Design and Optimization of the VLSI Architecture for Discrete Cosine Transform Used in Image Compression

Mario Kovač[1], Mario Žagar[1] and N. Ranganathan[2]

[1] Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, CROATIA
[2] Center for Microelectronics Research, Department of Computer Science and Engineering, University of South Florida, Tampa, USA

Keywords: DCT, data compression, VLSI, computer architecture, algorithms, multimedia.

1. Introduction

Presentation of images plays a significant role in today's information exchange. Numerous applications that have been introduced in last few years, such as video teleconferencing, HDTV, wirephoto, fax, computer tomography, interactive visualization, multimedia and other, are based on image presentation and distribution procedures. Disadvantage of using digital images in these applications is enormous amount of space needed for image storage. For example, a 1024 × 1024 color image with 24 bits per pixel requires 3.15 M bytes in the raw form. It is obvious that efficient handling of images is possible only with the introduction of data compression techniques. Data compression is the reduction or elimination of redundancy in order to achieve savings in storage and communication costs. Data compression techniques can be classified in many ways, but the most common classification defines two basic categories: lossless and lossy. In lossless methods, the exact original data is recovered while in lossy methods only a close approximation of the original data can be obtained. Lossless methods are used in applications where no loss of information is allowed, such as text compression, medical imaging, and similar. Lossy methods are mostly used in image and audio compression where designers or users can select the quality of the restored data. Disadvantage of lossless methods is that the compression ratio is relatively small and runs between 3:1 and 4:1, while lossy methods can achieve compression ratios of up to 100:1. Even with compression ratios of that magnitude, due to the large volume of data, real-time operation of image using applications requires development of very high speed implementation of image compression/decompression techniques.

In recent years, a working group known as Joint Photographic Expert Group (JPEG) consisting of three international standard organizations, International Telegraph and Telephone Consultative Committee (CCITT), International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), has established an international standard for coding and compression of continuous-tone still images. This standard is commonly referred to as the JPEG standard. The primary aim of the JPEG standard is to propose an application independent image compression algorithm that would be and aid VLSI implementation of data compression [32].

In this paper we describe design and optimization of the most critical part of the efficient single chip architecture for JPEG image compres-

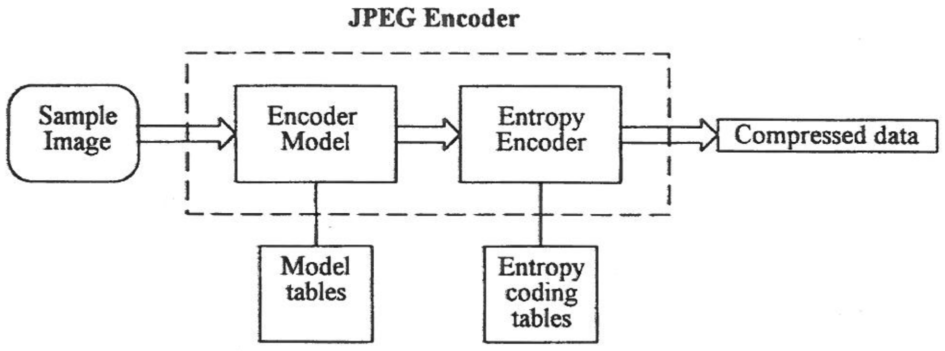


Fig. 1. JPEG Encoder

sion: the Discrete Cosine Transform (DCT) module. To maximize the throughput and reduce the silicon area required the architecture has been designed on the basis of the novel hardware algorithms, and later optimized to reduce any redundancy in the logic.

The paper is organized as follows. In the second chapter we briefly describe JPEG Compression Standard. The third chapter reviews previous work in the field. The fourth chapter presents detailed description of the design, optimization and implementation of the DCT architecture and it is followed by conclusions and references.

2. JPEG Compression Standard

The basic model for the JPEG encoder is shown in Fig. 1. The encoder model transforms the input image into an abstract representation which is more suitable for further processing. To achieve this transformation, the encoder model may require parameters stored in some model tables. The entropy encoder is a compression procedure which converts the output of the encoder model into a compressed form. Also, the entropy encoder may use tables for storing the entropy codes. Four distinct coding processes were derived based on the above described JPEG model: (i) baseline process, (ii) extended DCT-based process, (iii) lossless process and (iv) hierarchical process.

The baseline and the extended processes are also known as DCT-based processes since they use DCT within the encoder model. The lossless process uses prediction based methods within the encoder model. The hierarchical process uses the encoder model from the extended process or the lossless process. The baseline process uses Huffman codes for entropy encoding, while the other three processes use either Huffman or arithmetic. Since the focus of this paper is on VLSI implementation of the baseline process, we describe the baseline process in detail in the rest of this section. For a complete overview of the JPEG standard and the various processes, the reader is referred to [32,44].

The encoder model for the baseline process is shown in Fig. 2. The input image is divided

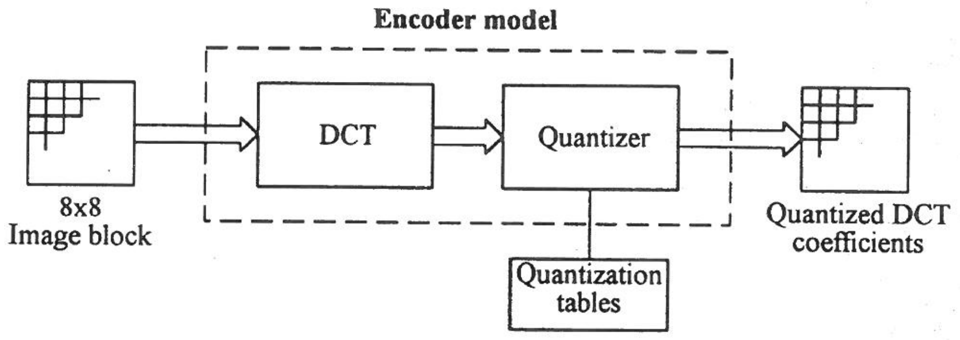


Fig. 2. JPEG Baseline Encoder Model

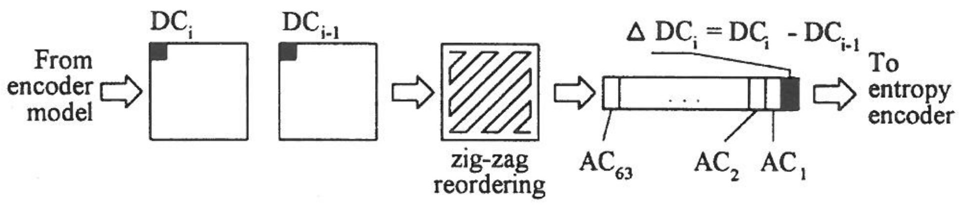


Fig. 3. Reordering of DCT output

into nonoverlapping blocks of 8×8 pixels and input to the baseline encoder. The pixel values are converted from unsigned integer format to signed integer format and DCT computation is performed on each block. DCT transforms the pixel data into a block of spatial frequencies that are called the DCT coefficients. Since the pixels in the 8x8 neighborhood typically have small variations in gray levels, the output of DCT will result in most of the block energy being stored in the lower spatial frequencies. On the other hand, the higher frequencies will have values equal to or close to zero and hence, can be ignored during encoding without significantly affecting the image quality. Selection of frequencies based on their importance can affect the quality of the final image. JPEG allows for this by letting the user predefine the quantization tables used in the quantization step following the DCT computation. The selection of quantization values is critical, since it affects both the compression efficiency and the reconstructed image quality.

The block of DCT coefficients output by the encoder model is rearranged into one dimensional data using zig-zag reordering as shown in Fig. 3. The location $(0, 0)$ of each block I contains the DC coefficient for the block, represented as DC_i. This DC coefficient is replaced by the value ΔDC_i which is the difference between the DC coefficients of block I and block $I - 1$. Since the pixels of adjacent blocks are likely to have similar average energy levels, only the difference between the current and previous DC coefficients is used, which is commonly known as differential pulse code modulation (DPCM) technique. It should be noted that due to the zig-zag reordering the high frequency coefficients that are more likely to be zeroes, get grouped at the end of the one dimensional data.

The entropy encoder details are shown in Fig. 4. In order to encode the rearranged DCT coefficients the entropy encoder uses variable length encoding based on a statistical model. In the entropy encoder the quantized DCT coefficients are converted into a stream of [runlength count, category] pairs. For each pair, there is a corresponding variable length Huffman code, which will be used by the Huffman encoder to perform the compression. The Huffman codes are stored in a table.

In order to achieve better compression results, input images are transformed very often to a

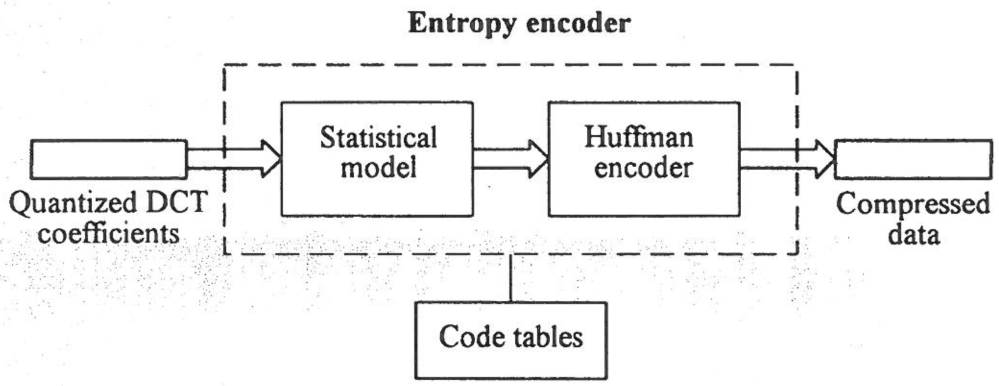


Fig. 4. JPEG Baseline Entropy Encoder

different color space (or color coordinates) representation before being input to the encoder. Although the JPEG algorithm is unaffected by the color, since it processes each color independently, it has been shown that by changing the color space, the compression ratio can be significantly improved.

3. Related work

Discrete Cosine Transform, a key function in the JPEG compression/decompression process, has been widely used in many applications and hence there is a vast amount of research work published on this topic. Of particular interest are the papers discussing hardware implementation approaches. It is well known that two-dimensional DCT computation can be implemented as a sequence of two one-dimensional DCT's which is commonly referred to as the separability property. It is simpler to implement this approach in hardware. It was shown by Haralick [21] that the DCT of N points can be computed using two N-point FFT's by exploiting the symmetry of the inputs. Later, Tseng and Miller [49] showed that the DCT can be obtained more efficiently by just computing the real part of the first N coefficients of the 2N-point DFT. The computation of 8–point DCT needed for JPEG can be replaced by 16–point DFT computation followed by scaling. An optimum form for 16–point DFT was developed by Winograd [53]. Arai, Agui and Nakagima adapted Winograd's solution for 8–point DCT reducing the computation by means of the symmetry property [7]. The hardware implementation of one-dimensional scaled DCT in our architecture is based on the algorithm by Arai et.al. [7]. Their computational flowgraph requires 5 multiplications, 29 additions and 16 two's complement operations (referred to as multiplications by -1 by Arai et.al. [7]). In our paper [38] we have described a new algorithm for one dimensional DCT computation which further reduces the number of operations. By using this approach one dimensional, scaled DCT introduces a 25% reduction in the number of two's complement operations compared to Arai et.al. In this paper we will describe the design and optimization process of the VLSI architecture that is based on that algorithm.

A few special purpose VLSI chips implementing the JPEG baseline compression standard have been built and successfully commercialized. The Intel's i750 video processor [2,3,26] consists of two chips, the 82 750PB pixel processor and the 82 750DB display processor. The pixel processor can be programmed to implement the JPEG compression standard. The C-CUBE CL550 is a single chip processor for JPEG image compression and decompression [38]. The core of the chip is a compression/decompression unit consisting of the FDCT/IDCT, the quantizer, the run-length encoder/decoder and the Huffman encoder/decoder. The chip can operate at up to 35 MHZ. The chip can draw the data at rates up to 17.5 million pixels per second and produce compressed data at a rate of approximately 2 million bytes per second. Since the entropy encoder in the chip operates at a slower speed than the DCT module, a FIFO buffer is used between the two modules to avoid overflow during compression. Whenever the amount of data in the buffer reaches a certain level a delay signal is generated, which stalls the DCT computation and the data input to the system. LSI Logic announced a chipset for JPEG compression that consists of L64735 DCT processor, L64745 JPEG coder and L74765 color and raster-block converter. The chipset operates at the maximum rate of 35 MHZ and processes still image data at up to 30 million bytes per second. LSI Logic offers a single chip JPEG coprocessor L64702 designed for graphics and video applications in personal computers, engineering workstations and laser printers [40]. The chip is capable of compressing and decompressing the data at rates up to 8.25 million bytes per second with an operating frequency of 33 MHZ.

Design and optimization of high speed, pipeline architecture for DCT and category selection was one of the most demanding tasks in the development of the single chip for the JPEG compression. Our goal was to design the architecture that would be able to accept new data and compute new result every clock cycle. In this paper we will present the details of both DCT and category selection architectures that are based on the novel algorithms developed by the authors. Efficient logic that controls the operation of the circuitry is described as well.

4. DCT Architecture

Efficient hardware implementation of two dimensional DCT is feasible using the separability property of the transform. A sequence of two one-dimensional DCT's will produce the same result but the amount of logic needed to implement two 1D DCT's is significantly smaller. Another reduction in computation can be achieved if DCT coefficients are allowed to be scaled by some constant factor (which is the case with the DFT based methods, as explained in previous chapter). Since DCT is followed by the quantization procedure, as per JPEG standard, all scaling factors can be combined with the quantization factors, resulting in the notable reduction in computation. We have chosen this approach in the development of the DCT algorithm. For the sake of reference this algorithm is restated in Table 1.

Based on the above algorithm we have developed the efficient, fully pipelined, VLSI architecture with minimized control logic. It is seen from the algorithm, that if appropriate logic and control signals are designed all steps can be computed parallelly within the 8 clock cycles. Number of 8 clock cycles was chosen on the grounds of the efficiency calculation for the whole circuit.

Maximum throughput for the above algorithm, i.e. 8 pixels/clock cycle can be achieved if all computations in a step are performed parallelly (Fig. 5). In that case, DCT architecture would consist of up to 8 arithmetic units for each step. Although attractive, this approach is inappropriate for most of the applications for two major reasons: (I) JPEG decoders should be cost effective and an unreasonable amount of logic (arithmetic units) would significantly increase the chip cost, and (ii) entropy encoding portion of the JPEG architecture processes data in a serial fashion and, using the same technology, is not capable of processing 8 coefficients/clock cycle.

Step 1 :			
$b_0 = a_0 + a_7;$	$b_1 = a_1 + a_6;$	$b_2 = a_3 - a_4;$	$b_3 = a_1 - a_6;$
$b_4 = a_2 + a_5;$	$b_5 = a_3 + a_4;$	$b_6 = a_2 - a_5;$	$b_7 = a_0 - a_7;$
Step 2 :			
$c_0 = b_0 + b_5;$	$c_1 = b_1 - b_4;$	$c_2 = b_2 + b_6;$	$c_3 = b_1 + b_4;$
$c_4 = b_0 - b_5;$	$c_5 = b_3 + b_7;$	$c_6 = b_3 + b_6;$	$c_7 = b_7;$
Step 3 :			
$d_0 = c_0 + c_3;$	$d_1 = c_0 - c_3;$	$d_2 = c_2;$	$d_3 = c_1 + c_4;$
$d_4 = c_2 - c_5;$	$d_5 = c_4;$	$d_6 = c_5;$	$d_7 = c_6;$
$d_8 = c_7;$			
Step 4 :			
$e_0 = d_0;$	$e_1 = d_1;$	$e_2 = m_3 * d_2;$	$e_3 = m_1 * d_7;$
$e_4 = m_4 * d_6$	$e_5 = d_5;$	$e_6 = m_1 * d_3;$	$e_7 = m_2 * d_4;$
$e_8 = d_8;$			
Step 5 :			
$f_0 = e_0;$	$f_1 = e_1;$	$f_2 = e_5 + e_6;$	$f_3 = e_5 - e_6;$
$f_4 = e_3 + e_8;$	$f_5 = e_8 - e_3;$	$f_6 = e_2 + e_7;$	$f_7 = e_4 + e_7;$
Step 6 :			
$S_0 = f_0;$	$S_1 = f_4 + f_7;$	$S_2 = f_2;$	$S_3 = f_5 - f_6;$
$S_4 = f_1;$	$S_5 = f_5 + f_6;$	$S_6 = f_3;$	$S_7 = f_4 - f_7;$

where: $\mathrm{a_i}$ input elements (0 i 7)
$\mathrm{S_i}$ scaled DFT coefficients (0 i 7)
$\mathrm{m_i}$ fixed multipliers:
$\mathrm{m_1} = \cos(4/16);$
$\mathrm{m_2} = \cos(6/16);$
$\mathrm{m_3} = \cos(2/16) - \cos(6/16);$
$\mathrm{m_4} = \cos(2/16) + \cos(6/16);$

Table 1.

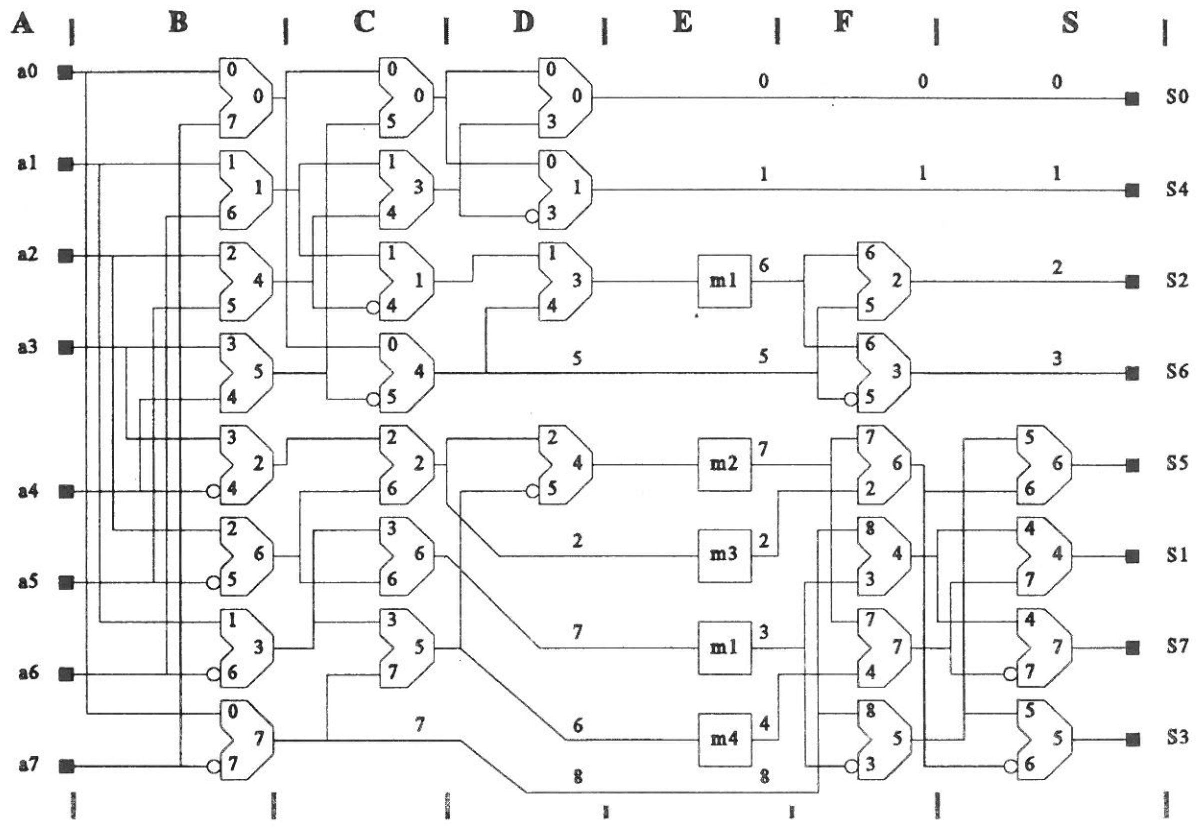


Fig. 5. 1D DCT Module

We have chosen another approach which is not computationally that intensive and which maximizes performance/silicon ratio. Since entropy encoding logic can't accept more than one element per clock cycle, DCT was designed to have exactly the same throughput. The algorithm allows that all operations are performed within the common time frame of eight clock cycles by using only one arithmetic unit per step. This is an optimal performance since it provides maximal output rate that further logic can handle (max. performance); at the same time, any further reduction of the number of arithmetic units would reduce that output rate (min. silicon needed). The algorithm was developed for the VLSI implementation, with special care to remove any operand feedbacks. While designing the architecture, several rules have been followed to enable maximum throughput:

– All stages should follow the same speed of operation (1 pixel/clock cycle)

– No intermediate result of the operation should be forwarded from one step to the next one until all operations in that particular step are completed. This rule allows fine optimization of the architecture for each step, maintaining the consistency of the overall architecture.

– Based on the statistical information, arithmetic units within each stage should be optimized for speed and silicon.

Following the above rules we have designed a 1D-DCT module that maps our DCT algorithm in the hardware. The architecture consists of six partitions as shown in the Fig. 6. so that each step in the algorithm corresponds to a partition in the architecture. Each partition contains a register set (RS), an arithmetic unit and the associated control logic. To enable parallel execution of operations each register set was designed to be able to concurrently evaluate algorithm formulas and receive new operands to be used later in the process. For that reason each register set contains several register pairs where a register pair is added for every input variable

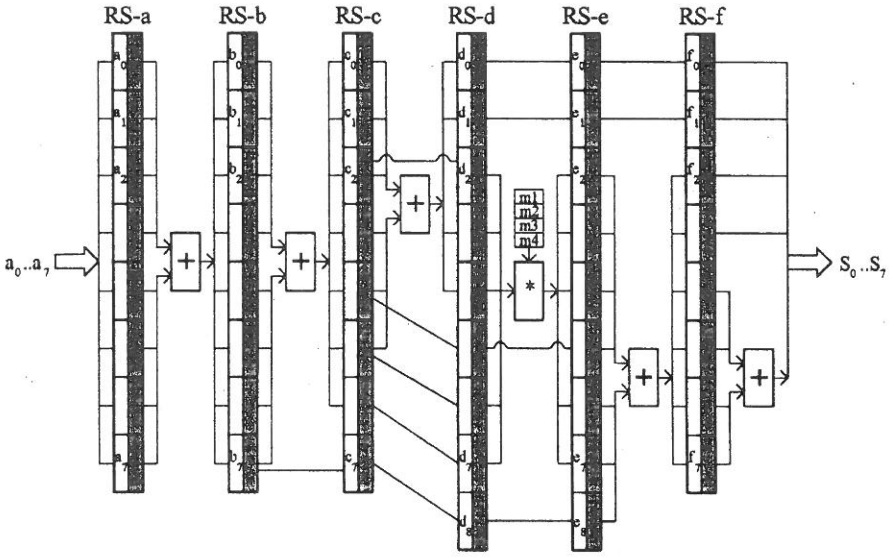


Fig. 6. Proposed DCT Architecture

in the algorithm step. As mentioned earlier, all the computations for a single step are performed within 8 clock cycles time frame. The circuit accepts one pixel per clock cycle and the entire processing is performed as a linear pipe. During each clock cycle input pixels are stored in one of the eight registers from the left column of the register set RS-a. After eight clock cycles, the left column is filled with eight data elements that are now ready for processing, and the entire column is copied onto the corresponding registers in the right column. During the next eight clock cycles the adder logic performs the computations as in step 1 of the algorithm while the left column keeps receiving new input data. A similar process occurs in each of the partitions simultaneously. By using this approach we have successfully created the architecture that utilizes arithmetic units to the maximum extent, providing the required output data rate.

Another goal in the design process was to optimize arithmetic units (adders, multiplier) to

a) reduce DCT architecture latency

b) reduce silicon area

Both of the above tasks are closely related. Reduction of the circuit latency can be achieved by the logic reduction, which will then result in a smaller silicon area. Reduction of logic was based on the theoretical analysis performed by the authors. During the analysis we were looking for max. values that can be expected as a result of every operation. This gave us the maximum number of bits needed for the result. In addition, we have intentionally introduced reductions in the output precision and calculated the error caused by that reduction. Since JPEG allows small variations in the results, we performed over 300 simulations with different reductions to derive the optimal size of the arithmetic units. After the sizes were defined we focused our efforts on the design of high speed CLA's (Carry Lookahead Adders) with one, two, three and four bit inputs. These optimized adders were then used for designing larger adders and a multiplier. Every adder within the DCT architecture was designed as a cluster of several CLA's which resulted in the high speed operation. All adders, including the largest adder (14 bit) that is part of the second 1D-DCT module, were extensively simulated and it was shown, using the worst-case timing analysis, that they could perform operations within a single clock cycle.

Multiplication unit has been designed as a reduced Wallace tree multiplier with seven segments. As known from the literature, the most complex operation in the Wallace tree multiplier takes place during the final addition of intermediate results, while other steps are simple and straightforward. Hence, our 7 segment Wallace tree multiplier requires 4 clock cycles to perform multiplication. First six steps in the process of the multiplication are performed us-

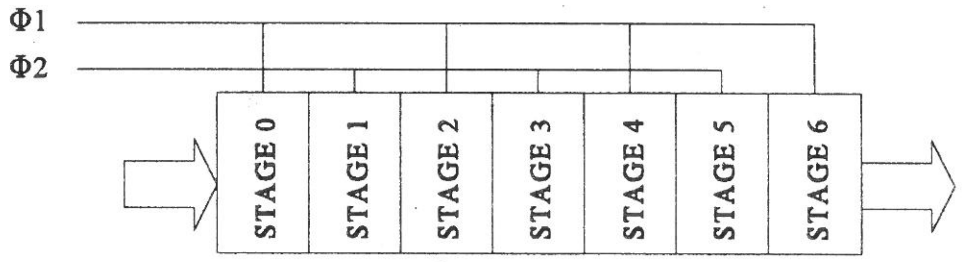


Fig. 7. Proposed DCT Architecture

ing the CSA's (Carry Save Adders) during 3 clock cycles, while the final step is performed in the 4th clock cycle. Since the optimization of the multiplier requires significant design time, it was decided to design and optimize one multiplier and reuse the design in both 1D-DCT modules. Fig. 7 shows pipelined organization of the multiplier stages. Due to the limited page width and size of the multiplier schematics it was not possible to show all details in this figure.

In one multiplier design we also addressed the fact that second operands can be chosen from the set of only 4 constants (m_1, m_2, m_3 and m_4) listed in the Table 2. Using the simulation we have slightly reduced the number of bits for these constants to reduce multiplier logic. The principle was to try to eliminate 1's from certain bit position for all four constants. If this was possible then this bit position was excluded from the list of partial sums in the multiplication. Without significantly affecting the precision we have arrived to the new constants that are listed in Table 3. In this table we show final values for the constants, together with the error introduced and the ratio of 1's after and before this rounding.

After the simulation of a circuitry was performed and arithmetic units optimized in size, we were able to define the final architecture of the DCT module. Table 4 lists final sizes for elements of the first 1D-DCT circuit. Latency of the arithmetic units introduced 'phase shifting' of the control signals between register sets. For certain algorithm steps it was possible to compensate this delay as the circuitry is not

$$
\begin{aligned}
m_1 &= \cos\frac{4\pi}{16} = 0.70710678 = (0.1011\,0101\,0000\,0100\,1111)_2 = (0.B504F)_{16}\\
m_2 &= \cos\frac{6\pi}{16} = 0.38268343 = (0.0110\,0001\,1111\,0111\,1000)_2 = (0.61F78)_{16}\\
m_3 &= \cos\frac{2\pi}{16} - \cos\frac{6\pi}{16} = 0.5411961 = (0.1000\,1010\,1000\,1011\,1101)_2 = (0.8A8ED)_{16}\\
m_4 &= \cos\frac{2\pi}{16} + \cos\frac{6\pi}{16} = 1.30656296 = (1.0100\,1110\,0111\,1010\,1110)_2 = (1.4E7AE)_{16}
\end{aligned}
$$

Table 2. Constants used for DCT

$m_1 = (0.B50)_{16}$	$\Delta = (0.0001)_{10}$,	5/10
$m_2 = (0.620)_{16}$	$\Delta = (0.0003)_{10}$,	3/11
$m_3 = (0.8A9)_{16}$	$\Delta = (0.0001)_{10}$,	5/10
$m_4 = (1.4E8)_{16}$	$\Delta = (0.00005)_{10}$,	6/13

Table 3. New DCT

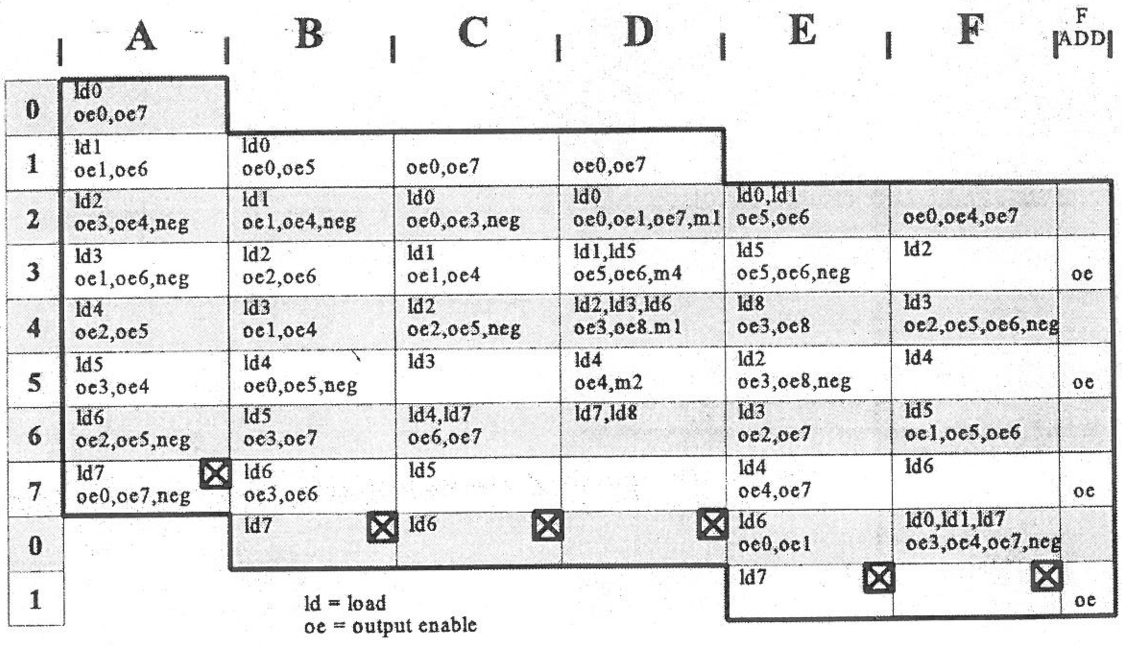
DCT 1		
1	a0,. . . ,a7 : 8 bits	+ : 8 bits
2	b0,. . . ,b7 : 9 bits	+ : 9 bits
3	c0,. . . ,c6 : 10 bits; c7 : 9 bits	+ : 10 bits
4	d0,. . . ,d7 : 11 bits; d8 : 9 bits; m : 10 bits	: 11×10 bits
5	e0,. . . ,e8 : 10 bits	+ : 10 bits
6	f0,. . . ,f7 : 11 bits	+ : 11 bits

Module		
	registers : width	arithm. unit : width

Table 4. Width of the devices in the DCT module

used in all 8 clock cycles and it was possible to execute concurrently two operations from a single step. Fig. 8 shows all details of the control needed for the high speed operation of the proposed 1D-DCT VLSI architecture. Bold X-boxes represent the moment when values are copied from input to the output registers within the register set. As it can be seen from the figure control logic for the whole DCT module can be done using the single three-bit counter and 3-to-8 decoder. The counter is directly connected to the system clock while decoder outputs are connected to the appropriate control lines of every register set and arithmetic unit. Such efficient control approach significantly improved the DCT performance and allowed faster clock speeds. DCT circuit has a latency of 50 clock cycles calculated as 9 clock cycles for stages RS-a and RS-b and 8 clock cycles for all other stages. The same architecture is replicated for column-wise DCT computation.

After the design and simulation under worst-case timing conditions, we verified whether the architecture was capable of processing one pixel in every clock cycle at the frequency of 100 MHZ resulting in the compression speed of 30 1k by 1k color images per second. By means of the Cadence Opus design tools the architecture was mapped into the VLSI chip using the 2 micron CMOS technology. The chip was designed using the 2–phase non-overlapping clocking scheme and fitted on the 6.8 by 6.9 mm^2 MOSIS frame. Prototype samples of the



	A	B	C	D	E	F	F ADD
0	ld0 oe0,oe7						
1	ld1 oe1,oe6	ld0 oe0,oe5	oe0,oe7	oe0,oe7			
2	ld2 oe3,oe4,neg	ld1 oe1,oe4,neg	ld0 oe0,oe3,neg	ld0 oe0,oe1,oe7,m1	ld0,ld1 oe5,oe6	oe0,oe4,oe7	
3	ld3 oe1,oe6,neg	ld2 oe2,oe6	ld1 oe1,oe4	ld1,ld5 oe5,oe6,m4	ld5 oe5,oe6,neg	ld2	oe
4	ld4 oe2,oe5	ld3 oe1,oe4	ld2 oe2,oe5,neg	ld2,ld3,ld6 oe3,oe8.m1	ld8 oe3,oe8	ld3 oe2,oe5,oe6,neg	
5	ld5 oe3,oe4	ld4 oe0,oe5,neg	ld3	ld4 oe4,m2	ld2 oe3,oe8,neg	ld4	oe
6	ld6 oe2,oe5,neg	ld5 oe3,oe7	ld4,ld7 oe6,oe7	ld7,ld8	ld3 oe2,oe7	ld5 oe1,oe5,oe6	
7	ld7 oe0,oe7,neg ☒	ld6 oe3,oe6	ld5		ld4 oe4,oe7	ld6	oe
0		ld7 ☒	ld6 ☒	☒	ld6 oe0,oe1	ld0,ld1,ld7 oe3,oe4,oe7,neg	
1					ld7 ☒	☒	oe

ld = load
oe = output enable

Fig. 8. DCT Module control signals

Fig. 9. Chip prototype

chip were fabricated under MOSIS[38] (Fig. 9).

5. Conclusions

DCT architecture represents the key module in the JPEG compression/decompression chip. In this paper we have presented design, optimization and implementation issues of an efficient, highly parallelized architecture that performs DCT computations. The architecture is organized as a multistage linear pipeline with minimal control logic using only one 3 bit counter and a 3-to-8 decoder. The circuitry within the architecture has been optimized for both speed and size, resulting in only few major computing elements. Intensive simulation using Verilog and other Cadence design tools has shown that the architecture can operate with clock frequency of more than 100 Mhz. As a result, DCT module can transform 30 color images of 1024 $\times$ 1024 pixels per second. A prototype chip implementing the proposed architecture has been designed and fabricated under MOSIS.

References

[1] "IEEE Standard Specifications for the Implementation of 8 $\times$ 8 Inverse Discrete Cosine Transform", IEEE Std 1180–1990.

[2] *82750DB Display Processor databook,* INTEL, Santa Clara, September 1993.

[3] *82750PB Pixel Processor databook,* INTEL, Santa Clara, October 1993.

[4] AHMED N., NATARAJAN T., RAO K. R., "Discrete Cosine Transform", IEEE Trans. Comp., Vol. C–23, pp. 90–93, 1974.

[5] AKL S. G., *The Design and Analysis of Parallel Algorithms,* Prentice Hall, 1989.

[6] ANG P. H., RUETZ P. A., AULD D., "Video Compression Makes Big Gains", IEEE Spectrum, pp. 16–19, Oct. 1991.

[7] ARAI Y., AGUI T., NAKAJIMA M., "A Fast DCT-SQ Scheme for Images", Trans. IEICE, Vol. E71, No.11, pp. 1095–1097, 1988.

[8] ARPS R. B., "Bibliography on Binary Image Compression", Proc. IEEE, Vol 68, No. 7, pp. 922–924, 1980.

[9] BAASE S., *Computer Algorithms,* Introduction to Design and Analysis, Addison Wessley, 1989.

[10] BAER J. L., *Computer Systems Architecture,* Computer science press, 1980.

[11] BELL T. C., CLEARY J. G., WITTEN I. H., *Text Compression,* Prentice Hall, 1990.

[12] BOOKSTEIN A., STORER J.A., "Data Compression", Journal of Information Processing and Management, 1992.

[13] *C-CUBE CL550 JPEG Image Compression Processor User's Manual,* C-Cube Microsystems, Milpitas, 1992.

[14] *Cadence Opus, Custom IC Design System Manuals,* Cadence Design Systems, 1994.

[15] CHEN W., SMITH C. H., FRALICK S. C., "A Fast Computational Agorithm for the Discrete Cosine Transform", IEEE Trans. Communications, Vol. COM–25, No. 9, pp. 1004–1009, 1977.

[16] DAVISSON L. D., GRAY R. M., "Advances in Data Compression", Advances in Communication Systems 4, Academic Press, pp. 199–228, 1975.

[17] DECEGAMA A. L., *The Technology of Parallel Processing: Parallel Processing Architectures and VLSI Hardware,* Prentice Hall, 1989.

[18] DUHAMEL P., GUILLEMOT C., "Polynomial Transform Computation of the 2–D DCT", Proc. ICASSP, Albuquerque, Apr.3–6, pp. 1515–1518, 1990.

[19] FEIG E., "A Fast Scaled-DCT Algorithm", Proc. SPIE, Santa Clara, Feb 12–14, Vol.1224, pp. 2–13, 1990.

[20] GERSHO A., GRAY R. M., *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1990.

[21] HARALICK R. M., "A storage efficent way to implement the discrete cosine transform", IEEE Trans. Comp., vol. C–25, pp. 764–765, July 1976.

[22] HELD G., *Data Compression: Techniques and Applications, Hardware and Software Consideration*, Wiley, 1983.

[23] HOU H. S., "A Fast Recursive Algorithm for Computing the Discrete Cosine Transform", IEEE Trans. ASSP, Vol. ASSP–32, No.10, pp. 1455–1461.

[24] HUDSON G. P., YASUD H. A, SEBESTYEN I., "The International Standardization of a Still Picture Compression Technique", Proc. IEEE Global Telecomm. Conf., Hollywood, Nov. 28.–Dec. 1., pp. 1016–1021, 1988.

[25] HWANG K., BRIGGS F. A., *Computer Architecture and Parallel Processing*, McGraw Hill, 1984.

[26] *i750 Video Processors – Data sheet*, INTEL, Santa Clara, September 1993.

[27] IFEACHOR E. C., JERVIS B. W., *Digital Signal Processing*, Addison Wessley, New York, 1993.

[28] *IIT Vision Controller – Data Sheet*, Integrated Information Technology, 1993.

[29] *IIT Vision Processor – Data Sheet*, Integrated Information Technology, 1993.

[30] *IIT Vision Codec and Processor – Preliminary Data Sheet*, Integrated Information Technology, 1993.

[31] ISO/IEC JTC 1/SC 29/WG 10, personal communications

[32] ISO/IEC, International Standard DIS 10918, "Digital Compression and Coding of Continuous-Tone Still Images",

[33] *JESSI News*, Joint European Submicron Silicon Program, Jan. 1995.

[34] *JPEG Chipset Technical Manual*, LSI Logic, Milpitas, Jan. 1993.

[35] KOVAČ M., *High Speed VLSI Architecture for Galois Field GF(2^m) Arithmetic Operations*, M. S. thesis, University of Zagreb, 1991.

[36] KOVAČ M., RANGANATHAN N., VARANASI M., "SIGMA: A VLSI Systolic Array Implementation of a Galois Field GF(2m) Based Multiplication and Division Algorithm", IEEE Trans. VLSI, Vol. 1, No. 1., March 1993.

[37] KOVAČ M., *Chip Architecture for Real-Time Image Compression*, Ph.D. thesis, University of Zagreb, 1995.

[38] KOVAČ M., RANGANATHAN N., "JAGUAR: A Fully Pipelined VLSI Architecture for JPEG Image Compression Standard", Proc. IEEE, vol. 38, No. 2, Feb. 1995.

[39] KOVAČ M., RANGANATHAN N., "VLSI Circuit Structure for Implementing JPEG Image Compression Standard", US Patent pending

[40] *L64702 JPEG Coprocessor Technical Manual*, LSI Logic, Milpitas, July 1993.

[41] LINZER E., FEIG E., "New Scaled DCT Algorithms for Fused Multiply/Add Architectures", ICASSP, Toronto, May 14–17, pp. 2201–2204, 1991.

[42] LSI Logic, personal communications, 1994.

[43] MCMILLAN L., WESTOVER L., "A Forward Mapping Realization of the Inverse Discrete Cosine Transform", Proc. Data Compression Conference, Snowbird, March 24–27, pp. 219–228, 1992.

[44] PENNEBAKER W. B., J. L. MITCHELL, *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

[45] RAO K. R., YIP P., *Discrete Cosine Transform*, Academic Press, San Diego, 1990.

[46] SILVERMAN H. F., "An Introduction to Programming the Winograd Fourier Transform Algorithm (WFTA)", IEEE Trans. ASSP, Vol ASSP–25, No. 2, pp. 152–165, 1977.

[47] STONE H. S., *High Performance Computer Architectures*, Addison Wessley, 1989.

[48] *Testing and Analyzing an IIT VC/VP Board Design —Application Note*, Integrated Information Technology, 1993.

[49] TSENG B. D., MILLER W. C., "On Computing Discrete Cosine Transform", IEEE Trans. Comp. Vol. C–27, No. 10, pp. 966–968, 1978.

[50] U.S. picture coding committee X3L3, personal communications

[51] VETTERLY M., NUSSBAUMER H. J., "Simple FFT and DCT Algorithms with Reduced Number of Operations", Signal Processing, Vol. 6, pp. 267–278, 1984.

[52] WESTE N. H. E., ESHRAGHIAN K., *Principles of CMOS VLSI Design —A Systems Perspective*, 2nd edition, Addison–Wesley, 1992.

[53] WINOGRAD S., "On Computing the Discrete Fouriere Transform", Mathematics of Computation, Vol. 32, No. 141, pp. 175–199, 1978.

Received: November, 1995
Accepted: October, 1996

Contact address:

Mario Kovač, Mario Žagar
Faculty of Electrical Engineering
and Computing
University of Zagreb
Unska 3, 10 000 Zagreb
Croatia

N. Ranganathan
Center for Microelectronics Research
Department of Computer Science
and Engineering
University of South Florida
Tampa,USA

MARIO KOVAČ received B.S., M.S. and Ph.D. in computer science and engineering from Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia in 1988, 1991 and 1995. respectively. He has been with the University of Zagreb since 1989. where he currently holds assistant professor position. He is also serving as a consultant in hardware/software development for several international companies and involved in the electric vehicles development at the Clean Energy and Vehicles Research Center, Tampa, FL, USA.

During 1990 and 1991 he was a visiting research scholar, while in the 1993/94 he was a Fulbright scholar at the University of South Florida, Tampa, Fl.

His research interests include design and implementation of VLSI architectures for multimedia data compression, HW/SW codesign, programmable logic design and implementation of algorithms and architectures in hardware. Prof. Kovač published many papers in journals and in conference proceedings, he is coauthor of a textbook "Introduction to microcomputers" and has US patent pending for the VLSI chip architecture. He received Student Award from University of Zagreb in 1988, Fulbright Award in 1993. and the Prof. A. K. Chowdhury Best Paper Award from the 8th International Conference on VLSI Design in 1995. and Faculty of Electrical Engineering and Computing I. Lončar Award in 1997. He is a member of IEEE Computer Society has been serving in the Program Committee for the Korema Conference, International IEEE Conference on VLSI Design and European Design and Test Conference.

MARIO ŽAGAR received the B.S. and M.S. degrees in Electrical Engineering and Doctorate degree in Computer Science all from the University of Zagreb, Faculty of Electrical Engineering and Computing (FER) in 1975, 1978, 1985 respectively. In 1977 he joined FER where he is currently Associate Professor. He received British Council fellowship (UMIST — Manchester, 1983) and Fulbright fellowship (UCSB Santa Barbara, 1983/84). His research interests include Computer Architectures, Design Automation, Real-time Microcomputer Systems, Operating Systems and Open Computing. M. Žagar is author and co-author of more than eighty articles and several books: UNIX and how to use it, Introduction to microcomputers, ATLAS — microcomputer architecture simulation. Prof. Žagar is a member of the IEEE, chairman of the Croatian Open Systems Users Group — HrOpen and EurOpen Forum executive board member.

N. RANGANATHAN received the B.E. (Honors) degree in Electrical and Electronics Engineering from Regional Engineering College, Tiruchirapalli, University of Madras, India in 1983, and the Ph.D. degree in computer science from the University of Central Florida, Orlando in 1988.

He is currently an Associate Professor in the Department of Computer Science and Engineering and the Center for Microelectronics Research at the University of South Florida, Tampa. His research interests include VLSI design and hardware algorithms, computer architecture and parallel processing. He is currently involved in the design of and implementation of VLSI architectures for computer vision, image processing, pattern recognition, databases, data compression and signal processing applications. He has published widely in reputed journals and conferences and owns four U.S. patents with one pending.

Dr. N. Ranganathan is a Senior Member of IEEE, member of IEEE Computer Society, IEEE CS Technical Committee on VLSI, ACM and VLSI Society of India. He served as the Program Co-Chair for VLSI Design'94 and as the General Co-Chair for VLSI Design'95. He has served on the program committees of international conferences such as ICCD, ICPP, IPPS, SPDP, VLSI Design, CAMP and ICHPC. Currently, he serves as an Associate Editor of IEEE Transactions on VLSI Systems. He is on the editorial boards of Pattern Recognition and VLSI Design journals. He guest edited a special issue of International Journal of Pattern Recognition and Artificial Intelligence on VLSI for PR and AI to be published in April 1995. He is the editor of a two-volume series titled VLSI Algorithms and Architectures published by IEEECS Press in June 1993.

Dr. Ranganathan was the recipient of IEEE Region X Outstanding Student Award in 1982, the Rotary Foundation Fellowship in 1984, IEEE Computer Society R. E. Merwin Scholarship Award in 1987, University of South Florida Outstanding Researcher Award in 1996 and the Prof. A. K. Chowdhury Best Paper Award in International Conference on VLSI Design, 1995.