Ain Shams University
Cairo, Egypt
Faculty of Engineering

Integrated Circuits Lab

Electronics and Communications Engineering department (ECE)

B. Sc. Graduation Project
1999-Y2K

VLSI Design and Implementation of Different DCT Architectures for Image Compression

By **Sherif T. EID**

Supervisor *Prof.* **Hani F. RAGAI**

Abstract

In the last decade the advancement in data communication techniques was significant, during the explosive growth of the Internet the demand for using multimedia has increased. Video and Audio data streams require a huge bandwidth to be transferred in an uncompressed form. Several ways of compressing multimedia streams evolved, some of them use the Discrete Cosine Transform (DCT) and its inverse (IDCT) for transform coding. This report discusses different ways of implementing DCT hardware coders that could be used in a wide range of video and audio applications. It presents two different DCT cores, the first could be used in real-time video decoding, the second is a new architecture that occupies a relatively small area, and could be used in applications where high speed DCT coding is not an issue (such as JPEG still image compression for personal digital cameras and audio applications). The second core was designed to handle different word sizes (8 to 12-bits) and it can perform both DCT/IDCT, while the first one handles a fixed word size of 12-bits and performs IDCT only. Both chips were designed and implemented using Mentor Graphics Tools and targeted to a 0.8µ double metal single poly process. The first chip occupies about 8.5 mm^2 and the second occupies 2.9 mm^2 of active silicon area. Back-annotation and simulation results showed that the small chip could operate at a maximum frequency of 51MHz. Power simulation was performed on the small chip using both MGC tools Eldo and MachPA, results showed that the small chip core consumes about 4mW/MHz. The low speed core was sent for fabrication and the chip is to be delivered in the middle of July Y2K. The small chip was also implemented using Xilinx' s Spartan FPGAs and occupied approximately 3000 gates.The FPGA physical testing was made using the PC' s parallel port, a program was written in C to accept input vectors from a file and write the output results to another file, showing real-time waveforms.

Introduction

Neighboring pixels within an image tend to be highly correlated. Thus, it is desired to use an invertible transform to concentrate randomness into fewer, de-correlated parameters. The Discrete Cosine Transform (DCT) has been shown to be near optimal for a large class of images in energy concentration and de-correlating (Karhunen Loeve Transform {KLT} is the optimal transform but it isn't used because it is very difficult to practically implement)[7]. The ultimate goal of DCT is to represent the image data within a different domain using the cosine function. The DCT decomposes the signal into underlying spatial frequencies, which then allow further processing techniques to reduce the precision of the DCT coefficients consistent with the Human Visual System (HVS) model. The equation describing the N point DCT

$$X(u) = \sqrt{\frac{2}{N}} C(u) \sum_{i=0}^{N-1} x(i) \cos\left(\frac{(2i+1)up}{2N} \right) \quad (1)$$

where $C(0) = 1/\sqrt{2}, C(u) = 1$ for $u \neq 0$

Two-dimensional DCT is used for image compression. An NxN block of pixels is transformed into another NxN block of real coefficients. The equation describing the 2D DCT is

$$X(u,v) = \frac{2}{N} C(u)C(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} x(i,j) \cos\left(\frac{(2i+1)up}{2N} \right) \cos\left(\frac{(2j+1)vp}{2N} \right) \quad (2)$$

where $C(0) = 1/\sqrt{2}, C(u) = C(v) = 1$ for $u, v \neq 0$

While the inverse 2D DCT equation is

$$x(i,j) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v)X(u,v) \cos\left(\frac{(2i+1)up}{2N} \right) \cos\left(\frac{(2j+1)vp}{2N} \right) \quad (3)$$

Discrete Cosine Transform decomposes the signal into weighted sums of cosine harmonics, unlike DCT the Discrete Fourier Transform decomposes the signal into weighted sums of orthogonal sines and cosines that when added together reproduce the original signal. DFT gives the frequency content of the signal and it's phase (complex output).

Design flow

The design was completely made using Mentor Graphics Tools and it was implemented in a fully automated approach. Mentor Graphics Packaged Power Elite 2.1 (currently renamed to FPGA Advantage) was used for design entry (Renoir), simulation (ModelSim) and synthesis (Leonardo Spectrum). All of both designs' blocks were written

in the form of synthesizable VHDL code, and Renoir was for connecting these blocks together and port mapping. VHDL test-benches were written and ModelSim was used for behavioral simulation. MATLAB 5.2 from MathWorks was used to generate random binary input for test; the test-bench reads the file generated by MATLAB, assigns stimuli to the input of the core and writes the output to some binary files, which are read once again by MATLAB. MATLAB reads both input and output binary files, calculates DCT/IDCT coefficients of the input file using floating-point precision and compares them to the output of the core.

After behavioral verification was done, Leonardo Spectrum was used to synthesize the VHDL code. We targeted both chips on the AMS CYB 0.8 µm ASIC standard cells technology, and the small chip was also targeted on Xilinx Spartan FPGAs. MGC IC Station was used to generate layout for both chips. After implementation, QuickSim was use for back-annotation and timing simulation. For the small chip extra analyses were performed. A SPICE netlist was extracted including the parasitics, and power consumption simulation was made using both MGC tools ELDO and MachPA.

Implementation of the small chip was also made on FPGAs, after downloading the design on the FPGA chip, physical chip testing was performed. A program written in C language was made to apply stimuli to the chip and read input from it using the PC's parallel port. The program reads the test vectors from an input file and writes the output to another output file, and then MATLAB was used again for results confirmation.

Moderate speed IDCT core (IDCT-M)

The IDCT-M core is a moderate speed IDCT processor that can handle a video rate of 2-bits/clock cycle. If the core works at a frequency of 62.5MHz, it can handle MP@ML (Main Profile Main Level) monochrome video, and MP@LL (Main Profile Low Level) video if working at 18.5MHz. Table I shows the upper bounds and rates for different predefined video profiles [1].

Picture Level		MP@LL	MP@ML	MP@H1440	MP@HL
Sampling density	Pixels/line	352	720	1440	1920
	Lines/frame	288	576	1152	1152
	Frames/sec	30	30	60	60
Luminance pixel rate (pixels/sec)		3 041 280	10 368 000	47 001 600	62 668 800
Luminance bit-rate (Mbits/sec)		36.495	124.416	564.19	752.026
Macroblock rate		11 880	40 500	183 600	244 800
Allowable time for one block (µs)		84.2	24.7	5.45	4.08

The main idea was taken from the high-speed IDCT processor of [1], but after some modifications we were able to design a lower speed core. The core uses Chen's algorithm [4] for calculating IDCT and uses an input word size of 12 bits and 16 bits output. The 2D IDCT is performed using one or two 1D IDCT blocks and a transposition memory,

usually indirect implementation is slower than direct implementation but the area occupied is much less. Reviewing the 2D IDCT equation (3) we deduce that the 2D operation could be resolved into two 1D operations by means of row-column decomposition as follows. Figure (1) shows the block diagram of a 2D IDCT processor using two 1D DCT units. The same concept can be used for the calculation of the 2D DCT. Examining the 1D DCT equation (1) we deduce that it necessitates totally 64 multiplications when N=8, for which the chip implementation can be hardly performed. Calculating the1D IDCT requires the same number of operations. Many fast algorithms exist for the computation of 1D IDCT, but we will mainly work with Chen' s algorithm (butterfly computation) with the use of distributed arithmetic. According to the Chen' s algorithm, 8point 1D IDCT is calculated by means of the following equations.

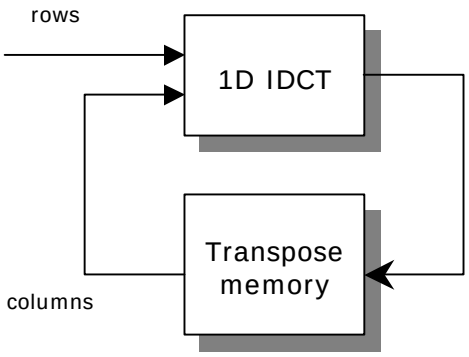


Figure (1) 2D IDCT block diagram

$$x_i = \frac{1}{2}\left[P \cdot X_i + Q \cdot X_e\right],\ x_e = \frac{1}{2}\left[P \cdot X_i - Q \cdot X_e\right] \qquad (4)$$

where

$$P = \begin{bmatrix} A & B & A & C \\ A & C & -A & -B \\ A & -C & -A & B \\ A & -B & A & -C \end{bmatrix}, \qquad Q = \begin{bmatrix} D & E & F & G \\ E & -G & -D & -F \\ F & -D & -G & E \\ G & -F & E & -D \end{bmatrix}$$

$x_i = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$ and $x_e = \begin{bmatrix} x_7 \\ x_6 \\ x_5 \\ x_4 \end{bmatrix}$ are the output inverse transformed coefficients

$X_i = \begin{bmatrix} X_0 \\ X_2 \\ X_4 \\ X_6 \end{bmatrix}$ and $X_e = \begin{bmatrix} X_1 \\ X_3 \\ X_5 \\ X_7 \end{bmatrix}$ are the input data

$A = \cos(p/4),\ B = \cos(p/8),\ C = \sin(p/8),\ D = \cos(p/16),\ E = \cos(3p/16),\ F = \sin(3p/16),$
$G = \sin(p/16)$

The algorithm shows a high degree of parallelism. Reviewing equation (4), for calculating $P \cdot X_i$ and $Q \cdot X_e$ a sophisticated distributed arithmetic is devised on the basis of the one proposed in [1]. The calculation of the previous matrix equation (9)

requires the use of multiplier accumulators and could be denoted by the following equations.

Take matrices $W = P \cdot X_i$ and $M = Q \cdot X_e$

$$W_i = \sum_{u=0}^{3} P_{iu} \cdot X_{2u+1} \tag{5}$$

$$M_j = \sum_{u=0}^{3} Q_{ju} \cdot X_{2u} \tag{6}$$

where P_{iu} and Q_{ju} are multiply coefficients of the matrices P and Q.
We can represent the input data X_l in the form of a 12-bits 2' s complement binary number using the following equation

$$X_l = -b_{l0} + \sum_{n=1}^{11} b_{ln} \cdot 2^{-n} \tag{7}$$

putting equation (7) into equations (5) and (6) we get

$$W_i = \sum_{u=0}^{3} P_{iu} \cdot \left[-b_{(2u+1)0}\right] + \sum_{n=1}^{11} \left[\sum_{u=0}^{3} P_{iu} \cdot b_{(2u+1)n} \right] \cdot 2^{-n} \tag{8}$$

$$M_j = \sum_{u=0}^{3} Q_{ju} \cdot \left[-b_{(2u)0}\right] + \sum_{n=1}^{11} \left[\sum_{u=0}^{3} Q_{ju} \cdot b_{(2u)n} \right] \cdot 2^{-n} \tag{9}$$

For calculation of equations (8) and (9), for each value of *i* and *j* we can use multiply

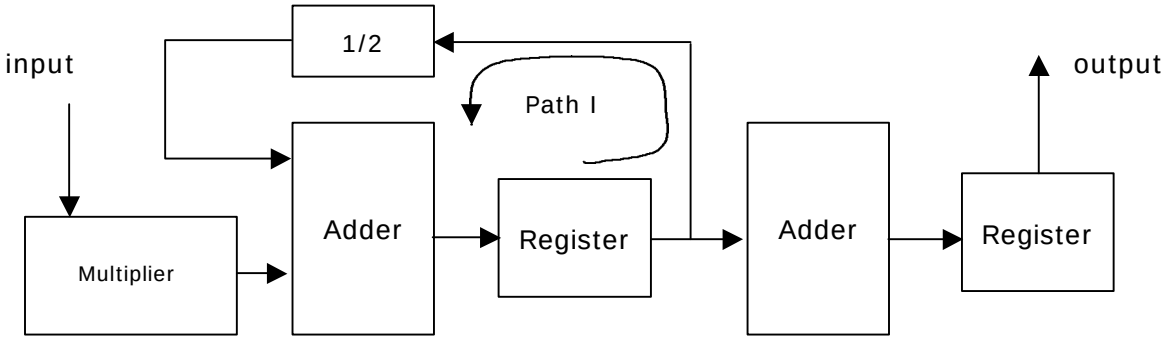


Figure (2) MAC unit

accumulate (MAC) units as denoted in figure (2). The previous MAC unit first calculates the product of $Q_{ju} \cdot \left[-b_{(2u)0}\right] + Q_{ju} \sum_{n=1}^{11} \cdot b_{(2u)n} \cdot 2^{-n}$ in 12 clock cycles for every value of u. Since that there are four values for u ($u=0,1,2,3$) it can calculate the value of W_i in 48 clock cycles. For calculation of W_i and M_j for all values of i and j, eight MAC units

have been put in parallel as in figure (3). The core is a pipelined core. The outputof the MAC units are input to a subtractor and an adder to calculate the values of both $W_i + M_j$ and $W_i - M_j$, thus calculating the values of x_i and x_e matrices (see equation (4)). Data words are input serially through the X_e and X_i inputs, bit by bit. The MAC units start calculating and outputs the inverse DCT processed outputs after 52 clock cycles.

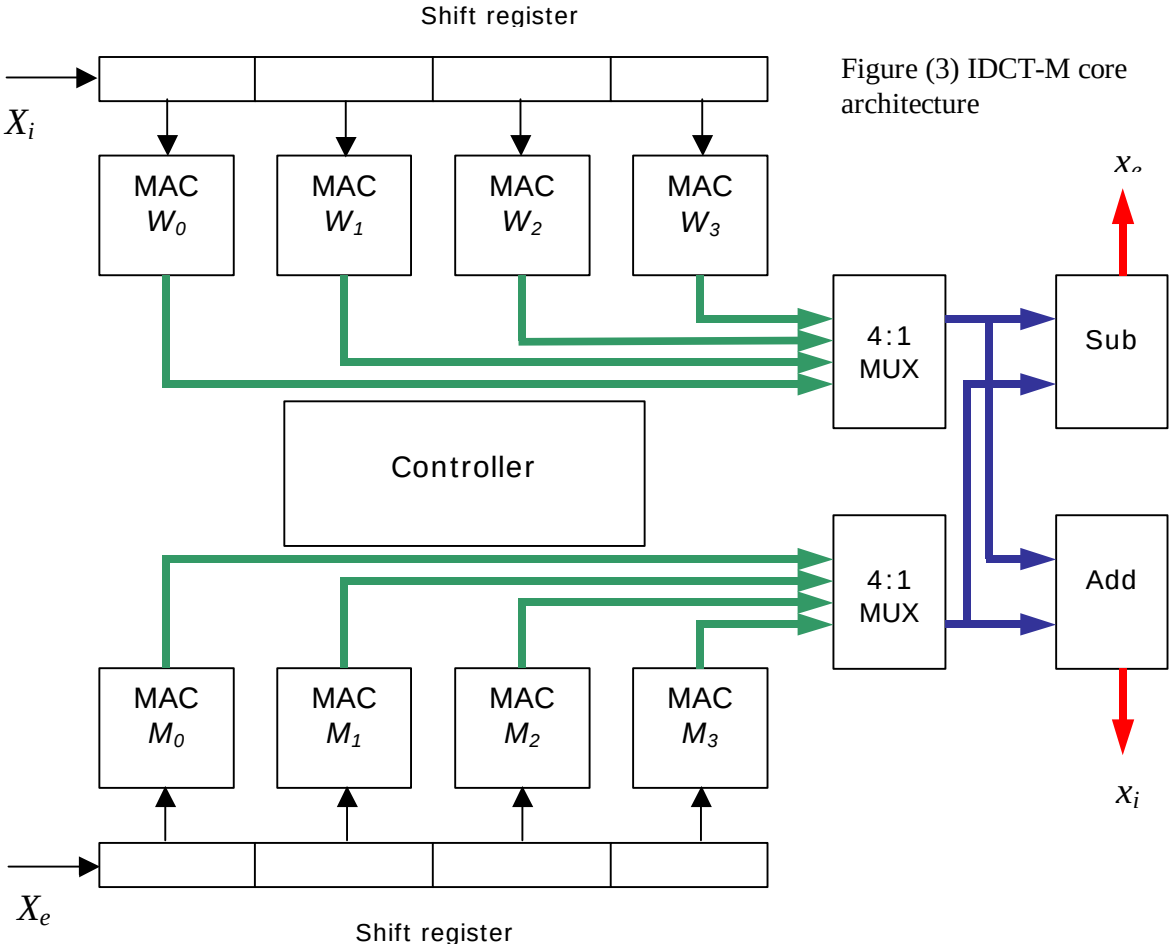
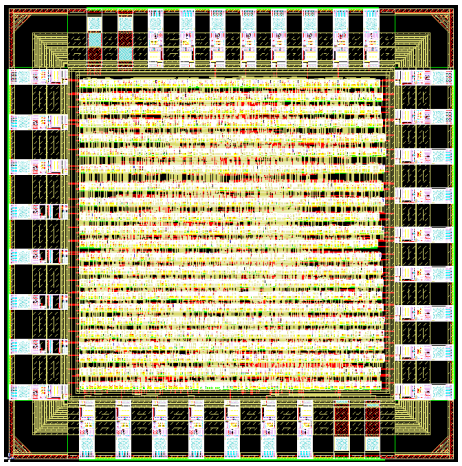


Figure (3) IDCT-M core architecture

Using MGC IC Station for generating the layout of the chip, it occupied an approximate silicon active area 8.5 mm2. Using QuickSim for back-annotation, results showed that the chip could operate at a maximum speed of 40 MHz using the 0.8-µm standard cells technology. Figure (4) shows a snapshot of the layout. The chip was too large to fabricate or implement on FPGAs.

Figure (4) Layout of the IDCT-M chip (using PADLIMITED library)



Low power DCT/IDCT core (DCT/IDCT-L)

In some portable applications like personal digital cameras high-speed DCT processing is not a main issue, when you take a shot you wait about a second or so for the image to appear on the LCD screen; and you also need that the camera works for a long period before you change or recharge the batteries. A new core suitable for such applications has been developed, the number of clock cycles needed for processing an 8x8 block of pixels is increased, but on the other side, the core area has been reduced. This reduction in area leads to reduction of the consumed power. The core' s architecture is based on the defining equation of the DCT and doesn' t use any patented algorithms. The DCT-L processor was implemented using both runs FPGA and ASIC. It was implemented using the previous 0.8µ process, and also using Xilinx Spartan FPGA family. The following sections describe the core structure and implementation steps.

Looking again at equation (1) defining the 1D DCT, when substituting N by 8 for calculation of the 8-points 1D DCT, we can gain matrix formula of equation (10)

$$X = R \cdot x \qquad \text{for forward DCT} \qquad (10)$$

and

$$x = R^T \cdot X \qquad \text{for IDCT}$$

where

$$X = \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \end{bmatrix}, \quad R = \begin{bmatrix} A & A & A & A & A & A & A & A \\ D & E & F & G & -G & -F & -E & -D \\ B & C & -C & -B & -B & -C & C & B \\ E & -G & -D & -F & F & D & G & -E \\ A & -A & -A & A & A & -A & -A & A \\ F & -D & G & E & -E & -G & D & -F \\ C & -B & B & -C & -C & B & -B & C \\ G & -F & E & -D & D & -E & F & -G \end{bmatrix}, \quad x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}$$

where the *A,B,C,D,E,F,G* are those defined in equation (4).

The matrix formula requires 64 additions and multiplications to be calculated, this was done using 2 MACs, a 64 words ROM and an 8 words special cyclic register. The first MAC is for multiplication and the second is for addition and accumulation. The consumed area decreases on the expense of the operating speed; the core calculates 8-points DCT using 8-bit words in 512 clock cycles. Figure (5) describes the block diagram of the core. The words are input through a serial bit stream, the first multiplier accumulator calculates the product of x_n by R_{mn} in a bit serial operation using a ROM.

The second multiply accumulator calculates the sum $\sum_{n=0}^{7} R_{mn} x_n$, thus calculating X_m. This operation is repeated eight times to calculate X_0 through X_7. The controller

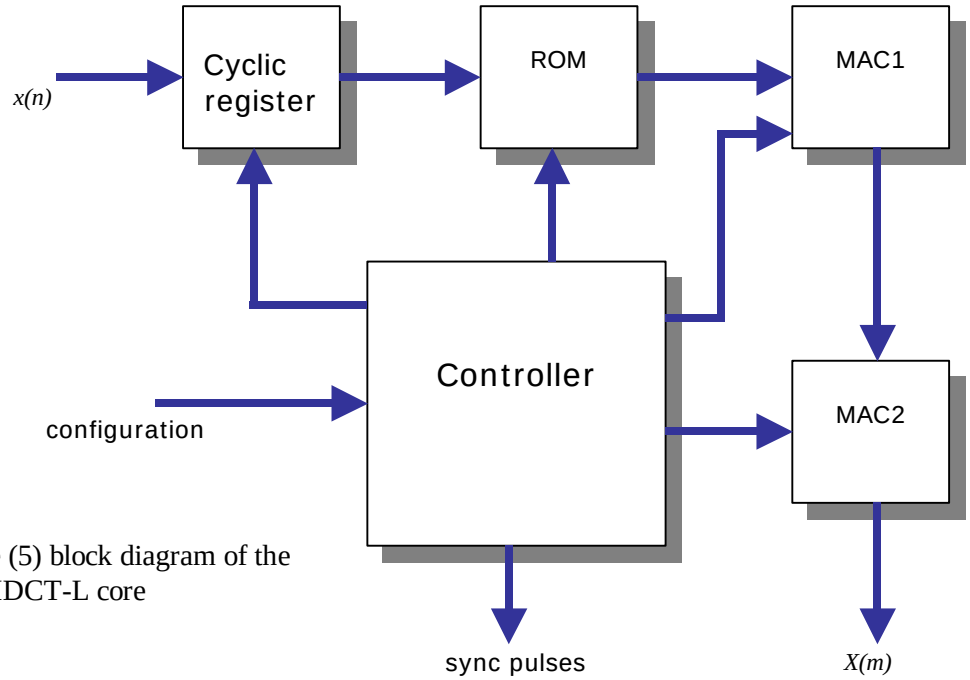


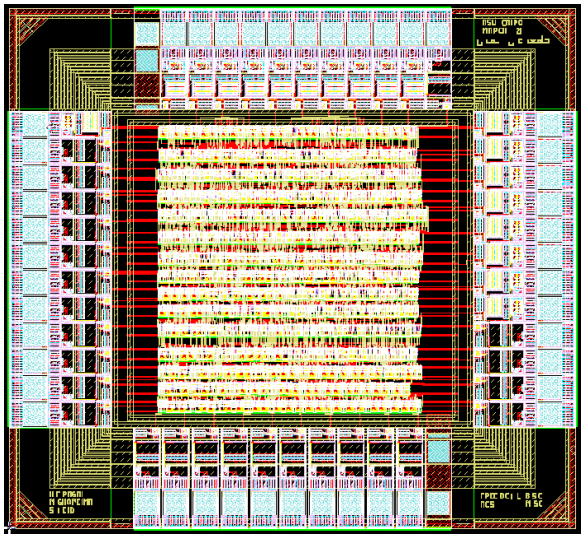
Figure (5) block diagram of the DCT/IDCT-L core

accepts configuration data such as selecting between forward or inverse DCT, it controls the timing signals for the MACs, and it outputs synchronization pulses to indicate valid instances for output reading. The ROM is used as a single bit multiplier as indicated in the previous architecture of the IDCT-M core, and transposing the matrix is simply done by switching the upper 3 bits of the ROM address with the lower 3-bits. The core is flexible; it can change the word size using a control input.

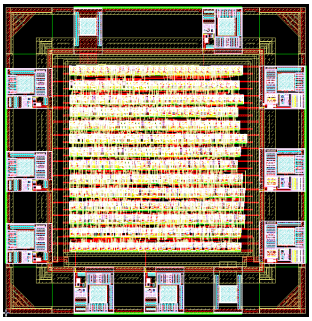
Simulation steps were done exactly like the steps of the previous core, MATLAB was used for generating the test vectors, and analysis of the output. The core was targeted on both ASIC an FPGA technologies. The core was synthesized using the area optimization flattened hierarchy and consumed about 3000 gates on the FPGA chip. The implementation flow was the same as the previous one. The timing simulation showed that the chip could operate at a maximum speed of 51 MHz. When the chip operates at that speed, it can decode a 640x640 pixels monochrome image in 1.577 seconds (12288

clocks for an 8x8 block) at a resolution of 12 bits/pixel. Figure (6) shows the layout of the small chip. The total chip area is about 2.9 mm^2; the core area is 0.91 mm^2.

Actually, the design was not sent for fabrication in this form. The chip' s area is smaller than the minimum area charge for fabrication. AMS fabs fabricate chips no less than 4 mm^2 using the 0.8µ technology, so by sending this chip we will lose die area uselessly. We had to embed two projects on the same chip giving every design its own pads; the submitted layout occupied an active area of about 5.1 mm^2. The design was submitted for fabrication in the May Y2K run, and the chip is expected to be delivered in the middle of July Y2K. Figure (7) shows the actual submitted layout.



Figure(7) The submitted chip



Figure(6) Layout of
the DCT/IDCT-L core

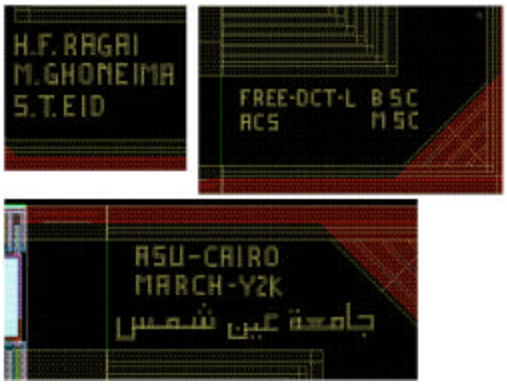


Figure (8) on chip text

We had to calculate the power consumption of the DCT/IDCT-L core. We used Mentor Graphics MachPA tool for power simulation. A Spice netlist was extracted with distributed parasitics using IC Extract and transient analysis was made. The average current was calculated through a time period of 100µs and it turned out to be 1.4mA/MHz. Thus the power consumption of the chip can be calculated, it is 7mW/MHz. At an operating frequency of 50MHz the chip will consume 350mW. ELDO was used for analog simulation, power consumption results using ELDO was 4mW/MHz.

After implementing the chip using FPGAs a testing procedure was done. The chip was tested to assure proper functionality after downloading the design on the FPGA. The functionality test was done using the PC' s Standard Parallel Port (SPP). A program written in C was used to read an input text file generated by MATLAB. It then waits for synchronization pulses from the chip and stimulates the chip' s input. The chip asserts some outputs to indicate proper instances for reading the calculated coefficients in serial. The program plots the waveforms of the signals as they are asserted. The parallel port supplies the chip with clock, clock frequencies can reach 10KHz maximum. For functionality test, the clock is set to 1 Hz to allow the observer to trace the waveforms. Figure (9) shows a snapshot caught from the program during simulation. The output is captured by the SPP and written to a text file, which could be analyzed by the previous MATLAB scripts

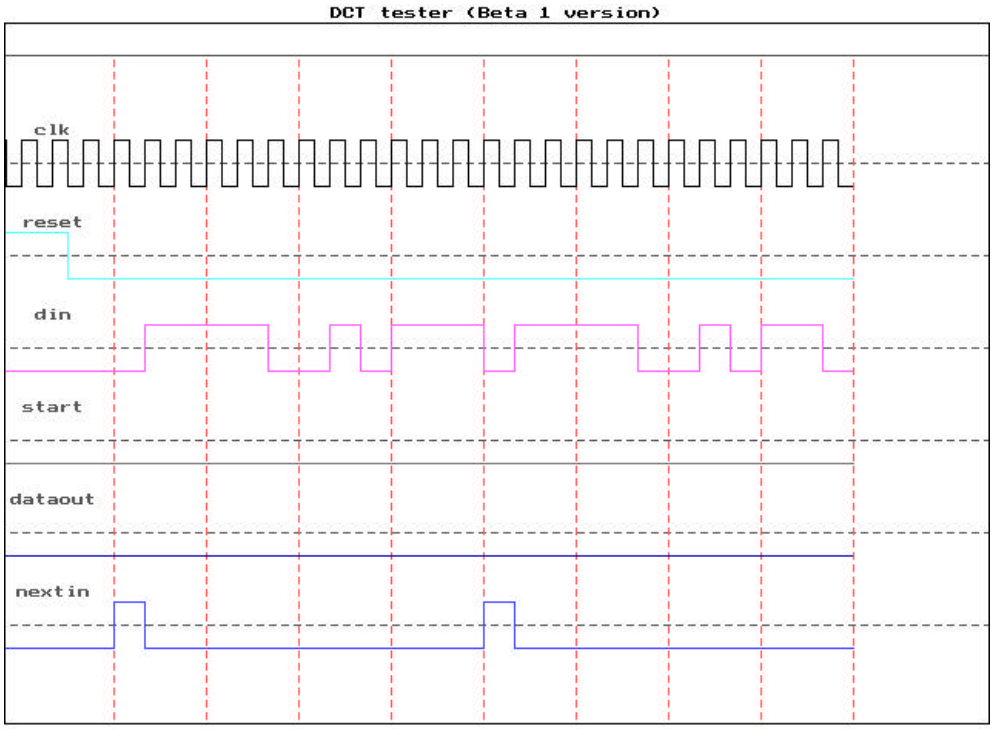


Figure (9) DCT tester program

References

[1] T. Masaki, Y. Morimoto, T. Onoye, and I. Shirakawa, "VLSI Implementation of Inverse Discrete Cosine Transformer and Motion Compensator for MPEG2 HDTV Decoding", *IEEE transactions on circuits and systems for video technology, vol. 5, no. 5, October 1995*

[2] V. Srinivasan, "Full Custom VLSI Implementation of Time-Recursive 2-D DCT/IDCT Chip", *Master Thesis, University of Maryland at College Park, College Park, Maryland 20742*

[3] K.J.R. Liu, C.T. Chiu, R.K. Kolagtla and J.F. Jala, "Optimal Unified Architectures for the Real-Time Computation of Time-Recursive Discrete Sinusoidal Transforms", *Electrical Engineering Department, Systems Research Center, University of Maryland, College Park, Maryland 20742*

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

Readings

[5] M. Kovac, N. Ranganathan and M. Zagar, "A Prototype VLSI Chip Architecture for JPEG Image Compression", *Faculty of Electrical Engineering, University of Zagreb, Av. Vukovar 39, 41000 Zagreb, CROATIA*

[6] B. Stott, D. Johnson and V. Akella, "Asynchronous 2-D Discrete Cosine Transform Core Processor", *Department of Electrical and Computer Engineering, Computer Engineering Research Laboratory, University of California at Davis, Davis, CA-95616*

[7] C-T. Chiu, "VLSI Algorithms and Architectures for Time-Recursive Discrete Sinusoidal Transforms with Applications to Real-Time Video Communications", *Ph.D. Thesis, University of Maryland at College Park, College Park, Maryland 20742*

[8] David Neal Johnson, "Design and Implementation of Low Power Discrete Cosine Transform Processors", *Master Thesis, University of California at Davis, Davis, CA-95616*

[9] SGS-THOMSON Microelectronics, "AN OVERVIEW OF THE MPEG COMPRESSION ALGORITHM", *Technical Note*

[10] Joan L. Mitchell, William B. Pennebaker, Chad E. Fogg and Didier J. LeGall, "MPEG Video Compression Standard",*Chapman & Hall*

[11] Steven W. Smith, "The Scientist and Engineer's Guide to Digital Signal Processing", *California Technical Publishing, San Diego, California* http://ww.dspguide.com

[12] "Introduction to MPEG Video compression", http://members.aol.com/symbandgrl/