EFFICIENT IDCT IMPLEMENTATIONS ON VLIW PROCESSORS

Daniele Bagni, Antonio Borneo, Luca Celetto

Advanced System Technology, STMicroelectronics
Via C. Olivetti, 2, 20041 Agrate Brianza (MI), Italy

daniele.bagni@st.com, antonio.borneo@st.com, luca.celetto@st.com

ABSTRACT

In this paper we describe two efficient software implementations of bi-dimensional IDCT (Inverse Discrete Cosine Transform). Instead of using a traditional separation into eight horizontal and vertical mono-dimensional IDCT stages, we apply a novel approach to directly represent the bi-dimensional IDCT into only eight mono-dimensional units followed by a network of addition and subtraction operations. We have then optimized this method in pure ANSI-C for 32-bit architecture VLIW (Very Long Instruction Word) processors. By arranging the network structure in a proper way to exploit sub-word parallelism and by defining totally new multimedia instructions, we have implemented a second version that is 23% more efficient than the previous one. Our fixed-point arithmetic IDCT implementations are fully compliant with the IEEE 1180 standard, as required by most of the video compression standards.

1. INTRODUCTION

In this paper we describe a fast algorithm of 8x8 bi-dimensional IDCT that is then implemented in two versions: the first in pure ANSI-C style, the second based on SIMD (Singe Instruction on Multiple Data) extensions to the C-language. The target processors are ST210 and TM1100, both having 32-bit VLIW CPU architectures.

The organisation of this paper is the following: Section 2 describes the main features of the two adopted VLIW CPUs. Section 3 illustrates the IDCT algorithm we have implemented in software (SW). Section 4 deals with the new SIMD we have defined to increase execution efficiency. Sections 5 and 6 respectively give performance figures and draw our conclusions.

2. TM1100 AND ST210 VLIW CPU CORES

Instruction Level Parallelism (ILP) speeds up programs by executing in parallel several elementary RISC operations, such as memory load and store, integer addition and multiplication. In a VLIW CPU these operations are taken from a single stream of execution, rather than from parallel tasks. Thanks to sophisticated algorithms that extract ILP from applications written in C-language, the compiler schedules (statically) the code to optimally fill most of the functional units of the VLIW CPU [16]. This parallelism is transparent to the user, although the programmer may restructure his code according with some proper rules to help the compiler to achieve a high degree of ILP.

Multimedia applications in digital consumer market require high performance and low cost silicon implementation combined with minimum time-to-market. The current trend of many silicon manufacturers is to build VLIW processors to allow many functions to be implemented as SW algorithms instead of HW circuits, since a customized VLIW-based device can offer greater flexibility at the cost of hard-wired logic.

To this purpose the ST200 family of VLIW customizable processors has been jointly developed by Hewlett-Packard Laboratories and STMicroelectronics [10] and recently presented [11,12]. The family is based on a modular 4-issues architectural block named “cluster”, with seven 32-bit functional units. A cluster can perform up to four integer RISC operations in every clock cycle (with the constraint of a single load/store per cycle), therefore the maximum ILP achievable is 4 and the maximum instruction length is 128 bits. *N* clusters can be connected together to form a CPU with 4x*N* issues. The first test chip of this family, named ST210, is a single-cluster 4-issues CPU, it allows the equivalent of 1GHz RISC performance. The ST210 processor is equipped with 64 general-purpose 32-bit registers, 32KB I-cache and 32KB D-cache memories. A key benefit of ST200 family is that it combines a relatively simple but highly flexible architecture with very sophisticated compilation tools. Therefore, enhancements to the micro-architecture, such as new custom-instructions and number of clusters, can be analyzed and the resulting cost/performance characteristics quickly and reliably investigated before committing to an

architectural decision. In Section 4, we define and propose new multimedia instructions for a future ST210-variant.

The TriMedia family was presented in [13]. Every device is a complete System-On-Chip (SOC) containing a VLIW CPU accompanied by intelligent A/V peripherals. The TM1100 processor [14] has 128 general-purpose 32-bit registers, D-cache size of 16 KB and I-cache of 32 KB. TM1100 has a very different VLIW approach in comparison with ST210 architectural simplicity: there are twenty-seven different functional units but only five of them can be filled in a clock cycle (with a maximum of two load/store operations per cycle). Because of the number of available functional units and their assignment, some operations may have to wait for one or more cycles before they are executed, which means that it is not possible to choose every mixing of such operations. The 133MHz clock frequency allows the equivalent of 666MHz RISC performance on its 5-issues VLIW CPU (without considering its SIMD extensions).

We have emphasized these aspects to show that, although dedicated mainly to ST210, our novel IDCT implementation is suitable for any VLIW processor, being ST210 and TM1100 very different machines.

3. INVERSE DCT

The DCT concentrates most of the energy distribution into a few frequency coefficients. This important propriety makes this transform, together with its inverse (IDCT), a valuable tool for well-known compression standards of still and moving pictures, like JPEG, MPEG-1, MPEG-2, MPEG-4, H-261 and H.263. In these systems the DCT is normally applied bi-dimensionally on a square block of 8x8 pixels. Being a separable transform, an 8x8 DCT can be separated, for example, in eight horizontal mono-dimensional DCT stages (each working on an 8-sample row) followed by eight vertical DCT stages (each working on a column of 8-coefficient previously produced by the horizontal modules). Vice-versa, the decomposition can be done first on vertical and then on horizontal direction; the result is exactly the same because of the DCT linearity property, given sufficient accuracy in fixed-point arithmetic implementation.

In theory, an 8-sample mono-dimensional DCT requires 64 multiplication and 56 addition operations, which makes its usage very expensive in consumer market devices. Therefore, several computationally efficient algorithms have been developed to reduce the DCT complexity. The methods reported on [1,2,3,4,5] represent just few examples among the large variety of articles available in literature and apply the above-mentioned property of separability. For instance, the MPEG-2 Test Model 5 video encoding SW reference model [6,7,8] makes use of the five-stages Wang's IDCT, based on sparse-matrix factorization (this class of matrices has few non-zero coefficients) [1]. Wang's 8-point DCT uses only 29 addition and 11 multiplication operations instead of the 56 plus 64 theoretically required. As we will describe in Section 5, this algorithm can be very efficiently optimized in C-language on VLIW CPUs.

In a recently appeared paper [9], Huang/Wu propose a fast direct bi-dimensional DCT based on index permutation. The mathematical description of such algorithm is beyond the scope of this paper. Coarsely, we can say that an 8x8 bi-dimensional DCT can be computed through eight (8-sample) mono-dimensional DCT units followed by a network of four butterfly stages, as shown in Figure 1. Since the network is independent on the mono-dimensional DCT calculation, any DCT algorithm can be used: in our case we have applied the above Wang's DCT given its good performance on VLIW CPUs. We emphasize that not all the final 8x8 outputs are exactly the required 8x8 DCT values, because of a simplification used in the algorithm; as a consequence, we have put fifteen additional multiplications to rescale the related output values that differ from the required ones (in the last stage "MULs" of Fig 1).

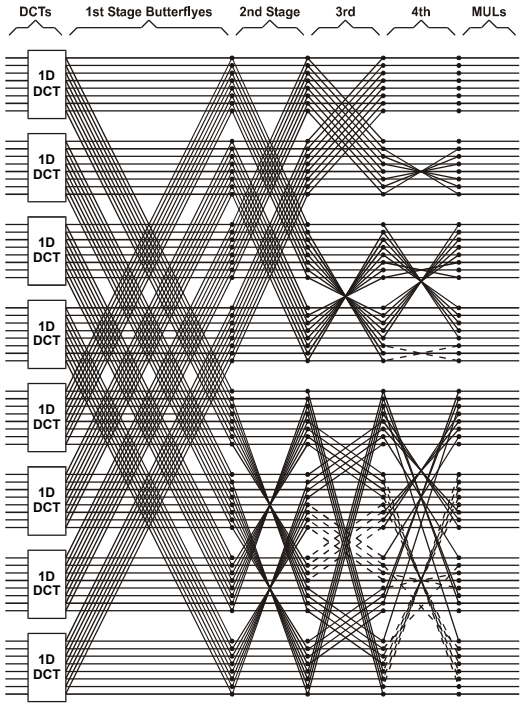

Figure 1: Structure of Huang/Wu direct DCT.

A butterfly is a structure with an addition-subtraction pair of operations, as shown in Figure 2. When we analyzed the algorithm of Huang/Wu, we had the intuition that the network of post-butterflies could be suitable for a SIMD implementation. Effectively, during the fixed-point arithmetic modeling, we realized that a 16-bit intermediate data representation in the network was

enough to provide a DCT suitable for image processing. The possibility to represent the network of butterflies with 16-bit samples, allows us to investigate on sub-word parallelism, by packing 16-bit data into 32-bit words, as a first step to target a SIMD implementation.

We have also inverted the proposed forward DCT in order to obtain its inverse transform. This IDCT can be graphically described by a network of butterflies, followed by eight mono-dimensional IDCT units; it corresponds to the same graph of Fig.1, with flow from right to left.

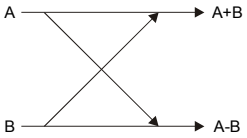

Figure 2: Graphical symbol of a butterfly structure.

4. BUTTERFLY SIMD

Several multimedia applications spend significant amounts of execution time dealing with 16-bit sub-words. Using 32-bit operations to manipulate these small data items makes inefficient use of a 32-bit ALU (Arithmetic Logical Unit). There is little cost difference between a standard 32-bit functional unit that can process one pair of 32-bit operands and a SIMD-enhanced ALU that can also process two pairs of 16-bit operands. If we could apply the 32-bit functional units to operate on two 16-bit data items simultaneously, performance would be improved by a significant factor. In fact, a SIMD is equivalent to several elementary RISC operations and can be issued in a single clock cycle like any other "traditional" operation by the VLIW CPU. However, a C-program that uses SIMD extensions is no more ANSI compliant: in fact the compiler has to manage SIMD instruction as *assembly intrinsic*, that is, built-in C-language functions corresponding to the processor assembly instructions.

The first stage of the network in Figure 1, immediately contiguous to the eight mono-dimensional DCT units, is very regular. With a careful selection of the pairs of 16-bit data within each 32-bit operand, this stage could be implemented by "classical" SIMD instruction like TM1100 *DualAdd* and *DualSub* [13]. Unfortunately, starting from the second stage, the same SIMD instructions are no more easily applicable. A complex set of operations to re-order the packed data is necessary before going on with the following stages, but the overhead of these extra packing/unpacking instructions diminishes dramatically the potential performance gain obtained by *DualAdd* and *DualSub* SIMD.

To solve this problem, since the only regular structure inside the post-butterflies network is the butterfly itself, we mapped it into a new set of SIMD instructions, never used by other microprocessor manufacturers to our knowledge. In order to freely select the input configuration, four possibilities are considered, depending on the desired high/low part of the input operands, named respectively *Butterfly_HH*, *Butterfly_LH*, *Butterfly_LL* and *Butterfly_HL*, as shown in Fig. 3. Each instruction has two input registers that provide the two 16-bit sub-words, the output is a packed 32-bit word composed by two 16-bit signed integer values: one represents the sum of two 16-bit data items, the other one their difference (note that reversing the higher or lower part of the output is transparent for the application and for this new SIMD proposal).

5. PERFORMANCE

To compare the results in a consistent way, we define the "normalized-ILP" as the ratio of "effectively achieved" and "theoretically achievable" ILP values. Therefore, the maximum normalized-ILP is always 1.0, independently on the number of issues of the processor (we remember that ST210 and TM1100 have respectively 4- and 5-issues VLIW CPUs). The IDCT is measured in clock cycles per one block of 8x8 pixels, but these cycles are of VLIW nature and then they will be multiplied by the effectively achieved ILP, in order to obtain the equivalent amount of operations per block from an ideal RISC processor, referred as "op/blk" in the following.

The five-stages Wang's bi-dimensional IDCT [1], optimized in C-language, takes around 1155 ops/blk on ST210 and 1114 ops/blk on TM1100 (without any optimization, the original C-program was about three times less efficient on both processors). The normalized-ILP values are respectively 0.84 (3.36/4.0) and 0.93 (4.66/5.0) for ST210 and TM1100, with related code size of 5240 and 4110 bytes.

Our ANSI-C implementation of Huang/Wu's IDCT requires respectively 980 and 884 ops/blk on ST210 and TM1100 CPUs, with a related normalized-ILP of 0.97 (3.88/4.0) and 0.92 (4.64/5.0). Again, as for the Wang's case, these ILP figures are quite good and demonstrate the efficient usage of the machine resources. However, Huang/Wu's algorithm is better than Wang's one for two reasons: 1) it is less complex and it saves about 15% (ST210) and 20% (TM1100) of ops/blk, 2) it allows code size reduction: 3976 (ST210) and 3226 (TM1100) bytes, which represent a factor respectively of 24% and 21%, depending on the processor.

Although the ST210 does not have SIMD in its instruction set, its tool-chain allows the simulation of new instructions for architectural exploration of its variants. By assuming that any Butterfly SIMD have a latency of 1 clock cycle on the ST210, the SIMD implementation achieves a result of 750 ops/blk with a normalized-ILP of 0.96 (3.84/4.0) and code size of 3088 bytes. In term of

ops/blk, this result represent a performance improvement of about 23%, related to the pure ANSI-C version (for sake of clarity, we emphasize that only the network is implemented with butterflies SIMD, the eight DCT units remain in pure ANSI-C). Unfortunately we could not test these SIMD on the commercially available TM1100 tool-chain, which does not allow to add and to simulate new instructions. If we compare the SIMD-based Huang/Wu implementation with ANSI-C Wang's one, our total gain is 35% in terms of ops/blk. Table 1 shows a performance summary.

Finally, both Wang's and Huang/Wu's (either ANSI-C or SIMD) fixed-point arithmetic implementations are fully compliant with the IEEE 1180 standard [15], as required by almost every picture and video compression standard.

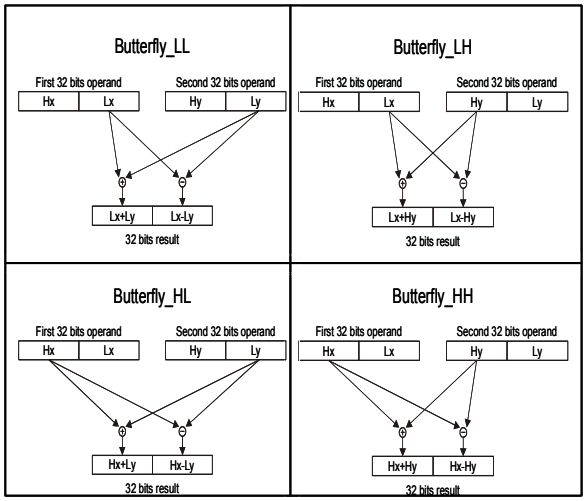

Figure3: The new butterfly SIMD instructions, proposed for an ST210-variant.

6. CONCLUSION

In this paper we presented our fixed-point arithmetic implementation of a direct IDCT, according to Huang/Wu [9], as the composition of eight mono-dimensional Wang's IDCT [1] modules plus a network of butterflies stages, for a pure SW implementation of DCT-based video encoding systems. This algorithm can be mapped very efficiently in ANSI-C language on different VLIW CPU architectures, like ST210 and TM1100, with a performance gain in the range of 15% and 20% when compared with a separable bi-dimensional pure Wang's IDCT [1].

For an ST210-variant, we also defined totally new SIMD instructions that fit the algorithm to the architecture with an efficiency 23% greater than its ANSI-C version and 35% better than the Wang's bi-dimensional IDCT.

algorithm	ST210		TM1100	
	ops/blk	code-size bytes	ops/blk	code-size bytes
Wang (ANSI-C)	1155	5240	1114	4110
Huang/Wu (ANSI-C)	980	3976	884	3226
Huang/Wu (SIMD)	750	3088		

Table 1: Performance summary.

REFERENCES

[1] Z. Wang, "Fast Algorithms for the Discrete W Transform and for the Discrete Fourier Transform", *IEEE Tr. on Acoustic, Speech, and Signal Processing.*, vol. ASSP-32, no. 4, pp. 803-816, Aug. 1984.

[2] W. H. Chen, C. H. Smith, S. C. Fralick, "A fast computational algorithm for the discrete cosine transform", *IEEE Tr. on Communication*, vol. COM-25, no. 9, pp. 1004-1009, Sep. 1977.

[3] Z. Wang, "Reconsideration of A Fast Computational Algorithm for the Discrete Cosine Transform"', *IEEE Tr.on Communications*, vol. COM-31, no. 1, pp. 121-123, Jan. 1983.

[4] X. Wan, Y. Wang, W. H. Chen, "Dynamic Range Analysis for the Implementation of Fast Transform", *IEEE Tr. on Circuits and Systems for Video Technology*, vol. 5, no. 2, pp. 178-180, Apr.1995.

[5] C. Loeffler, A. Ligtenberg, G. S. Moschytz, "Practical Fast 1-D DCT Algorithms with 11 Multiplications", *Proc.of Intern. Conf. on Acoustic, Speech and Signal Processing* (ICASSP-89), pp 988-991, Glasgow, Scotland, May 1989.

[6] ISO/IEC 13818-2, Draft International Standard, May 1994.

[7] ISO-IEC/JTC1/SC29/WG11, "Test Model 5", Apr. 1993.

[8] S. Eckart, C. Fogg, "MPEG-2 Encoder / Decoder", Version 1.2, July 1996, Copyright (c) 1996, MPEG SW Simulation Group. SW available from http://www.mpeg.org/MSSG/.

[9] Y-M. Huang, J-L.Wu, "A Refined Fast 2-D Discrete Cosine Transform Algorithm", *IEEE Tr. on Signal Processing,* vol 47, no. 3, Mar. 1999.

[10] http://www.hpl.hp.com/cambridge/projects/cfp/

[11] P. Faraboschi, G. Brown, J. Fisher, G. Desoli, F. Homewood, "Lx: a technology platform for customizable VLIW embedded processing," *Proc. of the 27th Intern. Symposium on Computer Architecture* (ISCA-27), Jun. 2000, pp 203-213.

[12] P. Faraboschi, F. Homewood, "ST200: A VLIW Architecture for Media-Oriented Applications", *Microprocessor Forum*, Oct. 2000.

[13] B. Case, "First TriMedia chip boards PCI bus", *Microprocessor Report*, vol.9, no. 15, Nov. 1995.

[14] TriMedia TM1100 Preliminary Data Book, March 1999 Second Draft. Philips Electronics North America Corporation.

[15] "IEEE Standard Specifications for the Implementations of 8x8 inverse Discrete Cosine Transform", IEEE Standard 1180-1990, Mar. 18, 1991.

[16] P. Faraboschi, G. Desoli, J.A. Fisher, "The Latest Word in Digital and Media Processing", *IEEE Signal Processing Magazine*, pp 59-85, Mar. 1998.