A Low-cost Media-Processor based Real-Time MPEG-4 Video Decoder

Jin-Hau Kuo, Ja-Ling Wu, *Senior Member, IEEE*, Jim Shiu*, and Kan-Li Huang

Communications and Multimedia Laboratory,
Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan,
**SetaBox Technology Corp. (STC), Taipei, Taiwan*

ABSTRACT

In this paper, by implementing a ***realistic*** *MPEG-4 core profile decoder over the TriMedia*^{TM} *DSP chip [7], we investigate a serious of important issues related to the use of a media-processor to satisfy multimedia compression and processing requirement. The realistic MPEG-4 decoder optimized implementation by this DSP chip can support resolutions up to 4CIF (720 x 576) at 30 fps. The issues we investigated include the overview of characteristics of media-processor architecture [8-15], the speedup of the MPEG-4 decoder based on the TriMedia*^{TM} *DSP chip, multimedia co-processor or hardware accelerator based approaches, Very Long Instruction Word (VLIW) and Single Instruction Multiple data (SIMD) programming technique and the media-processor based solution for set-top box applications.*

INTRODUCTION

Along with the high-speed development of semi-conductor technology, improvements in microprocessor architecture and performance have enabled general-purpose computer chips to process digital video, audio and graphics in recent years. But as the concept of Information Appliance (IA) has emerged, the similar functionality has shifted from the general-purpose to the so-called media-processor one. The processor is targeted to mass-produced ***consumer electronics*** devices, such as digital televisions, set-top boxes, DTV and as much qualifies as an embedded processor. Meanwhile, the progress of video compression techniques is maturing after years of efforts on development of many standard and de-facto codecs. The well-known moving picture expert group (MPEG) has worked out the newest version of video codec, MPEG-4 [1], possessing very low bit-rate and, yet acceptable quality. Recently, much research [2-6] has focused on considering the media-processor for these multimedia compression and processing requirements, such as the MPEG-4 decoder using a media-processor. The compelling reasons for it are flexibility and low-cost. A media-processor-based terminal architecture has to provide a cost effective (based on commodity pricing) execution environment for video applications because silicon resources are limited. Therefore, there are many challenges in designing and programming media-processors.

THE HIERARCHY ACCELERATION AND THE RELATED METHODOLOGIES

Two years prior to this work, we investigated several issues centered on the MPEG-1, 2 video decoder paired with the Intel MMX instruction set [16]. Relying on our previous findings, we first investigate the issue of speeding up the MPEG-4 decoder using the *TriMedia*^{TM} *DSP chip*. The steps of acceleration can be considered as a hierarchy stage shown in Fig. 1. From top to bottom are the stages of *Algorithm*, *Instruction Set* and *Assembly Level*, respectively. As expected, based on previous research, the inverse discrete cosine transform (**IDCT**) and motion compensation (**MC**) are the two most processing-intensive tasks of MPEG-4 video decoder (As shown in the performance profile in Fig. 2). The performance profile is obtained by porting a pure C code and non-accelerated MPEG-4 decoder over the *TriMedia*^{TM} *DSP chip* preliminary. After that, we demonstrated how to speed up MPEG-4 video decoder according to the VLIW architecture. First, we show how to use data and instruction parallelism by VLIW to enhance the inverse discrete cosine transform (**IDCT**), followed by then how to speed up the motion compensation (**MC**) module. Thirdly, we demonstrated how operations in the MPEG-4 video decoder are also enhanced by VLIW instruction set, as well as using the cache system of *TriMedia*^{TM} *DSP chip* to exploit data locality. The methods for speed-up **IDCT** can summarize as (1) Fast algorithm [17] (2) Loop unrolling - to unroll **IDCT** source code, we can schedule this module into VLIW architecture more compactly (3) In-place matrix transpose – just transposing matrix in place to avoid the matrix transpose in **IDCT** module. (4) *TriMedia*^{TM} SIMD instruction. The approaches to optimize the **MC** module we used are as (1) Loop unrolling (2) *TriMedia*^{TM} SIMD instruction. By using SIMD instructions, we can do multiple motion compensation at a time. But some overheads caused by the lacks of some frequently used SIMD instruction supports by the Intel **MMX & SSE** instruction set reduce the performance improvement (c.f. Fig. 3). These methods conform to the concept of the hierarchy stages of acceleration.

0-7803-7300-6/02 $17.00 © 2002 IEEE

CONCLUSION

We compared the similarities and dissimilarities of the Intel and *TriMedia™ DSP chip*. The drawbacks and the further possible improvement of the DSP chip are also discussed. By cooperation with the *SetaBox Technology Corp. (STC)*, we also investigated *the media-processor in set-top box application*. This includes the current and potential market, as well as making practical solutions for *set-top box* application.

REFERENCES

[1] ISO/IEC International Standard 14496, Information Technology – Generic Coding of Audio-Visual Objects – Part 1: System, Part 2: Visual, Part 3: Audio, Part 5: Reference software, Part 6: DMIF. 2000.

[2] Le, T.M., Nita, A., Giernalczyk, E., Denny Wong, Tuan Ho, "A low-power and scalable MPEG-4 solution for wireless communications," in IEEE Int. Conf. Consumer Electronics, 2001

[3] Fermo, A., Sicuranza, G.L., Pahor, V., "Hardware-oriented region based algorithm for low power motion estimation," in IEEE Proceedings of the 2nd International Symposium on Image and Signal Processing and Analysis, 2001

[4] Ramkishor, K., Gunashree, V., "Real time implementation of MPEG-4 video decoder on ARM7TDMI," in IEEE Proceedings of International Symposium on Intelligent Multimedia, Video and Speech Processing, 2001

[5] Burleson, W., Jain, P., Venkatraman, S., "Dynamically parameterized architectures for power-aware video coding: motion estimation and DCT," in IEEE Proceedings of the 2nd International Workshop on Digital and Computational Video, 2001

[6] "Special Issue on Multimedia Implementation," Call For Papers in IEEE Transaction on Circuits and System for Video Technology, to appear on March, 2002

[7] Philips TriMedia™ Software Development Environment Version 2.1.

[8] Pol, E.J.D., Aarts, B.J.M., van Eijndhoven, J.T.J., Struik, P., Sijstermans, F.W., Tromp, M.J.A., van de Waerdt, J.W., van der Wolf, P., "TriMedia CPU64 application development environment," in IEEE Int. Conf. On Computer Design, 1999

[9] Van Eijndhoven, J.T.J., Sijstermans, F.W., Vissers, K.A., Pol, E.J.D., Tromp, M.I.A., Struik, P.; Bloks, R.H.J., van der Wolf, P., Pimentel, A.D., Vranken, H.P.E., "TriMedia CPU64 architecture," in IEEE Int. Conf. On Computer Design, 1999

[10] Equator Technologies Inc. http://www.equator.com

[11] Texas Instrument Technologies Inc. http://www.ti.com

[12] Intel Architecture Optimization – Reference Manual

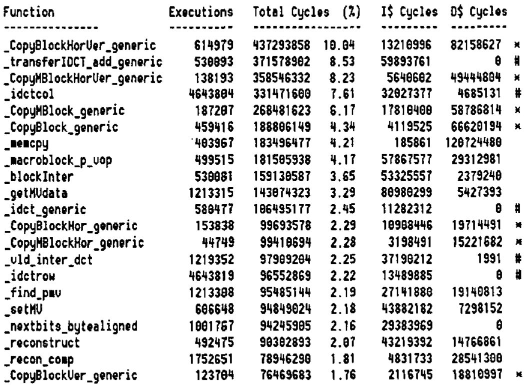
[13] Intel Architecture MMX New Instruction Technology

[14] Intel Architecture SSE (Katmai) New Instruction Technology

[15] MPEG Macroblock Parsing and Pel Reconstruction on an FPGA-augmented TriMedia Processor

[16] Yi-Shin Tung, Chia-Chiang Ho, Ja-Ling Wu, "The MMX-based IDCT and MC Algorithms for Real-Time Pure Software MPEG Decoding," ICMCS'99, Florence Italy

[17] Y. Arai, T.Agui, and M.Nakajima, "A Fast DCT-SQ Scheme for Images", Trans. Of the IEICE, E71(11):1095, Nov 1988.



```
Function                    Executions   Total Cycles   (%)   I$ Cycles  D$ Cycles
---------------             ----------   -----------------    ---------  ---------
_CopyBlockHorVer_generic        614979   437293858   10.04     13210996   82158627 *
_transferIDCT_add_generic       530093   371578902    8.53     59893761          0 #
_CopyMBlockHorVer_generic       138193   358546332    8.23      5640602   49444804 *
_idctcol                       4643804   331471600    7.61     32027377    4685131 #
_CopyMBlock_generic             187207   268481623    6.17     17810400   58786814 *
_CopyBlock_generic              459416   188806149    4.34      4119525   66620194 *
_memcpy                         403967   183496477    4.21       185861  120724480
_macroblock_p_vop               499515   181505938    4.17     57867577   29312981
_blockInter                     530081   159130587    3.65     53325557    2379240
_getMVdata                     1213315   143074323    3.29     80980299    5427393
_idct_generic                   580477   106495177    2.45     11282312          0 #
_CopyBlockHor_generic           153838    99693578    2.29     10908446   19714491 *
_CopyMBlockHor_generic           44749    99410694    2.28      3198491   15221682 *
_vld_inter_dct                 1219352    97909204    2.25     37190212       1991 #
_idctrow                       4643819    96552869    2.22     13489885          0 #
_find_pmv                      1213308    95485144    2.19     27141880   19140813
_setMV                          606648    94849024    2.18     43882182    7298152
_nextbits_bytealigned          1001767    94245905    2.16     29383969          0
_reconstruct                    492475    90302893    2.07     43219392   14766861
_recon_comp                    1752651    78946290    1.81      4831733   28541300
_CopyBlockVer_generic           123704    76469683    1.76      2116745   18810997 *
```

Figure 2: **The preliminary performance profile of the non-accelerated MPEG-4 decoder over the *TriMedia™ DSP chip***: The columns marked by the symbol * and # are the DSP cycles: instruction cycles and data cycles, executed by the MC and IDCT, respectively.

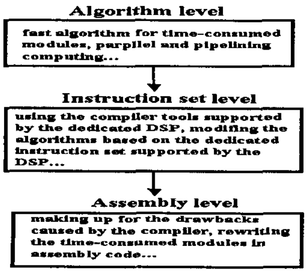
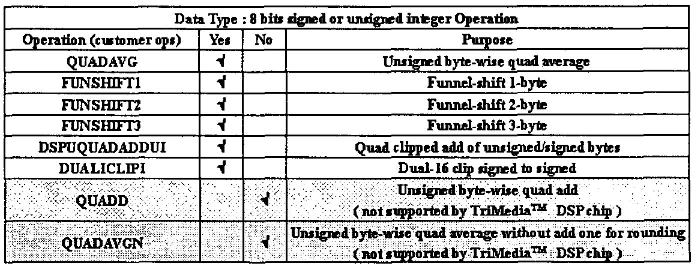

Algorithm level
fast algorithm for time-consumed modules, parpllel and pipelining computing...
Instruction set level
using the compiler tools supported by the dedicated DSP, modifing the algorithms based on the dedicated instruction set supported by the DSP...
Assembly level
making up for the drawbacks caused by the compiler, rewriting the time-consumed modules in assembly code...

Figure 1: **The hierarchy stages of acceleration and the related methodologies**



Data Type : 8 bits signed or unsigned integer Operation			
Operation (customer ops)	Yes	No	Purpose
QUADAVG	✓		Unsigned byte-wise quad average
FUNSHIFT1	✓		Funnel-shift 1-byte
FUNSHIFT2	✓		Funnel-shift 2-byte
FUNSHIFT3	✓		Funnel-shift 3-byte
DSPUQUADADDUI	✓		Quad clipped add of unsigned/signed bytes
DUALICLIPI	✓		Dual-16 clip signed to signed
QUADD		✓	Unsigned byte-wise quad add (not supported by TriMedia™ DSP chip)
QUADAVGN		✓	Unsigned byte-wise quad average without add one for rounding (not supported by TriMedia™ DSP chip)

Figure 3: **The Used Customer Operations in MC**