行政院國家科學委員會專題研究計畫　成果報告

下一代國際視訊壓縮標準（MPEG AVC/H.264）在嵌入式系統實作之分析

計畫類別：個別型計畫
計畫編號：NSC91-2213-E-009-146-
執行期間：91年11月01日至92年07月31日
執行單位：國立交通大學資訊工程學系

計畫主持人：蔡淳仁

計畫參與人員：王岳宜，邱正男，嚴梓鴻

報告類型：精簡報告

處理方式：本計畫可公開查詢

中　華　民　國 92年10月31日

行政院國家科學委員會補助專題研究計畫 ☑成果報告
□期中進度報告

下一代國際視訊壓縮標準(MPEG AVC/H.264)
在嵌入式系統實作之分析

計畫類別：☑ 個別型計畫　　□ 整合型計畫
計畫編號：NSC　91－2213－E－009－146－
執行期間：　92 年 11 月 1 日至　　92 年 7 月 31 日

計畫主持人：蔡淳仁
共同主持人：
計畫參與人員：王岳宜、顏梓鴻、邱正男

成果報告類型(依經費核定清單規定繳交)：☑精簡報告　□完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、
列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立交通大學資訊工程系

一、中英文摘要

關鍵詞：視訊壓縮、嵌入式系統、SoC、MPEG-4 AVC、H.264

ISO 和 ITU-T 在過去十多來年發展出了一系列的視訊壓縮標準，包括 MPEG-1 Video、MPEG-2 Video、MPEG-4 Visual、H.261、和 H.263 等等．這些視訊壓縮標準在制訂之時的一個重要的考量因素是壓縮演算法的複雜度必須是能夠使用價格合理的硬體設備來完成. 因此, 許多時候, 一些複雜但可提高壓縮效率的演算法就被排除在這些國際標準之外.

ISO 和 ITU-T 於西元 2001 十二月決定合作制訂下一代的視訊壓縮標準．這個新的標準在制訂時的原則包括：高壓縮效率、放寬壓縮演算法的複雜度限制、以及將壓縮編碼（source coding）和強韌性機制 （error-robustness mechanism）分開設計．ISO 和 ITU-T 希望未來這個視訊壓縮標準能夠慢慢淘汰現有的標準而成為單一的國際視訊壓縮標準．

本計畫的目的是對這個新標準在嵌入式系統上（特別是適用於手機上的 System-On-Chip 架構）實作的分析．前面提過，如果把所有 MPEG AVC/H.264 的壓縮工具全部加起來，它的複雜度因為太高，勢必沒辦法放在一般的手機晶片上．如何分析不同工具的記憶體和運算量相對於嵌入式系統的複雜度，如何儘量把所有的工具實作在特定的嵌入式系統上，還有如何在壓縮效率和複雜度之間取捨，是一個很重要的題目．由於目前這個標準才剛完成，未來會有一段時間接受工業界和學術界的意見回饋，本研究計畫的成果將可以提供國際標準組織做為未來修定這個標準的參考，有助於提升本研究單位的國際知名度．另一方面，也可幫助國內工業界發展下一代的手機和無線通訊電子助理．

Keywords: video codec, embedded systems, SoC, MPEG-4, H.264

For the past decades, ISO MPEG and ITU-T have developed several international video compression standards, including MPEG-1 video, MPEG-2 video, MPEG-4 Visual, H.261, and H.263, etc. These video codec standards were designed under the constraint of the hardware capability at the time of standardization. Therefore, many compression algorithms were ruled out because the complexities of these tools were too high, cost-performance wise, for existing hardware. There has been a lot of improvement on software and hardware capabilities for the past few years. Therefore, MPEG and ITU-T started a new video codec project in December of 2001. This new standard is designed under the following guidelines: at least twice the compression efficiency of existing standards, coding efficiency gain more appreciated than complexity-reduction, complexity scalability, and separation of source coding and error-robustness mechanism. On a performance/memory-constrained platform, a tradeoff between coding efficiency and complexity has to be made. The analysis of the performance tools provided by this new standard on an embedded platform will be evaluated in this project. Since ISO and ITU-T are still collecting feedback from the industry and the academia about the new standard, this project can provide useful information to ISO and ITU-T to help shape the new codec. It can also help our industry to development the next generation mobile handsets and wireless devices.

二、前言

With the next generation of video applications in mind, ISO MPEG and ITU-T VCEG started a Joint Video Team (JVT) in December, 2001. The task of JVT is to design a new high performance video coding standard. The outcome, MPEG-4 AVC/H.264 [1], is a hybrid motion-compensated transform video codec that aims to become an all-purpose International Standard for future digital video applications. Initial results show that, at low bitrate range (below 1 Mbps), it has about 30% to 50% coding efficiency gain over existing ISO and ITU-T video standards such as H.263 version 2 and MPEG-4 ASP. At the same time, it also showed outstanding coding performance at high bitrate range (8 Mbps and above). It is generally believed that with decent encoder optimization, H.264 can be used to store HD movies on standard DVD media.

However, the exciting performance of the new codec does not come without a price. Early reports showed that the decoder complexity is roughly three times higher and the encoder complexity, when reasonable tools are used, can easily be ten times higher than, say, an MPEG-4 Simple Profile codec. In this project, we studied the performance of different coding tools of the codec. According to our investigation, two of the most time consuming tools are the motion estimator and the in-loop filter. Therefore, we also designed efficient VLSI architecture for acceleration of these two tools. The final result was published in a conference paper (VLSI Design/CAD 2003).

三、研究目的

The goal of this project is to carry out thorough study on MPEG-4 AVC/H.264. In addition to evaluation of different coding tools, quality and complexity optimization for the encoder will also be investigated. Based on the research, efficient algorithm and VLSI accelerator architecture are designed for efficient embedded systems implementation, either using a software-only approach or a hardware/software co-design methodology.

四、文獻探討

There has been quite a few paper published about the new standard. In particular, there was a special issue of IEEE Transaction on Circuit and Systems for Video Technology on H.264. In [2] and [3], complexity analysis on the baseline decoder and an optimized decoder are presented. However, for our project, we are mainly interested in encoder analysis. [3] and [4], discusses the design of two complicated modules of H.264, namely CABAC and in-loop filter, respectively.

There were also few papers published on the VLSI design of some of the high complexity modules such as motion estimator [11] and in-loop filter [12]. However, based on our study, for a system as complicated as H.264 encoder, the best approach is to adopt hardware/software co-design. A pure VLSI implementation of the codec is too expensive.

On the other hand, a pure software implementation would be too slow for embedded real-time operations. The best approach for implementing an H.264 codec is to adopt hardware/software co-design approach.

五、研究方法

The first step of the investigation is to study the performance gain from different tools of H.264. We used the reference software JM6.1e to perform the analysis. An example is shown in Fig. 1. In Fig. 1, the base conditions (test curve 0) of the test is as follows:

Base test conditions (test curve 0):

- Coding pattern: IBBPBBP …
- Full search motion estimation, enable full quad-tree block partition, no Hadamard transform, one reference frame
- Entropy coder: UVLC
- No Loop filter

Curves 1 through 5 added the following tools respectively: 5 reference frame, CABAC coder, Hadamard transform, in-loop filer, and RDO. As shown in Fig. 3, each tool (except for Hadamard transform) contributes roughly equally to the coding gain.

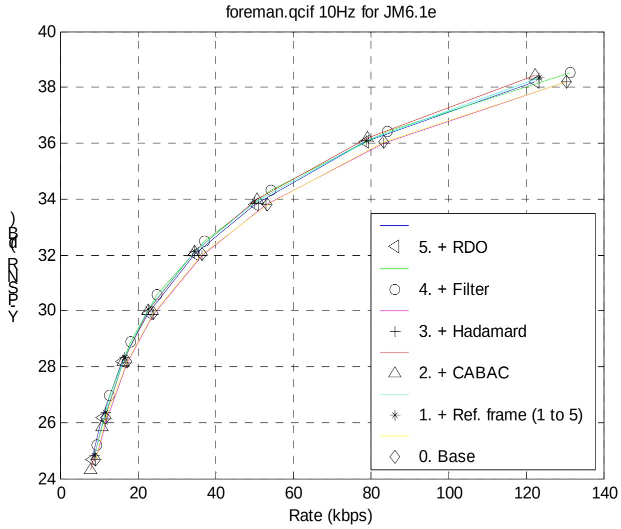

Figure 1. Coding performance of different tools.

After further investigation, we decided to look into a fast algorithm for motion estimation (ME) and see how the new algorithm impacts the coding efficiency. It is important to point out that for old video coding standard such as MPEG-4 SP, a fast ME algorithm can sometimes achieves better quality than full search ME since motion vector overhead can be very high for full search ME. However, with RDO, full search ME can always out perform a fast search ME. The fast algorithm we designed and the performance is discussed in section 6.2. Based on our investigation, it is obvious that for embedded systems, it is very difficult to implement a high quality H.264 encoder without ASIC accelerator. A comparison of memory bandwidth requirement and computational requirement between the ME of H.264 and that of the MPEG-4

SP are shown in Table 1 and 2. Therefore, the next step is to design an efficient architecture for hardware acceleration. The result is presented in section 6.3.

Memory Requirement		Memory Size	
		MPEG-4 SP	H.264 Baseline
Current Frame		384x320	384x320
Reference Frame		384x320	3x384x320
Search Window			
	Integer-pel	48x48	48x48
	Half-pel	8x8x8x4	8x4x4x16
	Quad-pel	N/A	8x4x4x16
Current MB		16x16	16x16
Total Memory		250368	498176

Table 1. Memory Requirement in Bytes

Operation	Number of 4x4 SAD (30 fps)	
	MPEG-4 SP	H.264 Baseline
8x8 (Int-pel)	320x10^{6}	N/A
8x8 (Sub-pel)	1.5x10^{6}	N/A
4x4 (Int-pel)	N/A	385x10^{6}
4x4 (sub-pel)	N/A	3x10^{6}(full,1/2,1/4-pel)
Total SAD	321.5x10^{6}	388x10^{6}

Table 2. #SAD for SP and H.264 Baseline

六、結果與討論

In this section, we presented a fast motion estimation algorithm in section 6.1 and an efficient VLSI architecture for hardware accelerator in section 6.2.

6.1 Fast motion estimator for H.264

For H.264 encoders, there are two major search dimensions. The first one is the spatio-temporal dimension (that is, determining both the reference frame(s) and the best motion vectors in the target reference frame). The second search dimension is the motion partition dimension (that is, determining block partition types and the corresponding vectors for each sub-partition), which is usually referred to as the mode decision problem. A hierarchical approach in spatio-temporal dimension and a bottom-up merge approach in motion partition dimension are proposed here to reduce the number of candidate match points.

Furthermore, when performing sub-pixel motion estimation (1/2-pel or 1/4-pel), simple on-the-fly bilinear transform interpolation instead of the full 6-tap filter plus linear interpolation **錯誤! 找不到參照來源。** is used to compute the SAD for motion estimation since this is a good low-complexity approximation of the coding residual energy.

6.1.1 Integer-pixel hierarchical motion estimation in spatio-temporal domain

The search algorithm first starts with hierarchical 16×16-motion search in spatio-temporal dimension. There have been quite a few literatures on multiple-step search in spatial domain ([6], [7]). We extend this idea across temporal domain for multiple candidate reference frames. As depicted in Figure 2, the best estimate from the most recent reference frame is used as the initial search point in the 2nd-recent reference frame, and the search window in that frame is significantly reduced. The rationale behind this idea is that the effectiveness of multiple reference frames for most sequences reduces rapidly along the reverse temporal direction [8].

6.1.3. Fast sub-pixel vector and partition mode determination

Once the best 16×16 motion vector is located in spatio-temporal domain, a second stage is applied to determine motion partition mode and sub-pixel motion vectors. Naturally, this is a suboptimal solution to the general multi-dimension search problem. To reduce the possibility of being trapped in local minimum, M best-match candidates can be used as initial estimates for the second stage of motion search. The number M can be chosen in real time based on the computational load.

There is a large amount of combinations of macroblock partition modes. Since the focus of the paper is about motion compensated partitions, intra-block modes (either 16×16 or 4×4) are ignored here for the simplicity of discussion. All together, there can be 2+44 motion sub-partition modes for each macroblock (excluding 16×16 mode). Obviously, the search space is too large for an embedded encoder. The key idea here is to perform only the 16 4×4 motion estimation for each macroblock using the integer-pixel motion vector as the initial guess with a search space of ±2 pixels. The resulting 4×4-block sub-motion vectors and the corresponding SADs will be used to decide the macroblock partition mode. For example, if two neighboring 4×4 blocks has similar (subject to some threshold) motion vectors, they will be merged into a single 4×8 (or 8×4) partition. The threshold must take into account both the motion vector coding overhead and the SAD values.

Once the partition mode is determined, sub-pixel motion estimation will be performed for each sub partition. Note that after the partition determination stage, we may end up with a single 16×16 partition because all the 4×4 sub-partitions are merged. The search for sub-pixel motion vector is done around the previously computed integer motion vector for each sub-partition, with a ±1 integer search space. Both half-pixel and quarter-pixel positions in this space will be searched. As mentioned before, the interpolation process used here is simply on-the-fly bilinear transform instead of the full 6-tap filtering plus linear interpolation as specified in the standard. This way, we can greatly reduce both the memory complexity and computational complexity.

6.1.4 Experiments

The standard test sequences FOREMAN and STEFAN are used to show the performance of the proposed algorithm (Fig. 2). Both sequences are in QCIF resolution, 15 fps. For H.264, the encoder is based on JM6.1e. RD optimization, three reference frames, IPPP… coding pattern, and CAVLC are used. Motion search range is ±16, and all subblock types are used. In the proposed approach, a step size of 8 is used for TWSS. Three best candidates are selected for the next level TSS refinement. The step sizes for TSS are 3, 2, 1, respectively for the 1st, 2nd, and 3rd steps. On-the-fly constrained bi-linear transform is not used in these experiments. The results for MPEG-4 Simple Profile using MS reference software are also provided as a comparison.

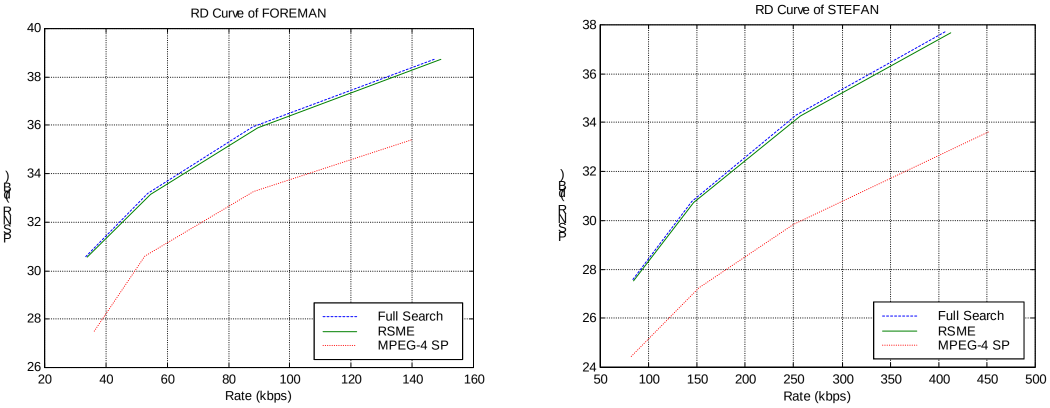

Figure 2. FOREMAN and STEFAN test results

6.2 VLSI Architecture Design

Due to the high complexity of H.264, it is very difficult to have a embedded real-time solution without resorting to hardware accelerators. For this purpose, we also investigated and designed efficient VLSI architecture for ME of H.264. The architecture of the motion estimator is shown in Fig. 3. The current and reference frame buffers are stored in the external memory (SDRAM). Up to four frames can be stored in the frame buffers at the same time during motion estimation and in-loop filtering. YCBCR data is separately stored in the memory. Motion estimation is performed on the Luma Y-channel. The Memory Ctrl module handles the interface with memory modules and the EBUS bus (AHB bus protocol is used for the EBUS in the platform). Two internal memory buffers are used for storing the search windows. To fully utilize the memory for motion estimation and to meet the complexity of memory bandwidth for multiple reference frames access for H.264 encoders, the write/read pipeline arrangements of the two memory buffers for loading reference search window and output to those SAD units for motion vector search during the integer-pixel ME among three reference search windows for one macroblock are specified in Fig. 4.

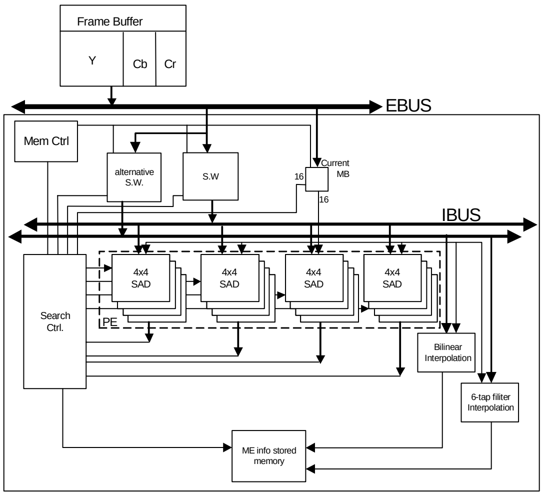

Figure 3. Architecture of Motion Estimation Design

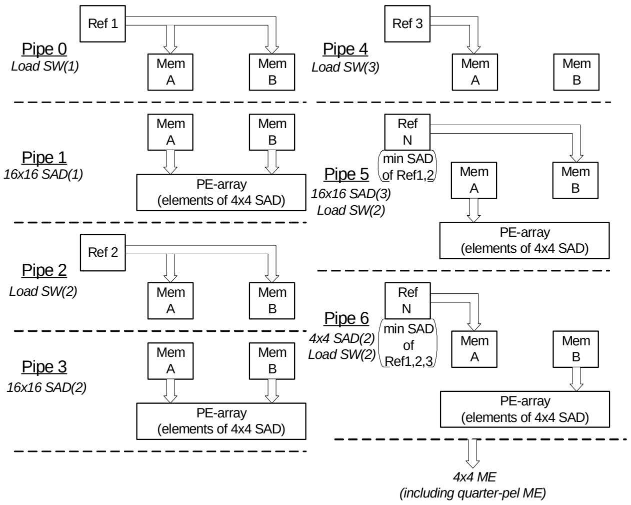

Figure 4. Memory pipeline control for 16x16 ME

七、參考文獻

[1] ISO/IEC JTC 1/SC 29/WG 11, *ISO/IEC 14496-10 Information technology- Coding of audio-visual objects- Part 10: Advanced Video Codec, FDIS*, ISO/IEC N5555, Pattaya, March, 2003.

[2] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis," *IEEE Trans. Circuits Systems for Video Technology*, vol. 13, No. 7, pp. 704-716, July. 2003.

[3] V. Lappalainen, A. Hallapuro, and T.D. Hamalainen, "H.264/AVC baseline profile decoder complexity analysis," *IEEE Trans. Circuits Systems for Video Technology*, vol. 13, No. 7, pp. 717-725, July. 2003.

[4] P. List, A. Joch, J. Lainema, G. Bjntegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Trans. Circuits Systems for Video Technology*, vol. 13, No. 7, pp. 717-725, July. 2003.

[5] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," *IEEE Trans. Circuits Systems for Video Technology*, vol. 13, No. 7, pp. 717-725, July. 2003.

[6] R. Li, B. Zeng, and M.L. Liou, "A new three-step search algorithm for block motion," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 4, No. 4, pp. 438-442, Aug. 1994.

[7] L.-M. Po and W.-C. Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation," *IEEE Trans. Circuits Systems for Video Technology*, vol. 6, No. 3, pp. 313-317, Jun. 1996.

[8] J. Boyce, "Coding Efficiency of Various Numbers of Reference Frames," *JVT-B060*, 2nd Joint Video Team Meeting, Geneva, Switzerland, Jan. 29-Feb. 1, 2002.

[9] F. H. Cheng, S.-N. Sun, "New Fast and Efficient Two-Step Search Algorithm for Block Motion Estimation," *IEEE Trans. Circuits for Systems Video Technology*, vol. 9, No. 7, pp. 977-983, Oct. 1999.

[10] Peter Kuhn, “Algorithms, Complexity Analysis and VLSI Architecture for MPEG-4 Motion Estimation,” *KLUWER ACADEMIC PUBLISHERS*, pp. 8, 1999

[11] Y-W Huang, T-C Wang, B-Y Hsieh, and L-G Chen, “Hardware Architecture Design for Variable Block Size Motion Estimation in MPEG-4 AVC/JVT/ITU-T H.264,” ISCAS 2003, Bangkok, Thailsnd, p. II-796, May 2003.

[12] Y-W Huang, T-W Chen, B-Y Hsieh, T-C Wang, T-H Chang, and L-G Chen, “Architecture Design for Deblocking Filter in H.264/JVT/AVC,” ISCAS 2003, Bangkok, Thailsnd, p. I-693, May 2003.

[13] T. Koga, K. Iinuma, A. Hirano, Y. Ijima, T. Ishiguro, “Motion compensated interframe coding for video conferencing,” *in Proc. Of NTC*, Dec. 1981

[14] Yuei-Yi Wang and Chun-Jen Tsai, “An Efficient VLSI Architecture for Fast Motion Estimator for MPEG-4 AVC/H.264 Encoders,” *Proceedings VLSI Design/CAD 2003*, Hua-Lian, Taiwan, August 2003.

八、計劃成果自評

- 研究內容與原計畫相符程度：
  十分相符.
- 達成預期目標情況：
  對 H.264 的複雜度及分項工具的效能分析均已完成，快速軟體演算法主要只設計了 Motion Estimator，另外，在最初計畫提案未提出而有執行的是設計了 H.264 的 VLSI 加速器架構.
- 研究成果之學術或應用價值
  複雜度及分項工具的效能分析部份同時具有學術參考價值及應用價值，另外，加速器的設計具有應用價值.
- 是否適合在學術期刊發表或申請專利
  本研究成果發表了一篇研討會論文[14]，目前正在延續此研究方向，整理撰寫期刊論中.
- 主要發現或其他有關價值
  主要研究發現有二. 一是 H.264 效能提昇的兩大主要工具是 CABAC 以及最小以 4x4 block 為單位的 motion mode partition. 至於其它單一工具對效能的幫助雖有，但並不算很高. 另外在 VLSI 的實作上，最大的挑戰是 memory bandwidth 的需求太高，未來會延續這一方面的研究，以期設計出更有效率的架構.