PIPELINED ARCHITECTURE FOR INVERSE DISCRETE COSINE TRANSFORM

Jari Nikara[1], *Jarmo Takala*[1], *David Akopian*[2], *Jukka Saarinen*[1], and *Jaakko Astola*[1]
[1]Dept. of Information Technology, Tampere University of Technology,
P.O.Box 553, 33101 Tampere, FINLAND
[2]Nokia Mobile Phones, P.O.Box 429, 33101 Tampere, FINLAND
Tel: +358 3 3652111; fax: +358 3 3654575
e-mail: jari.nikara@tut.fi, jarmo.takala@tut.fi, david.akopian@nokia.com, jukka.saarinen@tut.fi, jaakko.astola@tut.fi

ABSTRACT

In this paper, a pipelined architecture for inverse discrete cosine transform (IDCT) is presented. Pipeline architectures are popular in parallel fast Fourier transform implementations but they are rare in IDCT implementations due to the irregularities in fast IDCT algorithms. The proposed architecture is derived by applying vertical projection to in-place IDCT algorithm. The resulting structure is modular and easy to pipeline. The word width requirements in the internal arithmetic are estimated to fulfil the requirements set by IEEE standard for 8×8 inverse cosine transform.

1 INTRODUCTION

Discrete cosine transform (DCT) is a widely used tool in digital signal processing and it is a part of many international standards. In video coding applications, the accuracy of the implementation is important; e.g., IEEE Standard 1180-1990 [1] defines accuracy requirements for two-dimensional 8×8 inverse DCT (IDCT) implementations. Several architectures for such DCT implementations have been proposed for multimedia applications where the DCT circuit should support high data rates.

Typically high speed operation is achieved with the aid of parallelism. Uramoto *et al.* [2] have presented an architecture for 8×8 DCT, which is based on distributed arithmetic. Matsui *et al.* [3] have introduced a DCT macrocell, which is a direct realization of the fast DCT algorithm by Chen *et al.* [4]. Distributed arithmetic and sense-amplifying flip-flops were used to achieve high speed performance with 20 bits. Madisetti and Willson [5] have implemented a small area DCT processor using 22-bit internal accuracy.

In general, the cost of the implementation should be low, i.e., the number of arithmetic units in the architecture should be minimized. For this purpose the linear array processor approach described in [6] can be used. Such pipelined architectures have gained popularity in case of fast Fourier transform (FFT). The parallel architecture is obtained by applying vertical projection to signal flow graph of the algorithm, i.e., the parallel arithmetic operations are mapped onto a single resource thus the data samples are fed to the architecture in sequential fashion rather than in parallel.

Although several pipelined FFT architectures have been proposed, such architectures for DCT are rare due to the irregularity of fast DCT algorithms. Kovac *et al.* [7] have proposed a pipelined architecture based on fast DCT algorithm by Arai *et al.* [8]. Due to the irregularity of the signal flow graph of the algorithm, reordering of the samples is needed several times during the processing, which increases the latency and the number of registers in the architecture.

In this paper, a pipelined IDCT architecture based on the principles used in pipelined FFT architectures is proposed. The architecture is derived by utilizing the constant geometry DCT algorithm described in [9, 10]. By rescheduling the operations in the regular IDCT algorithm, it is possible to obtain an in-place IDCT algorithm. Vertical projection is then applied to the signal flow graph and the resulting dependency graph is mapped onto a hardware structure where no intermediate data reordering is needed. The modular structure is easy to pipeline for increasing the throughput. In addition, error analysis of the architecture for fulfilling the IEEE standard [1] is estimated.

2 IDCT ARCHITECTURE

The basis of our architectural derivation is the constant geometry fast DCT algorithm presented in [9, 10]. The corresponding signal flow graph (SFG) of an 8-point constant geometry inverse DCT algorithm can be seen in Fig. 1, where the coefficients d_i can be generated recursively as

$$\begin{aligned} d_1 &= \sqrt{0.5}, \\ d_{2i} &= \sqrt{0.5(1+d_i)}, \\ d_{2i+1} &= \sqrt{0.5(1-d_i)}. \end{aligned} \quad (1)$$

The constant geometry algorithm lends itself for horizontal projection as described in [9], which leads into a partial-column architecture. Here our purpose is to utilize vertical projection as described, e.g., in [11], where

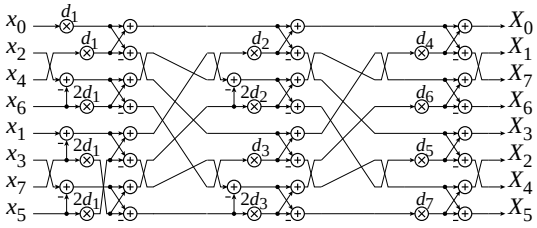

Figure 1: Signal flow graph of 8-point constant geometry inverse discrete cosine transform.

the projection is applied to in-place FFT algorithms. Therefore, the previous SFG for IDCT in Fig. 1 needs to be modified in order to obtain an in-place algorithm. The modification is done by rescheduling the operations in the SFG and one possible in-place schedule for an 8-point IDCT is illustrated in Fig. 2.

The SFG of the in-place algorithm reminds the SFG of the familiar Cooley-Tukey radix-2 FFT algorithm, i.e., the SFG contains the butterfly operations found in FFT. In traditional pipelined FFT architectures, the processing stages of the SFG consisting of parallel butterfly operations are mapped onto separate multiplier and processing elements (PE) containing an adder and subtractor as illustrated in Fig. 3.b). In an in-place algorithm, the data elements are processed in pairs, e.g., in the first butterfly stage in Fig. 2, elements x_i and x_{i+2}, $0 \leq i \leq 7$, are computed at the same time. In the second butterfly stage, elements x_i and x_{i+1} are processed as a pair. Therefore, the data reordering can simply be arranged by delaying the samples with an shift register.

The operation of the processing element is the following. In the stage, where data elements x_i and x_{i+d} are computed in pairs, the first d elements entering the PE are directed to the shift register of size d. When the element x_d is entering the PE, the element x_0 is available from the shift register and, therefore, the butterfly operation can be performed, i.e., $(x_d + x_0)$ and $(x_d - x_0)$ can be computed in parallel. According to the SFG in Fig. 2, the result of subtraction is fed to the output of the PE and the result of addition is directed to the shift register. Such an operation is performed for the next d

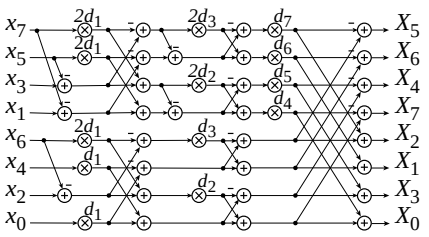

Figure 2: Signal flow graph of 8-point in-place inverse discrete cosine transform.

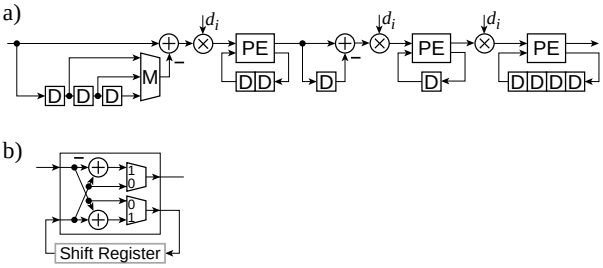

Figure 3: Pipelined architecture for 8-point in-place IDCT corresponding the SFG in Fig. 2: a) block diagram and b) butterfly processing element. D: Delay register. M: Multiplexer. PE: Processing element.

data elements. At this point we have output d results of subtraction and the result of the first addition is available from the shift register, thus next the values from the shift register are directed to the output of the PE and, at the same time, the next d elements entering the PE are directed to the shift register. By continuing this procedure the data reordering can be performed. The advantage of the pipelined architecture is that the multiplier is removed from the critical path, i.e., the feedback loop containing the shift register, and multicycle multipliers can be used to increase the throughput of the implementation.

The SFG of the in-place IDCT algorithm contains some irregularities, which are not present in FFT. For instance, at the first stage of the SFG in Fig. 2, three subtractions need to be performed for element pairs having different offsets. These operations can be realized with a special subtractor, which performs subtraction according to a control signal and otherwise is in bypass mode. The correct data reordering can be implemented with an elastic buffer, e.g., in the fist stage of the previous SFG, subtraction is performed for element pairs having offsets of one, two, and three, thus a feedforward shift register is needed to delay certain elements.

As a result of the vertical projection, according to the previous discussion, the SFG of the 8-point in-place IDCT can be mapped onto a structure illustrated in Fig. 3.a). The computation of butterfly operation is performed in PE in Fig. 3.b), for each input pair. It should be noted that the clock lines are not drawn in the figure. The input as well as the output of the architecture is sequential and the structure can be pipelined arbitrary to increase throughput. The only limiting factor in pipelining is the speed of the adder and subtractor in the PE.

The drawback of the architecture is that the utilization of adders and subtractors during the butterfly computations is only 50%. With some optimizations, it is possible to achieve higher utilization rate of these resources but then, in general, an additional reordering network is needed to maintain the correct order of data

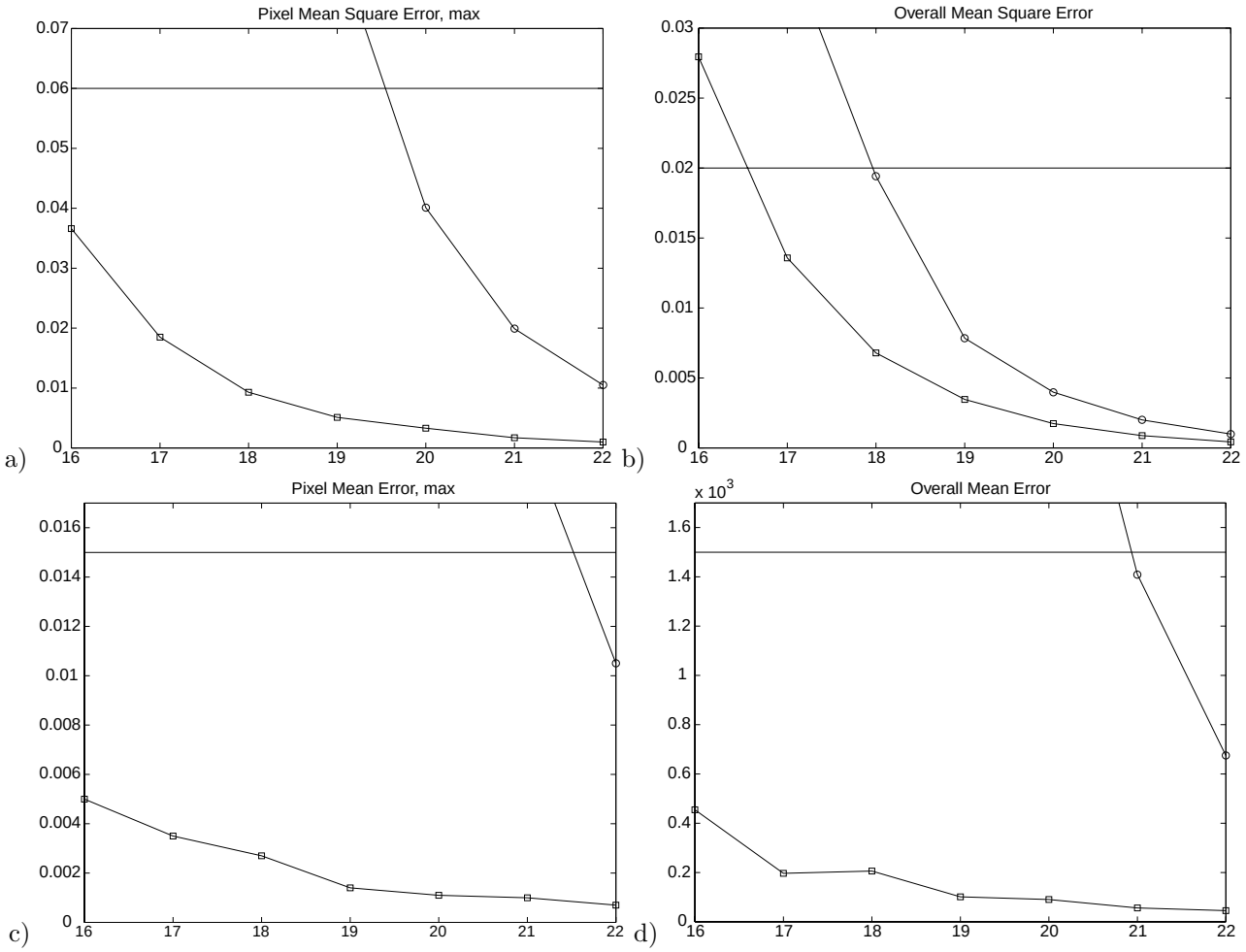

Figure 4: Error behaviour of the proposed architecture as a function of internal word width: a) pixel mean square error, b) overall mean square error, c) pixel mean error, and d) overall mean error. Line with squares: rounding, line with circles: two's complement, and the solid line: requirement of the IEEE Standard.

elements during the computation.

3 Error Analysis

In order to avoid overflow during the computations with fixed-point, i.e., fractional, number representation, the intermediate signal levels are scaled. Due to the fact, that all the intermediate data vectors are passed through multipliers, the signal levels can be adjusted at multipliers. This allows scaling factors to be selected with finer resolution without additional hardware costs. Typically scaling without additional hardware costs is done by rewiring, i.e., scaling factors are powers of two.

In fixed-point realizations, the main error source is the quantization error caused by the finite wordwidth in the intermediate arithmetic. In order to analyze the accyracy requirements of the architecture, the IEEE Standard 1180-1990 [1] is used as a test suite for the architecture. For this purpose, the proposed architecture should be extended to 2-D. This can be accomplished by utilizing the separability of the IDCT, i.e., the row-column approach, and realizing the 2-D transform with consecutive 1-D transforms with an additional 8×8 transpose unit.

The error behaviour of the pipeline architecture based on the IDCT algorithm shown in Fig. 2 is analyzed with simulations. The randam data generator described in the standard was used to create six test data sets. Then the performance of the architecture realized with different word widths and quantization methods was obtained with the aid of simulations. The used quantization methods where rounding to the nearest integer and truncation of two's complement ("rounding towards minus infinity"), which is the cheapest quantization method. In the simulations, the constant coefficients were rounded to the same word width as the internal data word width.

The obtained error values, mean error and mean square error per pixel and overall mean error and mean square error, are presented in Fig. 4 as a function of the word width of the internal arithmetic. The results show

that 17-bit representation is needed to fulfil the specifications when rounding is used (one sign bit and 16 bits for fraction).

When using the truncation of two's complement as the quantization method, the internal word width should be 22 bits. In this case, better performance could be expected when considering only the mean square error values. However, due to the fact that truncation of two's complement introdusis bias to the quantized value, the sign of error after the final rounding and clipping defined by the standard is in general the same. Therefore, the mean error value is almost the same as the mean square error. Lower word width can be expected if some variance to the error can be introduced.

4 CONCLUSION

In this paper, a pipelined architecture for inverse discrete cosine transform has been described, which is based on in-place IDCT algorithm. The actual architecture is obtained by collapsing the signal flow graph of the algorithm with vertical projection. In the resulting architecture, multipliers are removed from the critical path thus pipelined multipliers can be used to increase the throughput of the architecture. With the aid of simulations, it is estimated that internal accuracy of 17 bits in a fixed-point realization is needed to satisfy the requirements of the IEEE Standard 1180-1990 for 8×8 IDCT when using rounding as the quantization method. When quantization is performed with truncation of two's complement, 22-bit word width is required.

References

[1] IEEE Std 1180-1990, "IEEE standards specification for the implementations of 8×8 inverse discrete cosine transform," International standard, Institute of Electrical and Electronics Engineers, New York, U.S.A., Dec. 1990.

[2] S. Uramoto, Y. Inoue, A. Takabatake, J. Takeda, Y. Tamashita, H. Terane, and M. Yoshimoto, "A 100-MHz 2-D discrete cosine transform core processor," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 4, pp. 492–499, Apr. 1992.

[3] M. Matsui, Y. Uetani, L. Kim, T. Nagamatsu, Y. Watanabe, A. Chiba, and K. Matsuda, "A 200 MHz 13 mm^2 2-D DCT macrocell using sense-amplifying pipeline flip-flop scheme," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 12, pp. 1482–1490, Dec. 1994.

[4] W. H. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. on Communications*, vol. 25, no. 9, pp. 1004–1009, Sept. 1977.

[5] A. Madisetti and A. N. Wilson, "A 100 MHz 2-D 8×8 DCT/IDCT processor for HDTV applications," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 5, no. 2, pp. 158–165, Apr. 1995.

[6] H. L. Groginsky and G. A. Works, "A pipeline fast Fourier transform," *IEEE Trans. on Computers*, vol. 19, no. 11, pp. 1015–1019, Nov. 1970.

[7] M. Kovac and N. Ranganathan, "JAGUAR: A fully pipelined VLSI architecture for JPEG image compression standard," *Proceedings of the IEEE*, vol. 83, no. 2, pp. 247–257, Feb. 1995.

[8] Y. Arai, T. Agui, and M. Nakajima, "A fast DCT-SQ scheme for images," *Trans. IEICE E.*, vol. 71, no. 11, pp. 1095–1097, Nov. 1988.

[9] J. Astola and D. Akopian, "Architecture-oriented regular algorithms for discrete sine and cosine transforms," *IEEE Trans. on Signal Processing*, vol. 47, no. 4, pp. 1109–1124, Apr. 1999.

[10] J. Takala, D. Akopian, J. Astola, and J. Saarinen, "Constant geometry algorithm for discrete cosine transform," to appear in *IEEE Trans. Signal Process.*, June 2000.

[11] E. Bidet, D. Castelain, C. Joanblanq, and P. Senn, "A fast single-chip implementation of 8192 complex point FFT," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 3, pp. 300–305, Mar. 1995.