TEXTURE CODER DESIGN OF MPEG-4 VIDEO BY USING INTERLEAVING SCHEDULE

Chih-Wei Hsu, Wei-Min Chao, Yung-Chi Chang, and Liang-Gee Chen

DSP/IC Design Lab, Graduate Institute of Electronics Engineering,
Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan
{jeromn, hydra, watchman, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

For MPEG-4 texture coding, an efficient Interleaving DCT and IDCT Schedule (IDIS) is proposed. With this scheme, DCT-Q-IQ-IDCT coding loop can be implemented with no buffers and least latency, which in turn makes the number of buffer for MC a minimum of two. Also by the characteristics of IDIS, sub-structure sharing technique is applied for DC/AC prediction with Q and IQ to reduce hardware cost further. All the functions are integrated to comprise the block engine for texture coding operations in the MPEG-4 video standard. For encoding sequence of 720x480 at 30 fps, real-time requirement can be achieved at 54 MHz. The proposed scheduling can be further applied to other video coding standards for a cost-effective SOC implementation.

1. INTRODUCTION

MPEG-4 [1] targets to cover a wide range of multimedia applications. Due to this generality, MPEG-4 itself comprises a toolbox of coding functionalities. The selection and combination of specific coding tools are organized in profiles, which are separately designed for different applications, and a number of levels, which define source and channel parameters to be applicable. To give considerations to both flexibility of implementation and cost effectiveness, a combination of an embedded RISC processor and several dedicated hardware accelerators is adopted for an MPEG-4 codec system implementation [2][3]. While embedded RISC processor provides more programmability, the dedicated hardware is most suitable for video signal processing such as motion estimation (ME) and discrete cosine transform (DCT) to reduce cost and power consumption.

As in MPEG-1, 2, 4 and H.26x series, DCT and ME module have become integral parts of these hybrid coding standards and already drawn a high degree of optimization for different implementations. Compared with ME, which requires a large amount of data from neighboring frames, DCT, coupled with quantization (Q), inverse quantization (IQ) and inverse discrete cosine transform (IDCT), requires only luminance and chrominance data of one macroblock (MB) and all these functions can be integrated into a more specific module, block engine (BE), which is capable to perform texture coding on a local block level. These dedicated hardware units within BE may be all available and optimized in component level already, but still require attentions when they are integrated into system level and working with other components. For example, it may involve overhead, such as large size of buffers, to integrate all these dedicated components that perform different functions into BE.

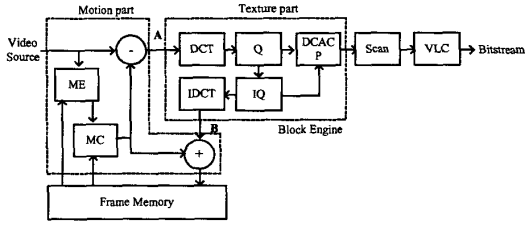


Fig.1. Block diagram of MPEG-4 encoder.

In this paper, we focus on BE that forms a dedicated module for performing all the operations on a block basis in the MPEG-4 video standard, including DCT, Q, IQ, IDCT, and adaptive DC/AC prediction. We propose an efficient scheduling scheme that can integrate all components with least buffers and can perform adaptive DC/AC prediction with minimum hardware cost with the aid of sub-structuring sharing technique.

This paper is organized as follows, in Sec. 2, the detailed functionalities of BE will be introduced. The proposed schedule and the method to implement DC/AC prediction using sub-structure sharing will be given in more details in Sec. 3 and 4. Sec. 5 is the evaluation for our work and Sec. 6 conclude this paper.

2. BLOCK ENGINE

BE performs DCT, Q, IQ, and IDCT, which compose of the texture coding loop in MPEG-4, and where IQ and IDCT are the embedded decoding routines as that in the decoder's side. A completely new coding tool in MPEG-4, the adaptive DC/AC prediction, is also included for improving the coding efficiency of intra MB in both I-VOP and P-VOP. The inputs to BE are block data of intra MB or motion compensated (MC) prediction error of inter MB. The output of IDCT module will be used to form the reconstructed frame data and will be stored in the frame memory and used in ME module for the next frame. The quantized DCT coefficients after DC/AC predictions will be sent to the VLC module in a predetermined scan order. The overall block diagram of MPEG-4 encoder is as shown in Fig.1.

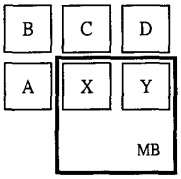


Fig.2. Location of neighboring blocks.

As shown in Fig.2, adaptive DC prediction of block X involves selection of either the quantized DC value of block C or that of the block A. The selected prediction direction is based on the comparison of horizontal and vertical quantized DC value gradients. The DC prediction is given as follows [4],

$$QDC = dc\_coeff \,//\, dc\_scaler \quad (1)$$

Where // represents for rounding away from zero.

if $(|QDC_A - QDC_B| < |QDC_B - QDC_C|)$

$$QDC_{X'} = QDC_C$$

else

$$QDC_{X'} = QDC_A$$

For the adaptive AC prediction, either coefficients from the first row or the first column of a previous coded block are used to predict the co-sited coefficients of the current block. On the block basis, the best direction (from among horizontal and vertical directions) for DC coefficient prediction is also used to select the direction for AC coefficients prediction. In addition, to compensate for differences in the quantization parameter (QP) of the prediction block used for AC prediction and that of the current one, scaling of prediction coefficients becomes necessary. Thus, these predictors are scaled by the ratio of the current QP value and that of the predictor block. The scaling operation is given as follows [4],

$$QAC_{i0} = (QAC_{i0A} \times QP_A) \,//\, QP_X \quad i = 1 \text{ to } 7 \quad (2)$$

$$QAC_{0j} = (QAC_{0jC} \times QP_C) \,//\, QP_X \quad j = 1 \text{ to } 7 \quad (3)$$

Where (2) and (3) are selected by the direction of prediction.

3. PROPOSED SCHEDULE

3.1 Interleaving DCT and IDCT schedule

As shown in Fig.1, the same copy of MC data will be used separately in both point A and B for inter MB. Due to the latency introduced when the input data go through the texture coding loop, DCT-Q-IQ-IDCT, it makes the MC module have to store the produced MC data or to perform the same MC operation twice. It is preferred to use additional local buffers instead of performing MC operation twice since this will consume a large amount of bus bandwidth for memory accessing. As the latency of texture coding loop increases and new block data enters successively, it requires more buffers to store the MC data for a number of block to compensate for the mismatch of timing between texture and motion parts.

Based on the observation above, an efficient interleaving DCT and IDCT schedule is proposed to make IDCT operation start as soon as the DCT coefficients are produced and properly processed. Thus the reconstructed data can be sent to MC module with least latency and the buffer will be available for the next block. As for the Madisetti's architecture [5] we adopt, it can perform two 1-D DCT/IDCT operations in a multiplexing way. So only one hardware unit will be used for both DCT and

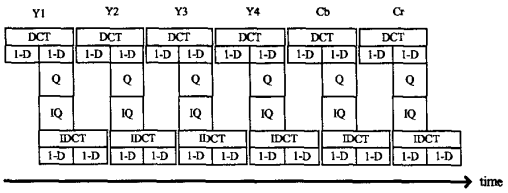


Fig.3. Timing diagram of IDIS.

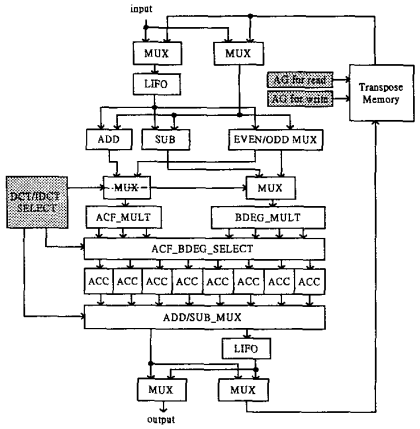


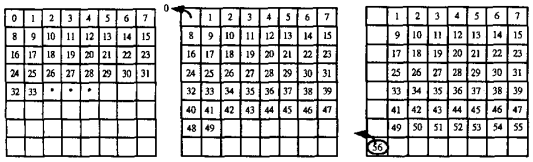
Fig.4. Architecture of DCT/IDCT.

IDCT operation and, however, a different multiplexing method is adopted for implementing IDIS. Since IDCT operation will follow DCT operation immediately in our schedule, the latency introduced by the Q and IQ module at this time is ignored, so the Madisetti's architecture will be used to process both the second 1-D DCT and the first 1-D IDCT at the same multiplexing time slot or vice versa. The timing diagram of IDIS is shown in Fig.3.

3.3. Implementation

IDIS can be achieved by simply adding a switch circuit that selects between the DCT and IDCT operations to the architecture. However, considering the intermediate Q and IQ operations between consecutive DCT and IDCT operations, latency will be introduced to implement the pipelining and meet the specification. Therefore, further modifications are needed. The architecture contains a transpose memory to store the intermediate 1-D data of the DCT/IDCT and two addressing modes, row-by-row and column-by-column, are used alternately. It is shown that a write operation always follows a read operation. But with IDIS, due to the latency introduced by Q and IQ modules as mentioned above, the first 1-D IDCT operation cannot start and multiplex with the second 1-D DCT operation at the same time and in this way the addressing scheme cannot work as the original way because the data to write is not available just after the read operation. Separate address generation (AG) is used to cope with this timing problem. The overall architecture of DCT/IDCT module is shown in Fig.4 and necessary modifications are added in shadowed blocks.

In spite of these modifications, two addressing mode are still the same and a write operation will follow a read operation after



(a) (b) (c)

Fig.5. Read/write actions of transpose memory (a) write in row-by-row mode (b) after writing into address 49, start reading in column-by-column mode (c) after another 8 cycles, the data in address 56 is ready and can be read immediately.

a proper delay. Since read operation is still ahead a write operation, the 1-D data that be read out and used for the next 1-D operation will not be overwritten mistakenly by the write operation.

Another AG takes only minor cost. However, due to this separate addressing scheme, further compacting the number of total processing cycles can be carried out. During the writing cycles of the data to the transpose memory, the data can be read immediately as soon as it is available. Let the addressing of the transpose memory is visualized as that of a 8x8 array and the writing sequence is of the mode of row-by-row, {0, 1, 2, 3, 4, 5, 6, 7, 8, ...}, when the 1-D data is written to the address of 49, the reading procedure can be started right away in the addressing mode of column-by-column,{0, 8, 16, 24, 32, 40, 48, 56, 1, 9, ...}, and after another 8 cycles the data written to address 56 is ready and at next cycle it can be read to proceed another 1-D operation. Afterward, all the data in the transpose memory can be read correctly since it is updated by the write operation already. The compact read/write actions are shown in Fig.5 and it is also valid as in another read/write working mode.

4. DC/AC PREDICTION USING SUB-STRUCTURE SHARING TECHNIQUE

The formulas of adaptive DC/AC prediction are rewritten here for the convenience of following explanation. It is shown in (1), (2) and (3) that extra multiplications and divisions for DC/AC prediction are required.

$$QDC = dc\_coeff \,//\, dc\_scaler \quad (1)$$

$$QAC_{i0} = (QAC_{i0A} \times QP_A) \,//\, QP_X \quad i = 1 \text{ to } 7 \quad (2)$$

$$QAC_{0j} = (QAC_{0jC} \times QP_C) \,//\, QP_X \quad j = 1 \text{ to } 7 \quad (3)$$

However, only DC and part of AC predictors require this pre-processing. It means that one suite of dedicated hardware for multiplications and divisions will only result in poor hardware utilization. So, sub-structure sharing technique is applied for the design of adaptive DC/AC prediction to remove this extra hardware cost. Sub-structure sharing is to extract the same terms in formulas for DC/AC prediction and in that of the existing Q and IQ operations. Observing the proposed IDIS, as the block engine process the first 1-D DCT operation, the Q module is in an idle state, as shown in Fig.3, and is suitable and sufficient to perform the divisions that DC/AC prediction require due to the division-like operations that the Q module owns. The quantization of DC coefficients in an intra MB is given as follows,

$$level = dc\_coef \,//\, dc\_scaler \quad (4)$$

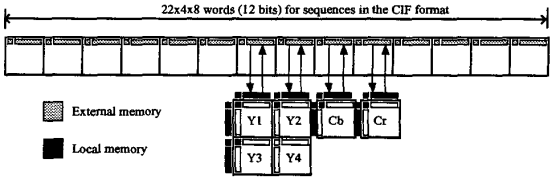


Fig.6. Local prediction buffer in one MB.

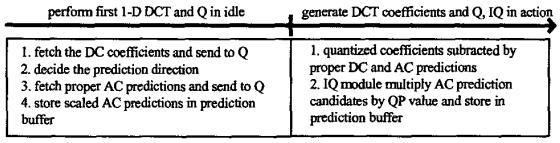


Fig.7. Implementation of DC/AC prediction using sub-structure sharing with Q and IQ module.

By sending proper input to Q module, i.e. three DC coefficients for (1) and the candidates of AC predictors for (2), (3) from neighboring blocks, and adjusting the divider of (4), i.e. dc_scaler and QP values of current block for (1) and (2),(3), respectively, then Q module can be configured to fulfill the divisions that DC/AC prediction requires, which is that in (1) and the later half of (2) and (3). All of these operations can be done before the second 1-D DCT operation starts, in that time DCT coefficients will be generated and then quantized and subtracted by the proper DC and AC predictors. When performing the IQ operation, multiplying the quantized AC coefficients by the QP value, which is the core operation of the IQ module and makes up the front half of (2) and (3), can be extracted intentionally and the results can be stored in the prediction buffer in advance. In this way, when processing the current block, the AC predictors that is properly multiplied by the QP value of that block can be fetched from the prediction buffer and what left of the scaling operation is to be divided by the QP value of the current block, and this can be done using the Q module as mentioned previously. In this way, the QP values of the previous blocks need not to be stored and all multiplications and divisions required for DC/AC prediction are performed by using sub-structure sharing with Q and IQ module.

A prediction buffer is required to store the DC coefficients and all possible candidates of AC predictors at least for an amount equal to the number of MB in a row of a frame. In our design, the prediction data will be stored in external memory and the prediction data needed for the current coding block will be fetched at MB level, resulting in a minimum required local buffer for DC/AC prediction, as shown in Fig.6.In the process of DC/AC prediction, proper replacements should be carried out to replace the data in the local prediction buffer for coding the next block and finally some prediction data will be sent back to the external memory. All needed procedures for implementing DC/AC prediction is shown in Fig.7.

5. EVALUATIONS

In our proposed IDIS, no buffer is required to hold any intermediate data except the transpose memory for DCT and IDCT operations since all the dedicated modules that consist of

Table1. Comparison results with double block scheme and parallel processing scheme.

	Double block pipeline [6]	Parallel processing [3]	Proposed IDIS
No. of DCT/IDCT	1	2	1
No. of Q/IQ	2	Using RISC	2
No. of Block buffer in BE	4	≥6	1
Processing time unit	18	14	13
Latency	3	8	1
No. of Block buffer for MC operation	4	12	2

Note: the processing time unit is counted for processing six blocks of one MB

the loop of texture coding, including DCT/IDCT, Q and IQ modules, are altogether working in a relay method. The output of one module can be directly fed to another. Compared with parallel processing scheme [3], i.e. six blocks of one MB are processed with pipelining, it takes at least one MB size of buffer to hold the intermediate data before it can pass the data to the next stage, for it works at MB level. As to double block pipeline scheme [6], the required number of buffer can be reduced to four block buffers. It is obvious that with IDIS the number of buffer required to integrate DCT, Q, IQ, and IDCT can be reduced substantially to almost none, except only one block buffer which is inherently necessary within DCT/IDCT architecture.

We define block time as the time for 64 inputs to be a basic unit for evaluation. Only 13 time units are required for processing six blocks of one MB in IDIS, as shown in Fig.3. The parallel processing scheme takes 14 time units although it can achieve higher throughput at the cost of using separate DCT and IDCT modules. The double block scheme requires 18 time units. With IDIS, the input block data can go through the texture coding loop with least latency, which is one time unit of latency and it takes two buffers working in ping-pong mode for MC operation in our design. The latency is 3 and 8 time units for double block pipeline and parallel processing scheme respectively and the number of buffers required for MC operation is 4 and 12, respectively. Compared with double block pipeline and parallel processing scheme, the proposed IDIS is more cost-effective and efficient in viewpoint of both the hardware usage and time unit required. The detailed comparison result is listed in table 1. With the compaction effort for IDIS, the required cycles for processing one block can be reduced from 165 to 144 and therefore it takes 935 cycles for one MB. For sequence of 720x480 at 30fps, our design can meet the real-time requirement while working at 54MHz.

With proposed IDIS, the hardware cost for DC/AC prediction is reduced a lot by using sub-structure sharing with Q and IQ module to perform the multiplications and divisions required for it. Besides, the buffer to store the previous QP values is eliminated and a minimum local buffer is achieved by applying DMA to exchange prediction data. Fig.8 is the comparison of sub-structure sharing method with our previous version of hardwired implementation. The gate count of our BE is listed in table.2.

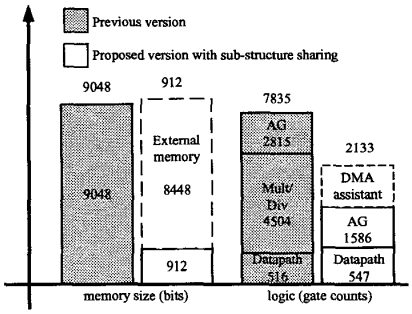


Fig.8. Comparison of DC/AC implementation.

Table2. Gate count of BE at 54 MHz.

Modules	Gate Counts
DCT/IDCT	16,511
Q	5,000
IQ	3,403
DC/AC prediction	2,133
Control	2,227
Total	29,274

6. CONCLUSION

In this paper, IDIS for MPEG-4 video texture coding is proposed. With IDIS, a cost-effective BE to perform DCT-Q-IQ-IDCT coding loop and DC/AC prediction is implemented. Compared with previous work, the coding loop after optimization contains no buffers and least latency, which in turn makes the number of buffer for MC a minimum of two. By the characteristics of IDIS, we apply sub-structure sharing technique for DC/AC prediction with Q and IQ to reduce hardware cost further. Our BE meets the real-time requirement for encoding sequences of 720x480 at 30fps with only 29,274 gates while working at 54MHz.

REFERENCE

[1] ISO/IEC 14496-2:1999/Amd.1:2000, "Coding of Audio Visual Objects – Part 2: Visual, Amendment 1: Visual Extensions," Maui, Dec. 1999.

[2] H.-J. Stolberg, M.Berekovic, P.Pirsch, H. Runge, H. Moller and J. Kneip, "The M-PIRE MPEG-4 CODEC DSP and its Macroblock Engine," *Proc. IEEE Int. Symp. Cir. Syst. (ISCAS)*, May 2000.

[3] M.Takahashi *et al.*, "A Scalable MPEG-4 Video Codec Architecture for IMT-2000 Multimedia Applications," *Proc. IEEE Int. Symp. Cir. Syst. (ISCAS)*, May 2000.

[4] ISO/IEC JTC1/SC29/WG11, N3908, "MPEG-4 Video Verification Model version 18.0," Pisa, Jan. 2001.

[5] A. Madisetti and A.N. Willson, "A 100 MHz 2-D 8x8DCT/IDCT processor for HDTV applications," *IEEE Trans. Cir. Syst. Video Technol.*, Vol.5, No.2, pp. 158-165, Apr.1995.

[6] Y.KATAYAMA, T.KITSUKI, Y.OOI, "A Block Processing Unit in a Single-Chip MPEG-2 Video Encoder LSI," *Journal of VLSI Signal Processing 22*, pp.59-64, 1999.