ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 2011. No. 7(113)
ELEKTRONIKA IR ELEKTROTECHNIKA

ELECTRONICS
T 170
ELEKTRONIKA

A 2-D DCT Hardware Codec based on Loeffler Algorithm

I. Martisius, D. Birvinskas, V. Jusas, Z. Tamosevicius
Software Engineering Department, Kaunas University of Technology,
Studentų str. 50-404, Kaunas, Lithuania, phone: 8 652 54447, e-mail: vacius.jusas@ktu.lt

crossref http://dx.doi.org/10.5755/j01.eee.113.7.611

Introduction

Digital picture and video applications are becoming a inseparable part of our everyday lives. Still image and video devices such as cell phones and cameras rely on codecs (coders and decoders) for image coding, processing and compression. These codecs must meet the ever-growing demands for power dissipation and operation speed. Currently the most popular standards of compression are JPEG for image, MPEG-1, 2, 4 and H263 for video [1]. All of these standards use the discrete cosine transform (DCT) and the inverse discrete cosine transform (IDCT) as a basis of their operation. DCT and the IDCT are some of the most computationally intensive blocks, therefore creating performance bottlenecks in picture and video data compression algorithms [2]. The computational complexity of the DCT in the coder surpasses than of any other unit, consuming 21% of the total computations [3, 4]. The IDCT also incurs the largest computational cost in the decoder. The complexity of these systems leads to large power dissipation and a fast, lossless yet energy efficient hardware codec is required. Implementing DCT/IDCT, as an ASIC is a design solution, which meets the real time processing requirements [2].

Since it's introduction, numerous fast algorithms have been proposed to calculate the DCT and IDCT. This paper focuses on the fast Loeffler algorithm and it's hardware applications.

Discrete cosine transform

The DCT transforms data from the spatial domain to the spatial frequency domain. It decorrelates the image data, which is typically highly correlated for small areas of the image. Heavy correlation provides much redundant information, whereas just a few pieces of uncorrelated information can represent the same data much more efficiently. Through the DCT transform, the energy of an image sample is packed into just a few spatial frequency coefficients. The DCT typically operates on 8x8 pixel blocks of image data. For an NxN block of samples *x(m,n)*, the two dimensional (2-D) DCT and IDCT are formulated:

$$Y(k,l)=\frac{2}{N}\alpha(k)\alpha(l)\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x(m,n)\cos\left(\frac{(2m+1)\pi k}{2N}\right)\cos\left(\frac{(2n+1)\pi l}{2N}\right), \quad (1)$$

$$x(m,n)=\frac{2}{N}\alpha(k)\alpha(l)\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}Y(k,l)\cos\left(\frac{(2m+1)\pi k}{2N}\right)\cos\left(\frac{(2n+1)\pi l}{2N}\right), \quad (2)$$

where $\alpha(0=\frac{1}{\sqrt{2}}$, α(j) = 1 for j ≠ 0. The straightforward implementations of DCT and IDCT require N^4 multiplications for an NxN block, so they are rarely implemented in hardware [5].

Beside that, computation of DCT/IDCT need floating-point multiplications, which is slow both in hardware and in software implementations. To achieve better operating speeds, coefficients can be scaled and approximated by integers so that floating-point multiplications can be replaced by integer multiplications.

DCT and IDCT are highly computational intensive, which creates prerequisites for performance bottlenecks in systems utilizing them. To overcome this problem, a number of algorithms have been proposed for more efficient computations of these transforms.

The measure of algorithm effectiveness is the number of mathematical operations needed to perform it. The number of multiplications is extremely important, since it is a relatively complex and power consuming operation, to be performed in hardware. The number of computations can be reduced from N^4 to $2Nlog_2N$ using fast DCT algorithms. Many one-dimensional algorithms were published over the years for N data points where $N = 2^m$, m ≥ 2 [6–8]. These algorithms are based on the sparse factorizations of the DCT matrix and many of them are recursive. The theoretical lower bound on the number of multiplications required for 1-D 8-point DCT has proven to be 11. In this sense, the method proposed by Loeffler, with 11 multiplications and 29 additions, is the most efficient solution. That's why it was chosen for this implementation.

Loeffler algorithm is combined out of stages. The stages of the algorithm numbered 1 to 4 are parts that have to be executed in serial mode due to the data dependency. In stage 2, the algorithm splits into two parts. One for the even coefficients, the other for the odd ones. The even part

is a 4 point DCT, again separated in even and odd parts in stage 3. The rounding block in Fig. 1 signifies a multiplication by a constant.

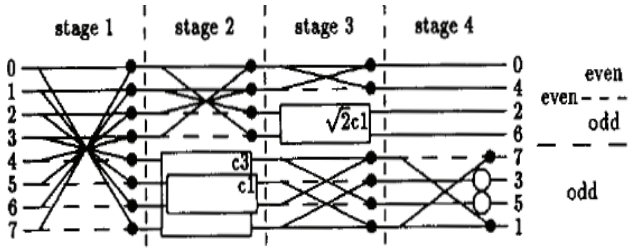


Fig. 1. 8-point Loeffler DCT algorithm [9]

Each stage must be executed in series and can not be evaluates in parallel because of data dependencies. However, calculations inside one stage can be parallelized. We clarify the building blocks of the algorithm in Fig. 2.

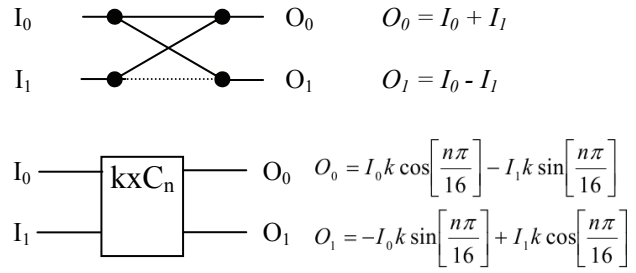


Fig. 2. The "butterfly" and the rotation block with their associated equations [9]

The rotation block can be calculated using 3 multiplications and 3 additions instead of 4 multiplications and 2 additions.

The two dimensional DCT transform equation (1) can be expressed as

$$Y(k,l) = \frac{2}{N}\alpha(k)\alpha(l)\sum_{n=0}^{N-1}\cos\left(\frac{(2m+1)\pi k}{2N}\right)\sum_{m=0}^{N-1}x(m,n)\cos\left(\frac{(2n+1)\pi l}{2N}\right). \quad (3)$$

This property, known as separability, has the advantage that 2D DCT can be computed in two steps by successive 1-D operations on rows and columns. The idea of a so-called row-column approach and is graphically illustrated in Fig. 3. It is the most popular approach for computing the 2D DCT/IDCT. It result in simple regular implementations that are less computationally efficient than direct 2D implementations.

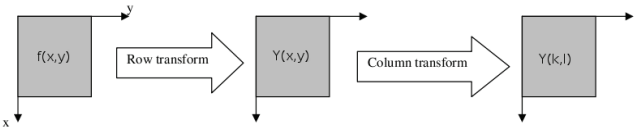


Fig. 3. Row-column approach for 2D-DCT/IDCT

Algorithm modelling

As we can see from the graph in Fig. 1, only 7 constant values need to be stored in the memory for Loeffler algorithm to function.

The algorithm signal-flow graph (Fig. 1) can be directly translated into a hardware implementation. However, it still uses floating-point multiplication which is too complex for hardware. The workaround for this problem is constant scaling. All constant can be multiplied by a factor of 2, so the floating point is eliminated, and the results are divided accordingly. In hardware applications, this kind of multiplication and division is easily done by shifting bits left or right. The choice of fixed-point constants is arbitrary. Using more bits gives better accuracy, because a more exact value of the constant is retained but with increased risk of overflow and surplus hardware register logic for storing and processing data.

To measure the quality of coding, the peak signal-to-noise ratio (PSNR) was calculated. It is most easily defined via the mean squared error (MSE) which for two m×n monochrome images I and K, where one of the images is considered a noisy approximation of the other, is defined as

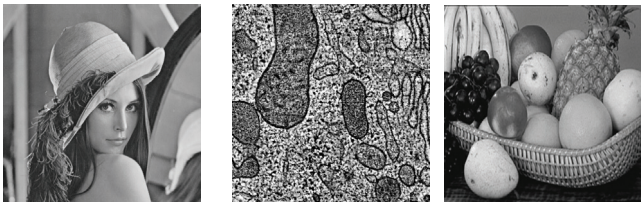
$$MSE = \frac{1}{64}\sum_{i=0}^{7}\sum_{j=0}^{7}[I(i,j) - K(i,j)]^2 . \quad (5)$$

The PSNR is defined as

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right), \quad (6)$$

where MAX_I is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per sample, this is 255 [10]. Typical values for the PSNR in lossy image and video compression are between 30 and 50 dB.

Mathematical modelling showed that the best choice for quality vs. size is a constant length of 10 bits. The codec using 10 bit constants is lossy because of fixed-point arithmetic. We show the different images used to test the algorithm Fig. 4.



a) b) c)

Fig. 4. Testing images: a – lena.bmp; b – cell.bmp; c –fruit.bmp

The PSNR measured for each of them are shown in Table 1.

Table 1. Mathematical modelling results

Picture	PSNR
Lena.bmp	33.6577
Cell.bmp	33.2790
Fruit.bmp	33.7290

Using a constant length of less than 10 bits leads to a lossy algorithm, which adds gross errors in the encoding of the image, which, when decoded, show up as visible relics in the picture. Although reducing constant bit size has a downside of reducing image quality, it also greatly reduces the area and power consumption of the finished schematic, there for it might still be used in applications, where image

quality is a secondary requirement, and power dissipation is a key, like portable video systems or cellular phones.

Hardware implementation

Hardware model of 2D-DCT is implemented using VHDL language. The system is divided into the following blocks:

- Computation unit;
- Data buffer;
- Counter;
- General control unit.

These blocks are shown in Fig. 5.

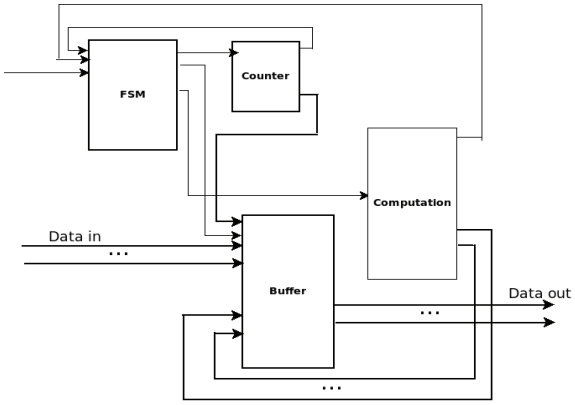


Fig. 5. Top level schematic

The computation unit performs one dimensional DCT or IDCT operation. This block has a generic interface, suitable for both DCT and IDCT algorithms. Interface has 8 data input buses and 8 output data buses. Beside data buses, the block uses *start* and *done* signals, for control and data flow simplification. This design allows reuse of all the system components for other DCT and IDCT computation algorithms.

For encoding, the system computes a DCT for a one-dimensional 8-point input sequence using Loeffler algorithm. This algorithm is directly translated from Fig. 1 and adapted to use fixed-point arithmetic with 10 bits constant length.

For decoding data DCT block is replaced with IDCT. Inverse transform may be computed by executing the stages backwards. A codec using any other 8-point 1D-DCT/IDCT computation algorithm can be implemented via reusing the components of this system.

Data buffer is the most complex block of the system. It has 4 functions:

1. Input and storing an 8x8 matrix;
2. Output lines of matrix to DCT processor and replace data lines with the result;
3. Output columns of the matrix to the DCT processor and replace data line with the result;
4. Output of the data matrix.

When 1D-DCT processor computes a result of any given 8-point input sequence the input data is no longer needed and can be overwritten with the computation result. This reduces memory size, resulting in smaller area and power consumption, however this method increases interconnect network.

General control unit is implemented as a finite state machine. It is used to control data flow between components.

Implementation results

We simulated VHDL register transfer level (RTL) code using Cadence simulation and modelling tools. Synthesis was performed using Synopsys tools. Since file formats for binary files, read by VHDL are not standardized, we used a pre/post-processor to convert binary images files to text format and vice versa.

Images used for testing where the same as in mathematical modelling. Results of synthesized gate level circuit simulation are shown in Table 2

Table 2. Gate level simulation results

Picture	PSNR
Lena.bmp	31.5951
Cell.bmp	30.8324
Fruit.bmp	31.8590

We show differences in PSNR for mathematical modelling and gate level circuit modelling of each image in Fig. 6.

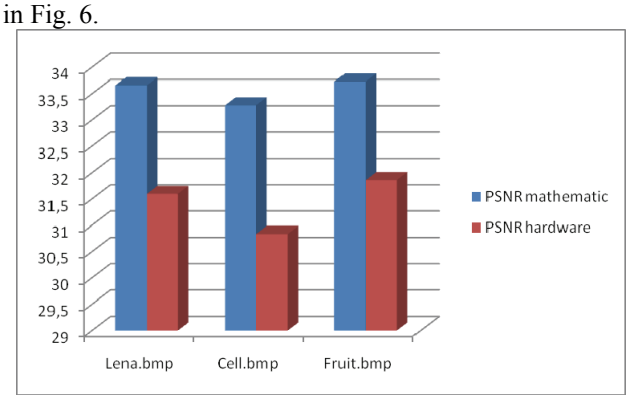


Fig. 6. Difference in PSNR

Implementation is also proven to work on FPGA matrices. A Xilinx Spartan-3AN was used to synthesize and test the RTL code. Utilization summary for the device is shown in Table 3.

Table 3. FPGA device utilization

Logic Utilization	Used	Available	Utilization
Slice Flip Flops	426	22,528	1%
4 input LUTs	2,280	22,528	10%
Occupied Slices	1,309	11,264	11%
Bonded IOBs	274	375	73%
MULT18X18SIOs	22	32	68%

Conclusions

In this paper, we reported a hardware implementation scheme and results of a Loeffler algorithm based 2-D discrete cosine transform codec. This codec performs a full 2-D DCT encoding and 2-D IDCT decoding. It is fully

functional, capable of being used in any hardware coding algorithm. The Loeffler algorithm is one of the newest and most effective fast DCT algorithms for hardware implementation. 2-D DCT/IDCT was computed by applying 1-D operations on matrix using a row/column approach. Mathematical and gate-level modelling of the algorithm has shown that fixed-point arithmetic and constant approximations add errors to the results of the algorithm.

For a hardware implementation, additional data buffers and control logic is used to perform 2-D transforms. Since hardware implementation is less accurate than mathematic modelling, more errors occur in the computational results. However, differences are small and cannot be directly observed. This codec should provide enough quality for low resolution image compression or processing algorithms at high speed.

References

1. **Atitallah A. B., Kadionik P., Ghozzi F., Nouel P., Masmoudi N., Marchegay P.** Optimization and implementation on FPGA of the DCT/IDCT algorithm // IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '06), 2006. – P. 928–931.
2. **Bukhari K. Z., Kuzmanov G. K., Vassiliadis S.** DCT and IDCT implementations on different FPGA technologies // Proceedings of the 13th Annual Workshop on Circuits, Systems and Signal Processing (ProRISC'02). – Veldhoven, The Netherlands, 2002. – P. 232–235.
3. **Bareiša E., Jusas V., Motiejūnas K., Šeinauskas R.** On the Enrichment of Functional Delay Fault Tests // Information Technology and Control, 2009. – No. 38(3). – P. 208–216.
4. **Bareiša E., Bieliauskas P., Jusas V., Targamadzė A., Motiejūnas L., Šeinauskas R.** Factor of Randomness in Functional Delay Test Generation for Full Scan Circuits // Electronics and Electrical Engineering. – Kaunas: Technologija, 2010. – No. 9(105). – P. 39–42.
5. **Narasimha M. J., Peterson A. M.** On the computation of the discrete cosine transform // IEEE Trans. Commun., 1978. – No. 26(6). – P. 934–936.
6. **Chen W.–H., Smith C. H., Fralick S.** A Fast Computational Algorithm for the Discrete Cosine Transform // IEEE Trans. on Communications, 1977. – P. 1004–1009.
7. **Bareiša E., Jusas V., Motiejūnas K., Šeinauskas R.** On the Enrichment of Static Functional Test // Electronics and Electrical Engineering. – Kaunas: Technologija, 2009. – No. 3(91). – P. 9–14.
8. **Bareiša E., Jusas V., Motiejūnas K., Šeinauskas R.** The non–scan delay test enrichment based on random generated long test sequences // Information Technology and Control, 2010. – No. 39(4). – P. 251–256.
9. **Loeffler C., Lightenberg A., Moschytz G. S.** Practical fast 1–D DCT algorithms with 11 Multiplications // Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP'89), 1989. – P. 988–991.
10. **August N. J., Dong Sam Ha.** Low Power Design of DCT and IDCT for Low Bit Rate Video Codecs // IEEE transactions on multimedia, 2004. – Vol. 6. – No. 3. – P. 414–422.

Received 2011 03 09

I. Martisius, D. Birvinskas, V. Jusas, Z. Tamosevicius. A 2-D DCT Hardware Codec based on Loeffler Algorithm // Electronics and Electrical Engineering. – Kaunas: Technologija, 2011. – No. 7(113). – P. 47–50.

In this paper, we report a hardware implementation scheme and results of a Loeffler algorithm based 2-D discrete cosine transform codec. This codec performs a full 2-D DCT encoding and 2-D IDCT decoding. It is fully functional, capable of being used in any hardware coding algorithm. The Loeffler algorithm is one of the newest and most effective fast DCT algorithms for hardware implementation. 2-D DCT/IDCT was computed by applying 1-D operations on matrix using a row/column approach. For a hardware implementation, additional data buffers and control logic is used to perform 2-D transforms. Since hardware implementation is less accurate than mathematical modelling, more errors occur in the computational results. However, differences are small and cannot be directly observed. Ill. 6, bibl. 10, tabl. 3 (in English; abstracts in English and Lithuanian).

I. Martišius, D. Birvinskas, V. Jusas, Ž. Tamoševičius. 2-D DCT kodeko realizacija, naudojant Loefflerio algoritmą // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2011. – Nr. 7(113). – P. 47–50.

Aprašoma aparatinio dvimatės kosinusinės transformacijos (DCT) kodeko realizacija. Šis kodekas atlieka visišką 2-D DCT kodavimą ir dekodavimą (2-D IDCT). Jis gali būti pritaikomas visuose aparatiniuose kodavimo algoritmuose. Loefflerio algoritmas yra vienas efektyviausių ir naujausių greitųjų DCT algoritmų. Dvimatė DCT/IDCT skaičiuojama pritaikant vienmatės DCT algoritmą eilučių ir stulpelių metodu. Atliekant dvimates DCT operacijas, panaudojama papildoma atmintis ir valdymo logika. Aparatinės realizacijos skaičiavimo tikslumas mažesnis nei matematinio modeliavimo, tačiau šie skirtumai nedideli ir plika akimi nematomi. Il. 6, bibl. 10, lent. 3 (anglų kalba; santraukos anglų ir lietuvių k.).