VLSI DESIGN
2000, Vol. 11, No. 4, pp. 397–403
Reprints available directly from the publisher
Photocopying permitted by license only

Low Power VLSI Implementation of the DCT on Single Multiplier DSP Processors*

S. MASUPE and T. ARSLAN†

Electronics and Electrical Eng. Department, University of Edinburgh, Kings Buildings Edinburgh, EH9 3JL

(Received 5 June 1999; In final form 2 September 1999)

A generic multiplication scheme for the low power VLSI implementation of the DCT is described in this paper. The scheme concurrently processes blocks of cosine coefficient and pixel values during the multiplication procedure, with the aim of reducing the total switched capacitance within the multiplier circuit. The cosine coefficients, within each block, are manipulated such that some are processed using shift operations only. The remaining coefficients are presented to the multiplier inputs as a sequence, ordered according to bit correlation between successive cosine coefficients. The paper describes the multiplication scheme, the power evaluation environment used, and presents results, with a number of standard benchmark examples, demonstrating upto 50% power saving.

Keywords: Low power; DCT; Image compression; VLSI

1. INTRODUCTION

Currently there is considerable interest in the low power implementation of the Discrete Cosine Transform (DCT). This is mainly due to the DCT being the computational bottleneck of standards such as JPEG and MPEG [1]. Most research work considering low power implementation of the DCT have targeted reducing the computational complexity of the design or modifying it for operation under a lower supply voltage [1, 2]. Both these techniques have a limited effect on power reduction. Another major contribution to power consumption is due to the effective switched capacitance [3, 4]. Only a few researchers have targeted reducing power of a DCT implementation through a reduction in the amount of switched capacitance. This reduction has been achieved through techniques such as the detection of zero-valued DCT coefficients and lookup table partitioning [5].

This paper presents a generic multiplication scheme for the low power VLSI implementation of the DCT on CMOS-Based Digital Signal

* Based on "Low Power DCT implementation approach for VLSI DSP Processors" by S. Masupe and T. Arslan which appeared in *Proceedings of International Symposium on Circuits and Systems*; Florida 30 May–2 June, 1999, Pages I–149 to I–152. © 1999 IEEE.

† Corresponding author. e-mail: Arslan@ee.ed.ac.uk

Processors. A basic form of the scheme, which operated on primary images with bi-level pixel value patterns, was first proposed in [6]. The scheme, outlined in this paper, can be used with a wide varity of images including those with continuous and sharply varying pixel values. The scheme concurrently processes blocks of cosine coefficient and pixel values during the multiplication procedure, with the aim of reducing the total switched capacitance within the multiplier circuit. The cosine coefficients, within each block, are processed such that some are processed using shift operations only. The remaining coefficients are presented to the multiplier inputs as a sequence, ordered according to bit correlation between successive cosine coefficients. The paper describes the multiplication scheme, the power evaluation environment used, and presents results, with a number of standard benchmark examples, demonstrating upto 50% power savings.

2. IMPLEMENTATION

The computational bottleneck for the implementation of the DCT is the multiplication of the cosine coefficients matrix [**E**], by the pixel matrix [**D**]. in order to obtain the DCT coefficients [**C**], *i.e.*,

$$[\mathbf{C}] = [\mathbf{E}] * [\mathbf{D}] \tag{1}$$

where each element C_{ik} in $[C]$ matrix of order n is given by:

$$C_{ik} = \sum_{x=1}^{n} E_{ix} D_{xk} \tag{2}$$

Traditionally, the multiplication process is performed in a row-by-column fashion [7, 8]. See Eq. (2). In some applications where the number of multipliers is limited, this implies new data at both multiplier inputs everytime a multiplication is conducted. This in turn implies a large switching activity inside the multiplier circuit.

A number of experiments were carried out with some cosine matrix and various pixel matrices. Careful analysis of the multiplication procedure revealed two distinct categories of cosine elements: (1) elements that are powers of 2, and (2) those which are non-powers of 2 numerics. For this reason, the algorithm processes the elements in (1) by performing a simple shift operation. It is well known that the switched capacitance of a shift is significantly less than that of a multiplication [2].

Examination of the multiplication procedure embodied in Eq. (2) can reveal more scope for power optimisation. This can be demonstrated by expanding Eq. (2) with $n = 8$, as shown in Figure 1. As could be deduced from the figure, when computation is performed on the basis of evaluating coefficients successively, *i.e.*, $C_{11} - C_{81}$, then new pixel and coefficient values are processed at both inputs of the multiplier. If the multiplication procedure is performed on a column-after-column basis (*i.e.*, $E_{i1} * D_{11}$, $E_{i2} * D_{12}$, and so on) then the value of D_{xk} will be constant (for 8 multiplications) at the corresponding multiplier input. This implies less switching activity at the multiplier inputs and memory buses. For each column being

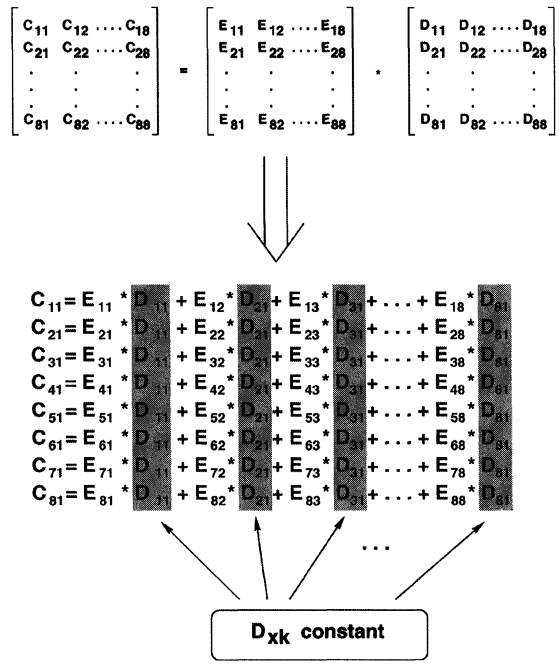

FIGURE 1 Matrix multiplication process.

processed, the multiplication procedure outlined above has the product form ($m *$ constant), where m are the cosine matrix elements in the column being processed. The *constant* remains valid for only these values of m, if the column changes, a new *constant* is activated. The multiplication results are unchanged irrespective of the order in which the *constant* is multiplied by the sequence of cosine coefficients in the column. For this reason, the algorithm performs the multiplication of elements in (2) in a column-by-column fashion and orders the elements, in the column being processed, according to some criteria. In our case the criteria being minimum switching activity (hamming distance) between consecutive coefficients multiplied by the *constant*. This guarantees that similar elements are subsequently applied to the inputs of the multiplier causing minimum switching activity within the internal circuit of the multiplier. Although, our investigations were carried out using the example of ordering elements according to minimum hamming distance, in practice, any ordering algorithm can be used and *the amount of power saving is determined by the power of the algorithm used.* The steps in Figure 2 outline the algorithm, which commences with the following initialisation steps: (1) Process entries E_{1x} with shift operation. (2) Order remaining coefficients according to some ordering algorithm. (3) Save entries E_{ix} in (E) memory, see Figure 3, with elements of the same column being adjacent to each other, followed by the next column, and so on.

To illustrate the above algorithm, consider an example where;

$$\mathbf{E} = \begin{pmatrix} 2.00 & 2.00 & 2.00 & 2.00 \\ 1.85 & 0.77 & -0.77 & -1.85 \\ 1.41 & -1.41 & -1.41 & 1.41 \\ 0.77 & -1.85 & 1.85 & -0.77 \end{pmatrix}$$

and

$$\mathbf{D} = \begin{pmatrix} 35 & 32 & 31 & 34 \\ 32 & 37 & 31 & 31 \\ 29 & 28 & 27 & 31 \\ 26 & 28 & 31 & 34 \end{pmatrix}$$

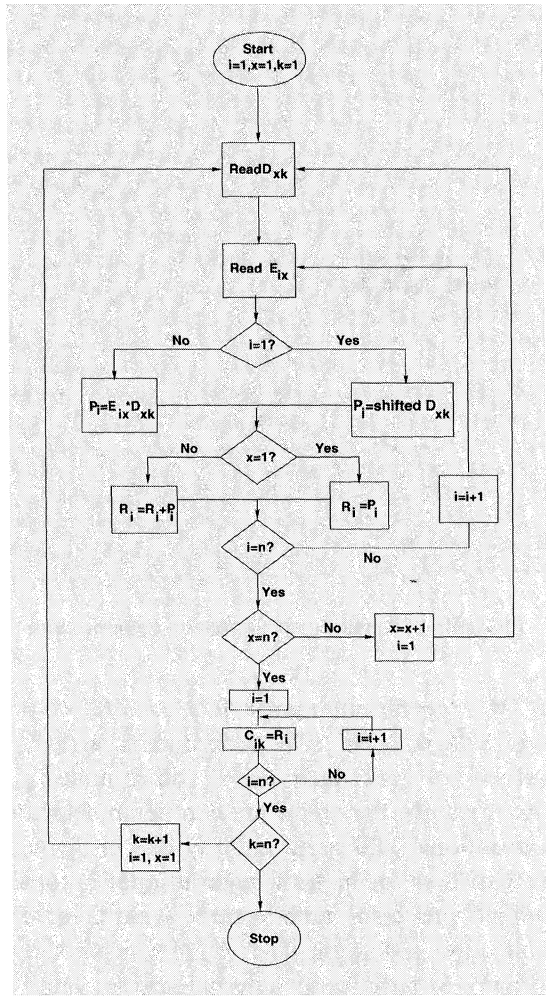

FIGURE 2 Flowchart of the algorithm.

After the first iteration ($i=1$, $x=1$ and $k=1$), the first column, E_{i1}, is ordered according to minimum hamming distance to produce the ordered sequence (2.00, 0.77, 1.41, 1.85). The original locations of these elements are stored. Next the above sequence is multiplied by the first entry in D_{x1}, 35. The entries in registers R_i will be (70, 64.75, 49.35, 26.95), where the first entry (70) is saved in R_1 and the second entry is saved in R_2 and so on. Similarly at the end of iteration 2 ($i=2$, $x=1$ and $k=1$), R_i will contain (134, 89.39, 4.23, -32.25). The last iteration ($i=4$, $x=1$ and $k=1$) for this particular column, results in R_i containing the first column of

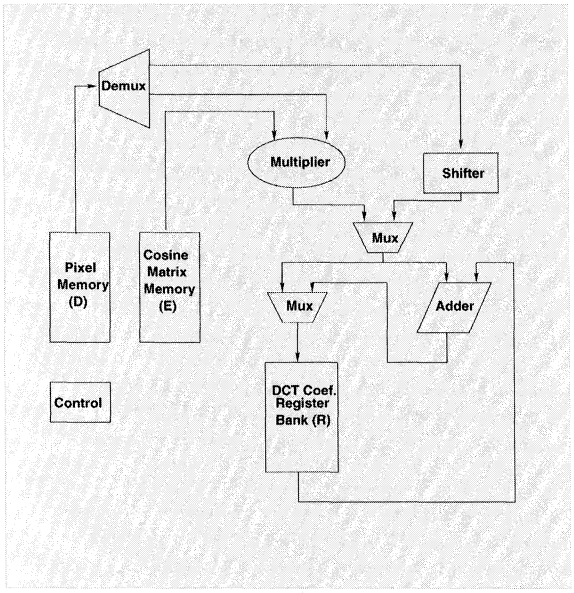

FIGURE 3 Simplified architecture of the processor.

the DCT coefficient matrix $[C]$, *i.e.*, $R_i = C_{x1} = (244, 18.96, 0, 1.35)^T$. This procedure is carried out until $x = k = n$, at which case R_i will contain C_{44}.

As it stands, the algorithm can be implemented on traditional DSPs without any loss in throughput. However for high throughput applications, a modified processor architecture is required. The architecture, a simplified version of which is shown in Figure 3, requires an internal register bank in order to store the partial products, (R), which eventually result in the DCT coefficients, $[C]$. In addition, a special memory unit is allocated for both the cosine and the pixel matrix elements, E_{ix} and D_{xk} respectively. A shifter is included to cope with the additional shift operations. The multiplier and the adder units are included to perform the normal multiply-add DSP operations. Since the outputs of both the multiplier and the shifter have to share the same input of the register bank, a multiplexer is needed to resolve which one output can use the register bank input. Another multiplexer is required to handle outputs from multiplier/shifter and adder units since some of the inputs to the register bank proceed directly to the bank without passing through the adder.

3. SIMULATION AND RESULTS

Figure 4 illustrates the framework utilised for the evaluation of the scheme with a number of 512×512 pixel example benchmark images. These, which include Lena, Man and checked images, are shown in Figure 5. The cosine coefficient matrix used was obtained using the MATLAB signal processing toolbox [9]. This was scaled such that entries can be represented by numbers between -128 and 127. The evaluation environment is based upon the MPEG standards, where images are processed in blocks of 8×8. For this reason our results are obtained with $n = 8$, *i.e.*, an 8×8 cosine matrix is used with 512×512 pixel images being processed in blocks of 8×8. An

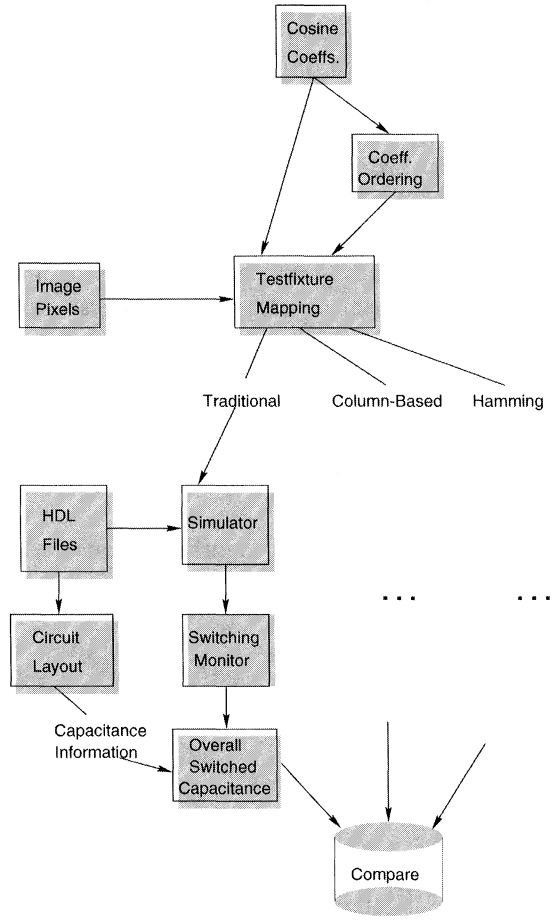

FIGURE 4 Framework for the algorithm evaluation.

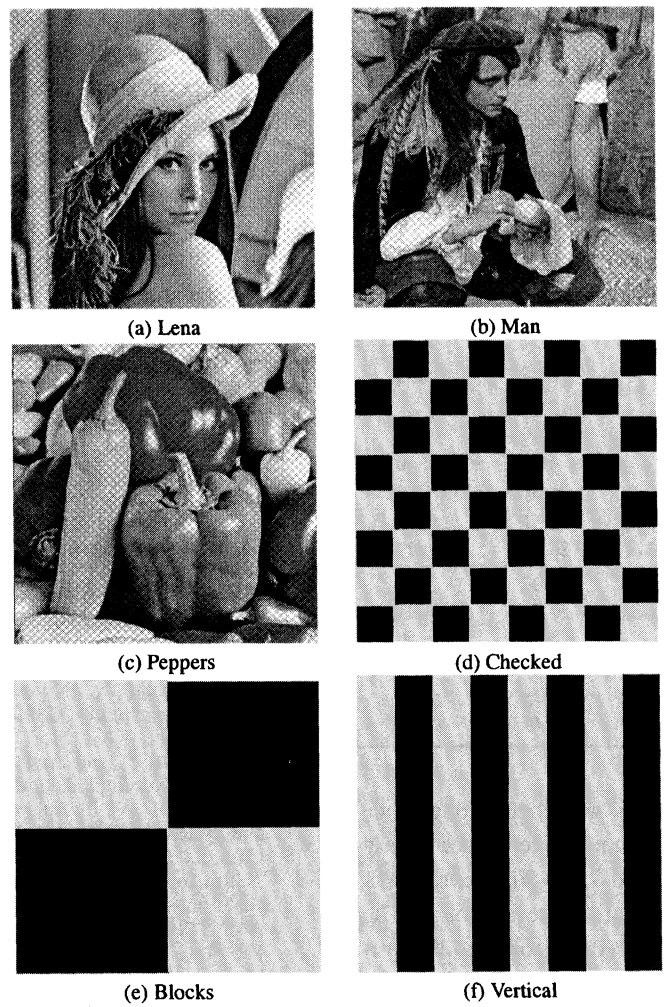

FIGURE 5 Some of the tested images.

8×8-bit array Multiplier was constructed, using the Cadence VLSI suite with ES2 0.7 CMOS Technology, and processed down to layout level. A *C*-program based *test-fixture* mapping system was developed to generate input simulation files for the Verilog-XL^{TM} digital simulator [10]. This involved forming the appropriate image-pixel/cosine-coefficient pairs, in the order imposed by the multiplication algorithm, so that they can be applied to the inputs of the hardware multiplier. Three types of input simulation files are generated, representing the use of one of the following: (1) Traditional cosine DCT multiplication (*Traditional*). (2) Column-based multiplication algorithm without ordering (*Column-based*). (3) Column-based multiplication algorithm with ordering according to minimum hamming distance (*Hamming*). In each simulation, the number of signal transitions (switching activity), at the output of each gate, is monitored. Capacitive information for each gate is extracted from the layout of the multiplier circuit. Both of these are used to obtain a figure for the total switched capacitance of the multiplier.

Table I illustrates the results obtained with the different bench mark images. Clearly, power saving is achieved in all cases with a maximum of 50% with the checked image using *Hamming*,

TABLE I Typical power savings

Image	Ordering	Switched capacitance (pF) $*10^3$	Switched capacitance reduction (%)
Horizontal Stripes	Traditional	13.77	–
	Column-Based	8.45	38.64
	Hamming	7.06	48.64
Vertical Stripes	Traditional	7.70	–
	Column-Based	7.55	1.97
	Hamming	6.41	16.84
Blocks	Traditional	9.25	–
	Column-Based	7.65	17.27
	Hamming	6.47	29.99
Checked	Traditional	13.81	–
	Column-Based	8.39	39.25
	Hamming	6.83	50.50
Lena	Traditional	39.01	–
	Column-Based	29.51	24.35
	Hamming	26.23	32.77
Man	Traditional	40.05	–
	Column-Based	30.08	24.91
	Hamming	26.63	33.50
Peppers	Traditional	39.57	–
	Column-Based	29.75	24.84
	Hamming	26.49	33.05

which reduces the amount of switched capacitance at both multiplier inputs. An average power saving of around 35% is achieved over all benchmark images. Our results illustrate that although the amount of power saving is proportional to the frequency of pixel value variation across the image, significant power saving is achieved with all image types. This include those with sharp variation in pixel value, such as the checked image, and those with continuous spectrum of pixel values, such as Lena, Man and Peppers images. This in turn implies the suitability of the scheme for wide ranges of images including coloured images.

4. CONCLUSIONS

The paper presents an effective scheme for the low power VLSI implementation of the DCT within an MPEG based environment. The scheme reduces power through the utilisation of shift operations, where possible, together with sequencing coefficient and pixel multiplications in an order dictated by the bit correlation of their successive values. This results in upto 50% power reduction with a number of standard benchmark images. Our investigations revealed that the scheme provides a generic framework for power saving in that significant power savings is achieved with a range of standard image types. These include those with sharp, continuous, and low pixel variation frequency. The scheme can easily be extended to the implementation of the 2D DCT.

References

[1] Pao, I. and Sun, M., "Computation reduction for discrete cosine transform", In: *International Symposium on Circuits and Systems*, **4**, 285–288, IEEE, 31 May–3 June, 1998.

[2] Chen, L., Jiu, J., Chang, H., Lee, Y. and Ku, C., "Low power 2D DCT chip design for wireless multimedia terminals", In: *International Symposium on Circuits and Systems*, **4**, 41–44, IEEE, 31 May–3 June, 1998.

[3] Chandrakasan, A. P. and Brodeserson, R. W. (1995). *Low Power Digital CMOS Design*. Kluwer Academic Publishers.

[4] Erdogan, A. T. and Arslan, T., "Data block processing for low power implementation on single multiplier CMOS DSP processors", In: International Symposium on Circuits and Systems, **5**, 441–444, IEEE, 31 May–3 June, 1998.

[5] Cho, S., Xanthopoulos, T. and Chandrakasan, A. P., "Ultra low power variable length decoder for MPEG-2 exploiting codeword distribution", In: *Proccedings of the 1998 IEEE Custom Integrated Circuits Conference*, **4**, 177–180, IEEE, May, 1998.

[6] Masupe, S. and Arslan, T., "Low power DCT implementation approach for VLSI DSP processors", In: *International Symposium on Circuits and Systems*, IEEE, 30 May–2 June, 1999. Accepted for publication.

[7] Chen, W. (1995). *The Circuits and Filters Handbook*. CRC Press Inc.

[8] Bhaskaran, V. and Konstantinedes, K. (1995). *Image and Video Compression Standards: Algorithms and Architectures*. Kluwer Academic Publishers.

[9] Little, J. and Shure, L., *Signal Processing toolbox for use with MATLAB*. The Mathworks Inc., July, 1992.

[10] Thomas, D. E. and Morby, P. R. (1995). *The Verilog Hardware Description Language*. Kluwer Academic Publishers.

Authors' Biographies

Shedden Masupe is a Ph.D. research student at the University of Edinburgh. His research interests include Low Power VLSI Design, DCT-Based Image Compression Techniques, HDL based

Design Methodologies and Microprocessor Architectures. He is a student member of the IEEE.

Dr. Tughrul Arslan is a Senior Lecturer in the University of Edinburgh, where he leads the System Level Integration activity. His research interests include Low Power VLSI Design and DSP, System-on-Chip applications, design automation, and test. Dr. Arslan is a member of the IEEE and the IEE.