we use a cyclic memory partitioning scheme which mapping adjacent rows into different banks so that multiple data accesses most likely access data in different banks. This reduces the probability of changing rows in a bank to save the latency on precharge/activate cycles.

To reduce the complexity, we use a 37-bit tag register to store the information of one-way cache. As shown in Fig. 3(b), the tag register includes reference frame index (*RefIdx*), x_postion (*X_pos*), y_postion (*Y_pos*), bank number (*bank*) and offset in bank (*offset*). The *RefIdx* uses four bit signal to represent at most 15 reference pixels. *X_pos* and *Y_pos* indicate the location of pixels in a frame, which support up to 8K UHDTV. To reduce bandwidth and power consumption, we also use *bank* and *offset* signal to achieve partially refreshing, which will be discussed in next sub-section. In addition, *flag* is a one-bit signal to indicate the availability of data to be replace. As long as the data in cache has not been used for interpolation, the *flag* signal is set to 1 and indicates it can be replaced for storing new data.

B. 4-way Parallel Cache Organization

This section describes the system architecture of our proposed four-way parallel MC cache with a partially replacement strategy. In addition, we also propose a data prefetch scheme to set up pipeline.

1) Four Parallel Data Flow

As shown in Fig. 4, the address of required pixels is generated by *MV Generator*, and then *Hit/Miss Resolution* compares this address with the information in four parallel *Tag Register File*. Once hit, the index indicating target pixels will be sent to *cache SRAM*, and the *interpolation* module could fetch these data from cache directly. Note that the flag of the hit cache should be set to 0 indicating not available for rewriting. In case of miss, the *Replacement Control* will decide which block to refresh, at the same time *Fetch Control* accesses to external memory and writes to cache. Subsequently, the cache tag for the missed cache updates immediately after requesting from external memory. There are four parallel paths of *cache SRAM* and each connecting to the *interpolation* module as the input data.

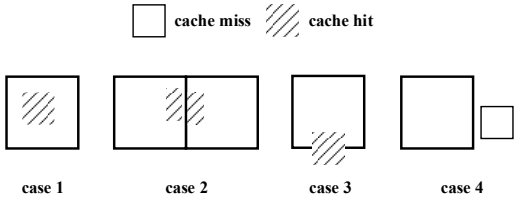

Fig 5. Cases for hit/miss resolution

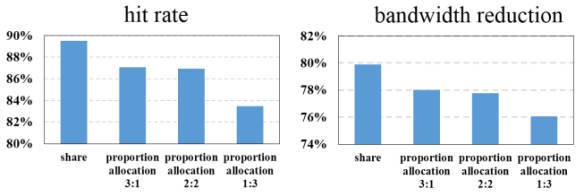

Fig 6. Cache performance for different allocation scheme

2) Partially Replacement Strategy

As described in section 2, we adopt FIFO replacement policy in the proposed cache system, which means the oldest cache will be replaced by new arrived data. To further reduce memory bandwidth and power consumption, we propose a partially replacement strategy. As shown in fig. 5, we illustrate four cases of cache hit/miss resolution. In case 1 and case 2, required pixels store in one or several cache blocks, which can be directly used by *interpolation*. In case 4, all cache blocks miss for the target pixels, and FIFO replacement is adopted to rewrite one cache block. But it can be seen in case 3, only a part of data missing in cache. Instead of refreshing the entire cache block, we can only refresh several cache lines, and fix the starting position by '*bank*' and '*offset*' signal in tag register using circular mapping strategy.

3) Data Organization in External Memory

To reduce the precharge/active frequency when accessing data in external memory, the reference pixels are stored in DDR by column instead of row. As the processing unit moves as raster scan order from left to right and top to bottom, it is most likely a few columns need to be refresh in cache block. The column based storage scheme changes the access pattern to load reference data and alleviate the DRAM access overhead, thus to reduce the memory bandwidth further.

C. Investigation on Bi-directional Prediction Mode

Bi-prediction performs motion compensation from two reference frame lists rather than one for a better coding performance, and two blocks of data are required. In general, these two reference blocks come from two different reference frames, thus may lead to cache miss. HLS provides us significantly improved design productivity compared to traditional RTL based design flow, so we explore large design space of cache allocation for bi-directional prediction to investigate the influence. As shown in Fig. 6, *'share'* means two reference frame lists share a common 4-way cache architecture,

TABLE I
HIT RATE AND BANDWIDTH REDUCTION RESULTS

Seq. Class	QP	hit rate		bandwidth reduction	
		configuration		configuration	
		RA	LD	RA	LD
Class A	30	91.28%	92.11%	80.49%	84.43%
	35	89.09%	90.97%	78.32%	82.67%
	40	86.23%	89.15%	76.26%	80.75%
	45	83.06%	86.81%	74.59%	78.96%
Class B	30	93.32%	94.08%	80.82%	86.86%
	35	91.57%	93.12%	78.15%	84.24%
	40	89.23%	91.95%	75.43%	82.23%
	45	87.34%	90.47%	73.86%	80.22%
Class C	30	92.27%	92.17%	82.70%	84.99%
	35	90.76%	91.23%	81.04%	83.31%
	40	88.38%	89.53%	78.85%	81.04%
	45	86.04%	87.68%	77.12%	79.26%
Class D	30	93.48%	91.85%	85.22%	88.01%
	35	92.27%	91.51%	82.76%	85.84%
	40	91.13%	90.79%	80.47%	82.56%
	45	91.26%	90.13%	78.75%	80.92%
Class E	30	87.57%	88.92%	76.18%	79.08%
	35	85.86%	86.90%	74.79%	77.17%
	40	86.45%	84.65%	73.86%	75.79%
	45	85.54%	83.33%	72.99%	75.00%
Class F	30	92.19%	90.90%	81.59%	83.95%
	35	91.06%	90.43%	79.37%	81.89%
	40	89.54%	89.43%	77.33%	79.85%
	45	86.63%	88.32%	75.94%	78.45%
average		**89.54%**		**79.88%**	

'*proportion allocation 3:1'* stands for allocating 3 way cache for list0, and 1 way cache for list1. It can be seen that although with switching reference frame, 4-way cache architecture is the best solution, which is chosen for the final design.

IV. EXPERIMENTAL RESULT

To evaluate the efficiency of the proposed cache architecture, we test both the *hit rate* and the *bandwidth reduction* for several sequences. The baseline of bandwidth reduction is the ideal bandwidth with no cache, which is the exact number of reference pixels needed for interpolation. We also show the synthesis result and compare our work with the state-of-the-art works.

A. Cache Performance Evaluation

We conduct the experiment in accordance with the common test conditions and test different sequences in class A (2560x1600), class B (1920x1080), class C (832x480), class D (416x240), class E (1280x720) and class F (1024x768). Both the low-delay (LD) and random-access (RA) configurations are tested with Quantization Parameter (QP) varies from 30 to 45. Table I summarizes the *hit rate* of cache and the *bandwidth reduction* of each sequence. The simulation result shows that the average hit rate of proposed cache architecture is up to 89.54%, and the average bandwidth reduction of proposed scheme is 79.88%, which validate the high efficiency of proposed cache architecture.

B. Synthesis Result and Comparison

TABLE II
SYNTHESIS RESULT COMPARISON WITH PREVIOUS WORKS

	[4]	[1]	[5]	[6]	Proposed
Standard	H.264	H.264	HEVC	HEVC	HEVC
ASIC Library /FPGA device	UMC 0.18	UMC 90nm	TSMC 90nm	40nm	Xilinx Kintex-7
Max Frequency	100 MHz	166 MHz	250 MHz	200 MHz	200 MHz
Core Area	13K	72K	104K	126K	96K
Bandwidth reduction	67.7%	71%	62.2%	67%	80%

The proposed MC cache architecture is first described using C code, and then synthesized into RTL using Xilinx Vivado HLS version 2013.2, subsequently implemented into FPGA configuration by Xilinx ISE 14.4. Our verification target is Xilinx Kintex-7 FPGA KC705 platform. The equivalent total gate count is about 96K. Table II lists the comparison results of proposed scheme with other works. Compared with the four previous designs, the proposed design achieves the best performance on bandwidth saving with moderate resource utilization.

V. CONCLUSION

In this paper, we propose an efficient cache-based MC architecture for HEVC decoder system to minimize the memory bandwidth. First, we describe a four-way parallel cache architecture including data mapping strategy and data flow, then partially replacement strategy is proposed to further reduce memory bandwidth and power consumption. In addition, column based storage scheme is proposed to precharge/active frequency. Our architecture design is targeted on FPGA and described in C code, subsequently converted into RTL automatically by high level synthesis tools. Experimental results show that the average bandwidth reduction is up to 79.9% with moderate resource utilization, which outperforms the state-of-the-art works.

ACKNOWLEDGEMENT

This work is partially supported by grants from National High Technology Research and Development Program of China (863 Program) under contract No.2015AA015903, the National Science Foundation of China under contract No. 61421062 and No.61520106004.

REFERENCES

[1] TD. Chuang et al. "Bandwidth-efficient cache-based motion compensation architecture with DRAM-friendly data access control." *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on.* IEEE, 2009.

[2] CY. Tsai et al. "Bandwidth optimized motion compensation hardware design for H. 264/AVC HDTV decoder." *Circuits and Systems, 2005. 48th Midwest Symposium on.* IEEE, 2005.

[3] R. Wang, J. Li and C. Huang. "Motion compensation memory access optimization strategies for H. 264/AVC decoder." *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on.* Vol. 5. IEEE, 2005.

[4] Y. Li, Y. Qu and Y. He. "Memory cache based motion compensation architecture for HDTV H. 264/AVC decoder." *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on.* IEEE, 2007.

[5] S. Wang, D. Zhou and S. Goto. "Motion compensation architecture for 8K UHDTV HEVC decoder." *Multimedia and Expo (ICME), 2014 IEEE International Conference on.* IEEE, 2014.

[6] M. Tikekar et al. "A 249-Mpixel/s HEVC video-decoder chip for 4K ultra-HD applications." *Solid-State Circuits, IEEE Journal of* 49.1 (2014): 61-72.

[7] J. Cong, et al. "High-level synthesis for FPGAs: From prototyping to deployment." *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 30.4 (2011): 473-491