Optimized mapping of pixels into memory for H.264/AVC decoding

Youhui Zhang[a), Yuejian Xie, and Weimin Zheng

Department of Computer Science and Technology, Tsinghua University, Beijng, 100084, China.

a) *zyh02@tsinghua.edu.cn*

Abstract: This paper presents an optimized mapping of pixels into the external SDRAM for H.264/AVC video decoding. The modern SDRAM system usually employs the burst-access mode as well as the multi-bank architecture to improve the utility efficiency. If the accessed video pixels can be placed uniformly and successively across different memory banks, both of them will be used with high efficiency. Based on a statistic method, this paper traces and analyzes the actual memory accesses of H.264/AVC decoding and gives an optimized placement of pixels in the SDRAM——the optimized data unit is 8x8. Moreover, the efficiencies of the optimized cache and pre-fetch technologies are also evaluated, which show that this optimization can improve both the cache hit ratio and the pre-fetch effect apparently.

Keywords: H.264/AVC decoding, memory mapping, DRAM, burst access

Classification: Integrated circuits

References

[1] H. Kim and I.-C. Park, "Array address translation for SDRAM-based video processing applications," *Electron. Lett.*, vol. 35, no. 22, pp. 1929–1931, Oct. 1999.

[2] E. G. T. Jaspers and P. H. N. de With, "Bandwidth reduction for video processing in consumer systems," *IEEE Trans. Consum. Electron.*, vol. 47, no. 4, pp. 885–894, Nov. 2001.

[3] P. Grun, N. Dutt, and A. Nicolau, "APEX: access pattern based memory architecture exploration," *Proc. 14th Int. Symp. System Synthesis*, pp. 25–32, 2001.

[4] J. Lee, C. Park, and S. Ha, "Memory access pattern analysis and stream cache design for multimedia applications," *Proc. 2003 conference on Asia South Pacific design automation*, pp. 22–27, 2003.

[5] K. Suhring, "H.264/AVC Software Coordination," Available, [Online] http://iphome.hhi.de/suehring/tml/.

[6] R. G. Wang, J. T. Li, and C. Huang, "Motion compensation memory access optimization strategies for H.264/AVC decoder," *Proc.* IEEE *International Conference on Acoustics, Speech, and Signal*, pp. 97–100, March 2005.

DOI: 10.1587/elex.6.283
Received December 10, 2008
Accepted February 12, 2009
Published March 25, 2009

[7] J. Edler and M. D. Hill, “Dinero IV Trace-Driven Uniprocessor Cache Simulator,” Available, [Online] http://www.cs.wisc.edu/˜markhill/DineroIV.

1 Introduction

The H.264/AVC video coding standard has gained more and more attention, mainly due to its higher coding efficiency compared with previous standards such as MPEG 1/2/4. However, along with the great improvement of compression rate, the access pressure of the external memory also increases correspondingly. For example, the H.264/AVC motion compensation algorithm will access the small blocks of reference frames randomly over a large range, therefore, its spatial locality is poor and the cache cannot achieve a high hit ratio. It means how to access the pixels in the external SDRAM efficiently is very important for decoding.

This paper focuses on the optimized mapping of video data into the memory to use the burst access mode and the multiple-bank architecture of SDRAM with high efficiency. We trace and analyze the actual memory accesses of H.264/AVC decoding and present an optimized placement of pixels in the SDRAM, which is suitable for software implementation. Moreover, the efficiencies of the optimized cache and pre-fetch technologies are also evaluated to show the improvement.

2 Previous work

Analysis of memory access of video decoding applications to decrease the transfer overhead between the memory and the processing unit have been proposed by some previous works. For example, a pixel address translation strategy is proposed in [1] to decrease the number of overhead cycles needed for row-activations and precharges. The features of SDRAM as well as the memory accesses in MPEG2 video processing applications are explored to find such a suitable translation. [2] can be regarded as an extension of [1], which analyzes the actual memory accesses of MPEG2 decoding, so that data dependencies are also considered.

Some other works propose adaptive cache strategies that employ suitable cache organizations for different decoding steps. For example, [3] presents APEX, an approach that analyzes the most active access patterns in the decoding application, and customizes the memory architecture to match the needs of the application. [4] gives a proposal to perform static analysis of data access pattern and then adopts suitable on-chip stream caches to improve the IO efficiency.

Inspired by [1, 2], we focus on the optimized mapping for H.264/AVC video. And as we know, there is no similar work on this aspect for H.264/AVC.

© IEICE 2009
DOI: 10.1587/elex.6.283
Received December 10, 2008
Accepted February 12, 2009
Published March 25, 2009

3 Descriptions of the problem

3.1 DDR SDRAM

DDR SDRAM and its succeeding use two main features: the burst-access mode and the multiple-bank architecture.

A modern SDRAM chip consists of multiple DRAM banks to allow multiple outstanding memory accesses to be handled in parallel if they require data from different banks. The data in a bank can be accessed only from the row-buffer, which can contain at most one row content. A bank contains a single row-buffer.

To access data in the memory, a row-activate command is issued to a bank to copy the row data into the row-buffer at first. Then, a read or write command for the same bank is issued to access the required unit in the row. When all required data in the row have been accessed, the corresponding bank can be precharged for the subsequent access in the same bank, because the read of DRAM is destructive. It means there is minimum time from the start of one row access to the start of the next access in the same bank, which is denoted as tRC and is typically 10 cycles for the current DDR SDRAM and $12 \sim 23$ cycles for DDR2 SDRAM. In another word, the same bank cannot be accessed again during the tRC period.

The multiple-bank architecture can be used to overcome this problem ——a bank can be accessed when others are precharged. Namely, sequential burst accesses can be issued to multiple banks one by one without any waiting time.

To optimize the utilization of the memory bandwidth, data had better be accessed at the grain size of a data burst, which is a non-overlapping block in the memory that can only be accessed as an entity. In the remainder of this paper, these blocks are referred to as data units.

3.2 The data mapping

How to map the pixels into the multiple banks is essential for the utilization of the memory bandwidth.

In some mapping mode(s), pixels are stored in the sequential banks line by line. Therefore, a required block of pixels may cover several data units and results in the transfer of all host units. To decrease the extra overhead, it appears that a smaller data unit is preferred. However, if the burst length is so small that the elapse time of accessing all banks is smaller than tRC, it will cause some waiting cycles. As mentioned in [2], to avoid this waste, a minimum burst length of DDR SDRAM should be calculated by Equation 1.

$$BurstLen \geq t_{RC} \times 2 \div Number\_of\_Bank \tag{1}$$

Because for DDR SDRAM, the frequency of data access doubles that of the memory cells, $t_{RC} \times 2$ is the actual transfer amount.

Like [2], we also assume there are 4 banks in the external SDRAM system and the bus width is 64 bits, so that the burst length should be set as 8, which is larger than the minimum $(10 \times 2 \div 4 = 5)$, and the data unit size is 64 bytes.

© IEICE 2009
DOI: 10.1587/elex.6.283
Received December 10, 2008
Accepted February 12, 2009
Published March 25, 2009

The same computation principle could be applied to DDR2 SDRAM, because for the latter, the main improvement lies in that the bus is clocked at twice the speed of the memory cells, so four bits of data can be transferred per memory cell cycle.

Of course, there are many mapping modes of pixels into memory. A naïve mode is to store 64 continuous pixels of a frame line into one data unit.

In this mode, when one 8x8 reference block is to be accessed, at least 64x8 pixels will be transferred. If the reference block crosses two units horizontally, the amount is even as much as 128x8 pixels. If the mapping mode is set as 16x4, the transfer efficiency will be higher (the transfer amount is not more than 32x12). However, forms of reference blocks are variant, which is the decisive factor for the transfer efficiency while other conditions are fixed.

Now, the key point is how to choose an optimized mapping mode for H.264/AVC decoding.

4 The method

Here four real H.264/AVC video clips compressed by JM 12.0 [5] are selected as the test input and JM is still used as the decoding software. During the compression process, many H.264/AVC features are enabled, including Intra / Inter Motion Compensation, I / B frames, and variant block search ranges (like 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4).

In our test, accesses of data for intra / inter prediction of macroblocks are considered, because they occupy about 75% of the total memory accesses [6].

As we know, the mapping forms of data units have the following types because the data unit size is 64.

$$\mathrm{A} = \{(64\mathrm{x}1), (32\mathrm{x}2), (16\mathrm{x}4), (8\mathrm{x}8), (4\mathrm{x}16), (2\mathrm{x}32), (1\mathrm{x}64)\}$$

For these forms, all testing videos are decoded and the memory access traces of inter / intra prediction are recorded. Through the analysis of the decoding source code, we can draw the following information of each memory access:

The upper-left corner position (x_0, y_0) of the current accessed macroblocks and the width and height.

Then, the ideal transferred amount (no any extra pixel is transferred) of reference blocks can be calculated.

In addition, under different mapping modes, the actual transferred data amount can be also calculated, which is then compared with the ideal amount to show the efficiencies of different mapping modes.

The concrete calculation formulas are presented as follows:

The dimension of the current accessed block is known as $W \times H$, and the form of data unit is assumed as $M \times N$. The real transferred number of data units is——

Case 1: if $(x_0 \bmod M) + W - M \leq 0$, the number of horizontal units covered by the required block is 1.

Case 2: Let $p = (W - M + x_0)$, then if $p \bmod M = 0$, the number of horizontal units covered by the required block is $\lfloor p \div M \rfloor + 1$; otherwise the number is $\lfloor p \div M \rfloor + 2$.

© IEICE 2009
DOI: 10.1587/elex.6.283
Received December 10, 2008
Accepted February 12, 2009
Published March 25, 2009

Similarly, the number of vertical units covered is 1, if $(y_0 \bmod N) + H - N \le 0$.

If $(y_0 \bmod N) + H - N > 0$, let $q = (H - N + y_0)$, and if $q \bmod N = 0$, the number of vertical units covered by the required block is $\lfloor q \div N \rfloor + 1$; otherwise the number is $\lfloor q \div N \rfloor + 2$.

The relationship of the above parameters is illustrated in Figure 1.

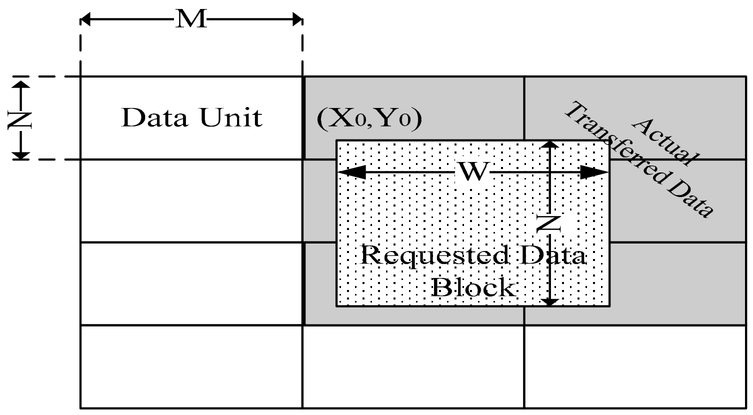

Fig. 1. The relationship of data unit, requested data and transferred data

At last, ratios of the real transferred amounts to the ideal amounts are presented in Table I.

Table I. Transfer efficiencies under different mapping

	highway	mobile	paris	tempete
64x1	450%	592%	537%	568%
32x2	264%	371%	304%	356%
16x4	178%	272%	195%	262%
8x8	**168%**	**247%**	**179%**	**242%**
4x16	190%	268%	194%	257%
2x32	284%	362%	302%	351%
1x64	487%	580%	532%	572%

It is apparent that 8x8 is the optimized mapping mode for H.264/AVC, because for all test videos, this mode introduces the least extra overheads.

It is necessary to note that this conclusion is based on the given SDRAM specification——the burst length is 64 bytes and the number of memory banks is 4. It is apparent that the optimized mode may be different if the specification is altered. General speaking, the optimal mapping depends on the following issues:

the SDRAM specification,
the forms of the requested data blocks,
the probability of their occurrence,
and the probability distribution of their positions;

© IEICE 2009
DOI: 10.1587/elex.6.283
Received December 10, 2008
Accepted February 12, 2009
Published March 25, 2009

The last two parameters mean that the statistics of memory accesses should be considered. Our method gives a statistical framework, which selects some representative samples as the input data to get the real access traces under any given SDRAM specification, and then evaluates diverse mapping modes.

5 Optimized cache and evaluation

Based on the optimized mapping, data are stored in the memory block by block rather than line by line. Then, it is apparent to design a new cache addressing method to adapt to the new storing mode.

In the traditional mode, the frame pixels are usually stored in a 2-dimension array $[x_0, y_0]$, where x_0 stands for the row position and y_0 for the column. In the new mapping mode, we adopt a different addressing method, where the first index (x') represents which data unit the request data is located in and the second (y') is its offset in this unit.

The conversion between them is described as follows:

$$\begin{aligned} dy &= y_0/N; \quad ry = y_0 \bmod N; \\ dx &= x_0/M; \quad rx = x_0 \bmod M. \end{aligned}$$

(dx, dy) stands for the data unit position of the pixel (x_0, y_0) while (rx, ry) is the offset. Then, the new address is:

$$\begin{aligned} X' &= dy + dx; \\ Y' &= ry \times M + rx. \end{aligned}$$

Especially, because the optimized data unit is 8x8, the calculations can be reduced as follows:

$$\begin{aligned} &dy = y_0 \gg 3; \quad ry = y_0 \ \&\ 0x7; \\ &dx = x_0 \gg 3; \quad rx = x_0 \ \&\ 0x7; \\ &Y' = ry \ll 3 + rx. \end{aligned}$$

These operations, like $\gg$ and &, can be performed very quickly by software. For other optimized data units, like 32x2, 16x4, et al., the reduction is still feasible because they are powers of 2.

Based on the memory access traces, we use Dinero [7] as the cache simulator to evaluate the cache efficiencies under the two addressing methods.

The cache configuration is set as follows:

Cache size: 64 KB
Cache line size: 64 Byte
Association: 4
Replacement strategy: FIFO
Write policy: write back

Then, the cache miss ratios of the two methods are presented in Table II: We can see that the miss ratios are decreased by 25% averagely.

DOI: 10.1587/elex.6.283
Received December 10, 2008
Accepted February 12, 2009
Published March 25, 2009

Table II. Cache miss ratios

Video Clips	Miss ratio of the traditional mode without pre-fetch	Miss ratio of the optimized mode without pre-fetch	Miss ratio of the optimized mode with pre-fetch
highway	0.0228	0.0184	0.0028
mobile	0.0328	0.0221	0.0046
paris	0.0195	0.0164	0.0021
tempete	0.0348	0.0235	0.0053

Under the new mode, the cache can also regard the data unit as the pre-fetch unit, which improves the efficiency further. The results are also presented in the right column of Table II.

Of course, pre-fetch will introduce more transfer overheads definitely. Still using Dinero, we compare the transfer amount of the traditional mode to the new one with pre-fetch. Results show that the optimized mode with pre-fetch is still better than the traditional one from the viewpoint of the data transfer amount: the amount is decreased by 11.2% averagely.

6 Conclusion

This paper presents an optimized strategy to store H.264/AVC pixels into SDRAM to decrease the transfer amount between the decoding unit and the memory. The access traces of decoding real videos are analyzed; and we calculate the actual data transfer amounts considering the effect of the burst access and the multiple-bank architecture. Results show that to place pixels into SDRAM with 8x8 data units is the optimized mode under a given SDRAM specification and the same method can be applied to different specifications to find the corresponding optimized mode. Moreover, a new addressing method is given to improve the cache efficiency under the optimized mapping configuration, as well as the corresponding pre-fetch technology. Evaluations also show that they both can gain better results than the original one.

Acknowledgments

The work is supported by the High Technology Research and Development Program of China under Grant No. 2006AA01Z111 and the National Science Foundation of China under Grant No. 60773147.

© IEICE 2009
DOI: 10.1587/elex.6.283
Received December 10, 2008
Accepted February 12, 2009
Published March 25, 2009