

LOW POWER ORDER BASED DCT PROCESSING ALGORITHM

S. Masupe and T. Arslan

The University of Edinburgh
Department of Electronics and Electrical Engineering
Mayfield Rd., Edinburgh EH9 3JL, United Kingdom

ABSTRACT

This paper presents an algorithm and an associated architecture for the low power implementation of the Discrete Cosine Transform. The algorithm reduces power by manipulating bit-correlation between successive cosine coefficients applied to the input of the Multiply-Accumulate section such that the effective switched capacitance is reduced. This reduces the switching activity in the Discrete Cosine Transform processor. The paper describes the algorithm and the proposed architecture. The evaluation procedure is also presented including the results from a number of example images illustrating up-to 29% power savings.

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 minimising the bitwidth of arithmetic operations in the presence of data spatial correlation [5]. This paper presents a technique for reducing power dissipation of the DCT by targeting the multiplier section of a DCT processor. The pivot of this technique is a multiplication algorithm for the low power implementation of the DCT. The algorithm reduces power consumption by reducing the effective switched capacitance of the multiplier through effective manipulation of the multiplication process between the cosine and data matrices. Our results indicate that the effective capacitance can be reduced significantly by performing the multiplications of the DCT in an order dictated by the amount of cor-

relation between subsequent cosine matrix elements applied to one of the inputs of the multiplier section. In addition we propose a modified processor architecture that accommodates this multiplication scheme and is open to exploitation by other ordering algorithms. We demonstrate our scheme by an example of ordering the cosine coefficients according to minimum Hamming distance, revealing up to 29% power savings with a number of image examples.

The scheme utilises the cosine coefficients ordering as in [6], however, the ordering is done in the rows of the cosine matrix rather than the columns. The advantage of ordering the coefficients in rows can be easily identified by the fact that no modification is needed in the data path as in [6]. Therefore the overheads will be minimum since the same data path is used as the in the conventional implementation [7].

2. IMPLEMENTATION

The most computationally intensive process in the implementation of the DCT is the multiplication of the cosine coefficients matrix $[E]$, with the pixel matrix $[D]$, in order to obtain the DCT coefficients $[C]$, i.e.

$$[C] = [E] * [D] \quad (1)$$

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

$$C_{ij} = \sum_{k=1}^n E_{ik} D_{kj} \quad (2)$$

The cosine coefficients for any DCT implementation are constant, therefore ordering can be done prior to loading them into a ROM unit. When the ordering is undertaken, the original row location of the coefficient is tagged along with the corresponding cosine coefficient word. The proposed cosine coefficient representation is shown in Figure 1. This allows the saved location to be used as the address of the corresponding pixel element to be computed. The three MSB's of Figure 1 represent the coefficient row location in the cosine coefficient matrix, whereas the remaining bits represent the value of the cosine coefficient.

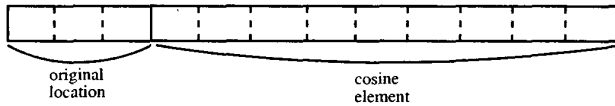


Figure 1: cosine coefficient with saved location

The flowchart for the implementation is depicted in Figure 2: The flowchart assumes that ordering of the cosine coefficients has already been done prior to commencement of the steps in the chart. Note that the flowchart is for processing one block of the image (8X8). For the processing of an entire image, the complete flowchart should be followed in order to process each 8X8 block. The steps for the scheme are listed below.

1. Decode the cosine word nE into its constituent parts, ie, pixel location (n) and cosine word(E)
2. Get the value pixel(D) from location specified by n in (1)
3. Multiply ; $E * D$ and add the result to the accumulator
4. Repeat steps (1) to (3) for the remaining row entries
5. Obtain the DCT coefficient (C) and clear the accumulator
6. repeat 1) to 5) until the last row of cosine matrix is processed
7. Restart processing of the next image block

To illustrate the above algorithm, consider an example where the cosine matrix is as shown in Figure 3(a). After sorting, according to minimum Hamming distance, the cosine matrix will be as in Figure 3(b).

As an example, consider the multiplication of nE (Figure 3(b)) with the following matrix.

$$D = \begin{bmatrix} 35 \\ 32 \\ 29 \\ 26 \end{bmatrix}$$

Evaluation of the first row in Figure 3(b), the multiplication procedure between this row and $[D]$ will follow the same steps as conventional matrix multiplication since the order has not been altered. The second row however has been ordered according to minimum Hamming distance. So the multiplication for the row will resume with $84*35$. This is because the first entry (84) in the second row has $n=0$ which implies that the first entry in $[D]$ should be processed. The next entry to be processed is 35 which has $n=2$, hence

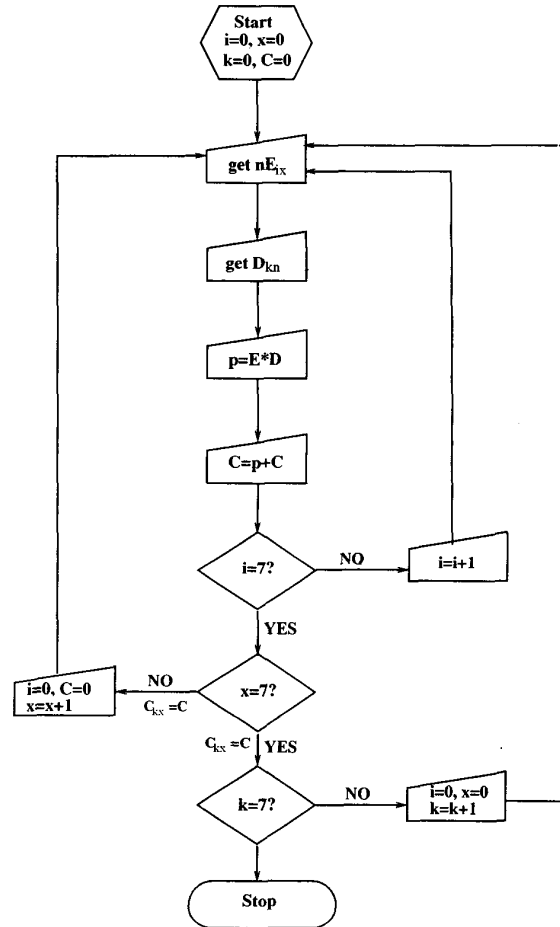


Figure 2: Flowchart of the algorithm

the multiplication $35*29$ will be carried out. The third computation is $-84*26$. Finally $35*32$ is processed. The multiplication procedure carries on to the next row of the cosine matrix and follows the same technique.

A simplified DCT processor for the proposed scheme is depicted in Figure 4. This differs with the conventional implementation by the fact that a slice of the data coming from *cosine coeff memory* is used as an address for the *pixel memory*. The depth of the *pixel memory* is 64, therefore a 6-bit address is required to access a memory of this depth. This problem is alleviated by using a 3-bit counter to generate the rest of the 6-bit address. The 3-bit address from the counter is appended as the most significant bits of the *pixel memory* address to make a 6 bit word. This counter is incremented after every 8 memory accesses. The overhead for this extra circuitry is very minimum since ordinarily a 6 bit counter would have been required to generate

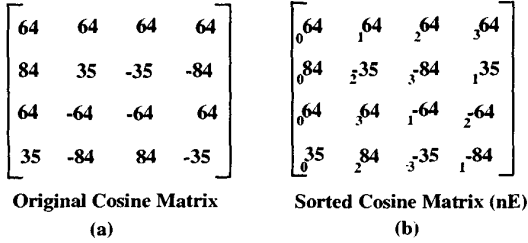


Figure 3: Example cosine matrix before and after ordering

the address for the *pixel memory*. Another issue that might arise from this design will be the larger data word size of the *cosine coeff memory* compared to the conventional implementation. This is covered by the fact that on a conventional implementation a 6-bit address bus will be required to connect between the *control* and the *pixel memory* (the dashed connection in Figure 4, labelled *p_addr*). So, as far as bus toggling is concerned, for the proposed implementation there are 3 bits that can toggle compared to the 6 bits that would be generated otherwise.

The same MAC unit architecture was used for both the conventional implementation and the cosine ordering scheme. The modules in the MAC unit are *mult*, *add* and *reg*. The module *mult*, was implemented using *Baugh-Wooley* multiplier. This architecture was chosen because of wide use in DSP research. For the *add* module, *Brent-Kung* adder implementation was used. After testing several adders, the *Brent-Kung* adder proved to be more power efficient and suitable for high speed applications.

3. SIMULATION AND RESULTS

Figure 5 illustrates the framework for the evaluation of the algorithm. The cosine coefficient matrix used was obtained using the MATLAB signal processing toolbox. The pixels were chosen to represent several types of scenarios including Lena, checked and striped images. The input images are pre-processed into blocks 8x8 matrices. The preprocessing is performed using a perl script. Three types of input simulation files are generated, representing the use of one of the following: (1) Traditional cosine DCT multiplication (*conventional*). For this simulation, the *ordering algorithm* block in Figure 5 is excluded. (2) Cosine ordering in ascending order (*ascending*). (3) Cosine ordering according to minimum Hamming distance (*Hamming*). Any ordering scheme that reduces bit toggling between coefficients can be used. The ordering algorithms used in this paper are for demonstration purposes only, hence the optimum solution for Hamming distance may have not been reached. For the generation of the netlist, Synopsys's *Design Compiler* was used and the design was mapped onto the Alcatel 0.35µm

CMOS library.

Table 1 illustrates the results obtained with the different scenarios used to test the algorithm. As the table shows, power savings are achieved with all examples, with a maximum of 29% power saving for the MAC unit. The horizontal striped image provides the most power saving. This is the result of the inherently low toggling of the pixels within the image. For this image, the conventional MAC consumed a total dynamic power (*tdp*) of 6.57mW. This is listed under *conventional* for the horizontal stripes image. For the horizontal stripes, the *tdp* consumption for *Hamming* is 4.75mW. This reflects a power saving of 22.25% compared to *conventional* approach. And finally *ascending* produces a power saving of 29%. The cell internal power (*cip*) of the MAC components is also listed in Table 1.

Image	Scheme	Power	mult	add	reg	Power Saving (%)
		(mW) tdp	(cip)	(cip)	(cip)	
Lena	conventional	9.84	2.33	1.75	1.17	-
	Hamming	8.16	1.84	1.49	1.08	17.07
	ascending	8.08	1.83	1.45	1.07	17.89
Wallace	conventional	9.94	2.41	1.75	1.18	-
	Hamming	8.22	1.91	1.46	1.08	17.30
	ascending	8.27	1.94	1.45	1.09	16.80
checked	conventional	10.67	2.77	1.66	1.05	-
	Hamming	8.76	2.21	1.38	0.99	17.90
	ascending	8.47	2.13	1.31	0.97	20.62
vertical stripes	conventional	10.43	2.73	1.59	1.03	-
	Hamming	8.81	2.27	1.35	0.98	15.53
	ascending	8.45	2.16	1.28	0.96	19.01
horizontal stripes	conventional	6.57	1.56	1.06	0.92	-
	Hamming	4.75	0.98	0.79	0.85	22.25
	ascending	4.63	0.963	0.75	0.83	29.53

Table 1: Typical power savings

4. CONCLUSIONS

An algorithm for the low power implementation of the Discrete Cosine Transform has been presented. The algorithm reduces power by manipulating data at multiplier inputs such that switching activity is reduced. This in turn has the effect of reducing the total switched capacitance. It has been shown that the algorithm can provide up to 29% power saving, when used in conjunction with an ordering algorithm which manipulates bit correlation of data values at multiplier inputs, and that it provides the potential for more savings in power. Since the sorting algorithms used are for demonstrating

5. REFERENCES

- [1] I.-M. Pao and M.-T. Sun, "Computation reduction for discrete cosine transform," in *International Symposium*

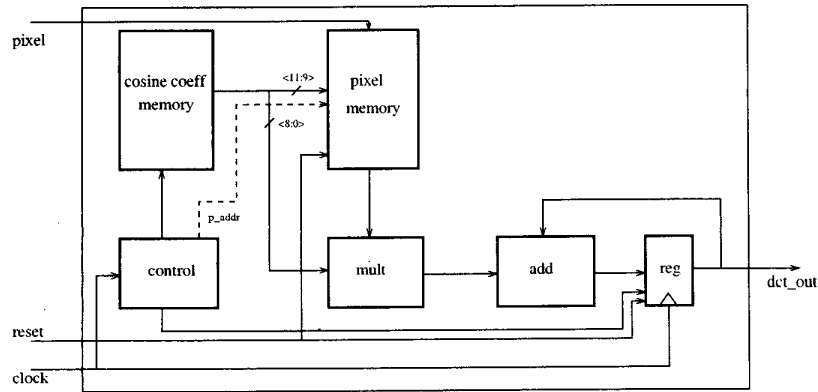


Figure 4: A simplified DCT processor architecture

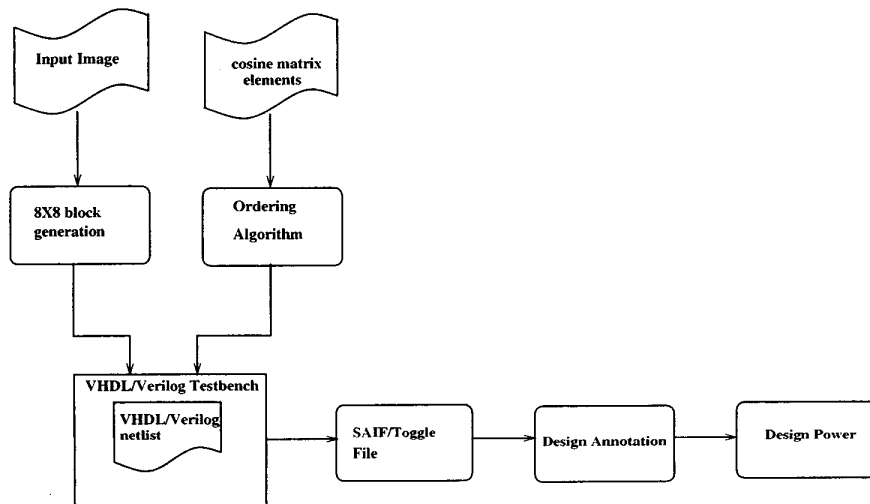


Figure 5: Framework for algorithm evaluation

- on *Circuits and Systems*, pp. 285 – 288, IEEE, 31 May - 3 June 1998.
- [2] L. G. Chen, J. Y. Jiu, H. C. Cheng, Y. P. Lee, and C. W. Ku, "Low power 2D DCT chip design for wireless multimedia terminals," in *International Symposium on Circuits and Systems*, vol. 4, pp. 41 – 44, IEEE, 31 May to 3 June 1998.
- [3] A. P. Chandrakasan and R. W. Brodersen, *Low Power Digital CMOS Design*. Kluwer Academic Publishers, 1995.
- [4] A. T. Erdogan and T. Arslan, "Data block processing for low power implementation on single multiplier CMOS DSP processors," in *International Symposium on Circuits and Systems*, IEEE, 31 May - 3 June 1998.
- [5] T. Xanthopoulos and A. P. Chandrakasan, "A low power DCT core using adaptive bitwidth and arithmetic exploiting signal correlations and quantization," in *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 740–750, May 1999.
- [6] S. Masupe and T. Arslan, "Low power DCT implementation approach for VLSI DSP processors," in *International Symposium on Circuits and Systems*, IEEE, 30 May - 2 June 1999.
- [7] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*. Kluwer Academic Publishers, 1995.