VLSI Design of High-Speed Time-Recursive 2-D DCT/IDCT Processor for Video Applications

Vishnu Srinivasan, *Member, IEEE,* and K. J. Ray Liu, *Senior Member, IEEE*

Abstract— **In this paper we present a full-custom VLSI design of high-speed 2-D DCT/IDCT processor based on the new class of time-recursive algorithms and architectures which has never been implemented to demonstrate its performance. We show that the VLSI implementation of this class of DCT/IDCT algorithms can easily meet the high-speed requirements of high-definition television (HDTV) due to its modularity, regularity, local connectivity, and scalability. Our design of the 8 × 8 DCT/IDCT can operate at 50 MHz (or have a 50 MSamples/s throughput) based on a very conservative estimate under 1.2 μ CMOS technology. In comparison to the existing designs, our approach offers many advantages that can be further explored for even higher performance.**

I. Introduction

ADVANCES in various aspects of digital technology have made possible many applications of digital video such as high-definition television (HDTV), teleconferencing, and multimedia communications. These applications require high-speed transmission of vast amounts of video data. Most video standards such as HDTV video coding, H.261, JPEG, and MPEG use discrete cosine transform (DCT) as a standard transform coding scheme [1]–[4]. The DCT is, however, very computationally intensive. To realize high-speed and cost-effective DCT for video coding, one needs efficient VLSI implementations so that the high throughput requirements can be matched. There has been considerable research in efficient mapping of these algorithms to practical and feasible VLSI implementations in the recent past [5]–[9]. These have, however, employed irregular butterfly structures with global communications resulting in complex layout, timing, and reliability concerns which severely limit the operating speed and expandability in VLSI implementations.

Recently a new class of transform coding architectures based on time-recursive approach has been proposed [10]–[12]. The complexity of this class of parallel architectures is low, e.g., only $4N - 4$ multipliers are needed for computing the 2-D DCT. To perform inverse DCT (IDCT), the computational structure is the same with only an additional multiplier needed [12]. Thus, the DCT and IDCT can be naturally combined and implemented together. This class of architectures has excellent scalability, i.e., the transform size N can be made any integer by adding or deleting computational modules [10]–[12]. In addition, these are highly parallel, modular, regular, fully-pipelined, and locally-connected. Thus it is a very good candidate for high-speed video applications. Also, the architecture is very suitable for real-time applications as the time-recursive concept has been exploited to eliminate the waiting time for data to arrive. From the VLSI implementation point of view, the need for global communication is eliminated as the parallel computational IIR structures are decoupled into independent modules.

In this paper we present a novel VLSI implementation for the time-recursive 2-D DCT/IDCT processor. This class of time-recursive parallel architectures has never been designed and implemented to prove its superior properties—our goal here is to show its performance under full-custom VLSI implementation and to make comparisons with other existing VLSI designs based on different algorithms. The chip design has been carefully optimized based on appropriate choice of wordlength and device elements to meet the required signal-to-noise ratio, the design of distributed arithmetic ROM units, and transformation and redistribution of clocking and pipelined stages to maximize the throughput. Complete functionality verification of the chip has been carried out. The timing simulations of the 8 × 8 2-D DCT/IDCT chip shows that it will operate at a system clock rate of 50 MHz corresponding to a data throughput of 50 MSamples/s under 1.2 μ CMOS technology which implies that it can perform DCT/IDCT under the HDTV requirements.

The paper is organized in the following manner. We give a brief summary of the algorithm and architecture in Section II. The finite wordlength and architectural considerations are given in Section III. Section IV discusses the hardware design and the VLSI implementation aspects. A comparison to existing DCT/IDCT VLSI design is presented in Section V, followed by the conclusion in Section VI.

II. Algorithm and Architecture

The time-recursive two-dimensional DCT for a $N \times N$ image block $\{x(m,n): m = t, t+1, \cdots, t+N-1; n = 0, 1, \cdots, N-1\}$ is defined [11], [12] as

$$X_c(k,l,t) = \frac{2}{N} C(k)C(l) \sum_{m=t}^{t+N-1} \sum_{n=0}^{N-1} x(m,n) \cdot \cos \frac{\pi[2(m-t)+1]k}{2N} \cos \frac{\pi(2n+1)l}{2N} \quad (1)$$

Manuscript received November 9, 1994; revised February 23, 1995 and October 5, 1995. This paper was recommended by Associate Editor T. Nishitani. This work was supported in part by NSF Grant MIP9457397, ONR Grant N00014-93-10566 and Maryland Industrial Partnership MIPS/Micro-Star Grant.

V. Srinivasan is with the Crystal Semiconductor Corporation, Austin, TX 78741 USA.

K. J. R. Liu is with the Electrical Engineering Department and Institute for System Research, University of Maryland at College Park, College Park, MD 20742 USA.

Publisher Item Identifier S 1051-8215(96)01322-5.

where

$$C(k) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } k = 0, \\ 1, & \text{otherwise.} \end{cases}$$

The time index t in $X_c(k,t)$ denotes that the transform starts from $x(t)$. The 2-D IDCT is defined in a similar manner

$$x_c(m,n) = \frac{2}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} X_c(k,l) C(k) C(l) \cdot \cos\left[\frac{\pi[2k+1]m}{2N}\right] \cos\left[\frac{\pi(2l+1)n}{2N}\right]. \quad (2)$$

DCT/IDCT is a very computationally intensive operation. To be able to use this technique for high-throughput applications such as HDTV coding, an efficient VLSI implementation is essential. In the past, several fast algorithms have been mapped onto VLSI chips, but they are not particularly well adapted for VLSI where regularity, modularity, timing, layout complexity, and area are of more concern. The IIR algorithm [12] for the computation of the DCT is a direct 2-D method and does not require transposition, unlike more traditional row-column algorithms. The structure is derived by considering the transform operation to be a filter which transforms the serial input data into their transform coefficients.

The 1-D DCT and its inverse for a N block input data, starting from $x(t)$ and ending with $x(t+N-1)$ are defined as

$$X_c(k, t+N-1) = C(k)\sqrt{\frac{2}{N}} \sum_{n=t}^{t+N-1} x(n) \cdot \cos\left[\left(n - t + \frac{1}{2}\right)\frac{\pi k}{N}\right], \quad k = 0, 1, \cdots, N-1, \quad (3)$$

$$x_c(n, t+N-1) = \sqrt{\frac{2}{N}} \sum_{k=t}^{t+N-1} C(k-t) X_c(k) \cdot \cos\left[\left(n + \frac{1}{2}\right)\frac{(k-t)\pi}{N}\right], \quad n = 0, 1, \cdots, N-1 \quad (4)$$

where $C(k)$ is as defined in (1). To derive the transfer function for the forward 1-D DCT, consider the one-sided Z transform of the left-hand side (L.H.S.) of (3):

$$Z(\text{L.H.S.}) = \sum_{t=0}^{\infty} X_c(k, t+N-1) z^{-t} = X_{c,k}(z) z^{N-1} \quad (5)$$

Likewise, the one-sided Z transform of the right-hand side (R.H.S.) in (4) can be written as:

$$Z(\text{R.H.S.}) = C(k)\sqrt{\frac{2}{N}}\frac{1}{2} \sum_{t=0}^{\infty} \sum_{n=0}^{N-1} x(n+t)(e^{j[(2n+1)\pi k/2N]} + e^{-j[(2n+1)\pi k/2N]} z^{-t}. \quad (6)$$

TABLE I
MULTIPLIER COEFFICIENTS

Multiplier	Forward DCT	Inverse DCT
$M1$	$-2\cos(\frac{\pi k}{N})$	$-2\cos(\frac{(2n+1)\pi}{2N})$
$M2$	$(-1)^k C(k)\sqrt{\frac{2}{N}}\cos(\frac{\pi k}{2N})$	$\sqrt{\frac{2}{N}}\cos\frac{(2n+1)(N-1)\pi}{2N}$
$M3$	Not applicable	$\sqrt{\frac{2}{N}}(\frac{1}{\sqrt{2}} - 1)$

Further manipulations on (5) and (6) show the transfer function for (3) to be (7).

$$H_c(z) = \frac{X_{c,k}(z)}{x(z)} = C(k)\sqrt{\frac{2}{N}}\cos\left(\frac{\pi k}{2N}\right) \cdot \frac{(1-z^{-1})((-1)^k - z^{-N})}{1 - 2\cos\left(\frac{\pi k}{N}\right)z^{-1} + z^{-2}}. \quad (7)$$

Similarly, taking the one-sided Z transform on (4), the transfer function of the 1-D inverse DCT transform can be shown in (8), at the bottom of the page.

If we are, however, interested in only the N-block transform, all z^{-k}, where $k \geq N$ terms drop off, and (7) and (8) can be simplified to:

$$H_c(z) = (-1)^k C(k)\sqrt{\frac{2}{N}}\cos\left(\frac{\pi k}{2N}\right) \cdot \frac{(1-z^{-1})}{1 - 2\cos\left(\frac{\pi k}{N}\right)z^{-1} + z^{-2}}, \quad (9)$$

$$H_{ic}(z) = \frac{\sqrt{\frac{2}{N}}\cos\frac{(2n+1)(N-1)\pi}{2N}}{1 - 2\cos\left(\frac{(2n+1)\pi}{2N}\right)z^{-1} + z^{-2}} + \sqrt{\frac{2}{N}}\left(\frac{1}{\sqrt{2}} - 1\right)z^{-(N-1)}. \quad (10)$$

In an actual implementation this is achieved by resetting all registers every N clock-cycles. The signal flow graph (SFG) shown in Fig. 1 implements (9) or (10), i.e., the forward and the inverse 1-D DCT depending on the multiplier coefficients and the modifications to the SFG as indicated by the dashed lines. The multiplier coefficients are essentially derived from (9) and (10). Depending on whether the Forward or the Inverse transform is being computed, $M1, M2$, and $M3$ are defined as shown in Table I.

The kernel shown in Fig. 1 computes a single DCT channel coefficient based on the multiplier coefficient encoded in that particular filter. N such parallel modules (each with the appropriate multiplier coefficient corresponding to k) form

$$H_{ic}(z) = \sqrt{\frac{2}{N}}\frac{\cos\left(\frac{(2n+1)(N-1)\pi}{2N}\right) - \cos\left(\frac{(2n+1)\pi}{2N}\right)z^{-N} + z^{-(N+1)}}{1 - 2\cos\left(\frac{(2n+1)\pi}{2N}\right)z^{-1} + z^{-2}} + \sqrt{\frac{2}{N}}\left(\frac{1}{\sqrt{2}} - 1\right)z^{-(N-1)} \quad (8)$$

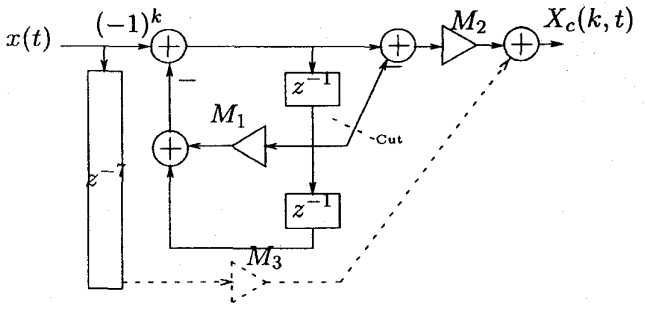


Fig. 1. IIR Structure for DCT/IDCT computation.

TABLE II
ROM VERSUS MULTIPLIER COMPARISONS

	Precision	Size	Delay
ROM (one coeff.)	12 x 12	1217λ x 1532λ	15 nS
ROM	12 x 12	1292λ x 1646λ	15 nS
ROM	12 x 16	1292λ x 1854λ	15 nS
Multiplier	16 x 16	3513λ x 1568λ	80 nS

a filter bank that computes the N coefficients of the 1-D transform. Every N cycles, the 1-D transform coefficients for a new data set are computed in parallel by the N filter bank modules. These 1-D transform coefficients are then fed through the circular shift array (CSA) into an identical but slowed down (N times) filter bank, which computes the 2-D transform of the N^2 data block. The block diagram for the 2-D DCT/IDCT architecture [11] is outlined in Fig. 2.

As we can see from the brief description, this algorithm leads to a highly modular and regular structure, as each of the channel modules (which compute the DCT/IDCT coefficient) are identical except for the multiplier coefficients. From the VLSI layout effort point of view, the kernel has to be designed only once for the 1-D and 2-D, and arrayed N times (with different multiplier coefficients) into a filter bank which computes the N 1-D and 2-D DCT/IDCT. Also, the algorithm does not impose any restriction on N—it can even be prime—so it is completely scalable. In our implementation, we have chosen $N = 8$.

III. Finite Wordlength and Architectural Considerations

In the realization of the DCT algorithm there are tight trade-offs between various criteria like accuracy, speed, and area of the chip. The implementation of the 2-D DCT/IDCT algorithm with finite precision arithmetic (due to fixed register length) introduces truncation errors. To minimize the effect of truncation errors, one needs to increase the register length, i.e., to have a larger internal bus precision. Doing so, however, results in larger area, and it also adversely affects the speed of submodules such as adders and multipliers. So we need to choose the optimum register length, which while ensuring the minimum accuracy criteria, would also lead to a high-speed implementation with small chip area.

To make sure the accumulated errors do not exceed the video coding requirements, we model the 2-D IIR DCT/IDCT architecture in C, and perform simulations to verify that peak

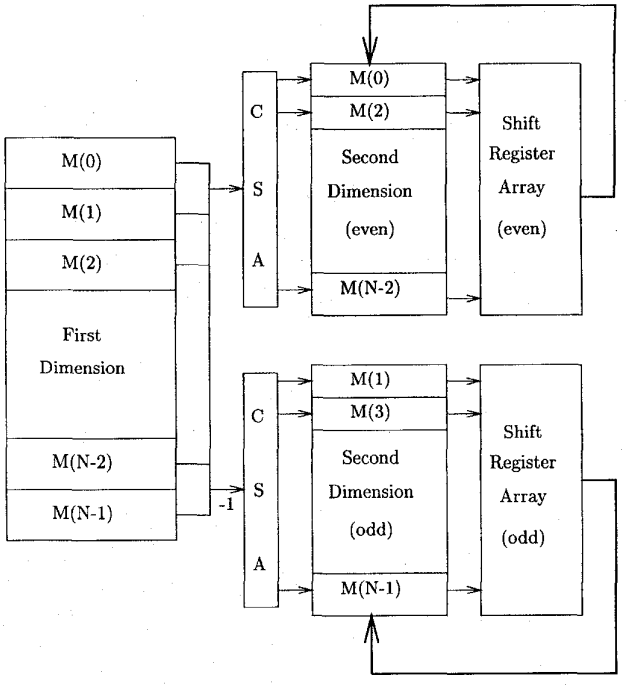


Fig. 2. Block 2-D DCT/IDCT architecture.

and average SNR requirements suitable for video coding applications are met [5]. These simulations, in conjunction with preliminary timing analysis of various submodules, helped in deciding the final architecture suitable for a high-performance high-speed 2-D DCT/IDCT chip. The truncation errors which are introduced in the system are quantified by peak signal-to-noise ratio (PSNR) and average signal-to-noise ration (SNR). The average SNR is defined as

$$\text{SNR} = 20 \log \frac{I(x,y)}{|O(x,y) - I(x,y)|} \tag{11}$$

where $O(x,y)$ and $I(x,y)$ are the output and input image pixel intensity values for position (x,y). The peak SNR is defined in a similar manner, with the only difference being that it focuses on the noise introduced due to truncation, and therefore assumes peak input intensity value [5].

ROM Lookup versus Parallel Multiplier: One of the critical submodules of this project was the multiplier implementation. This choice of multiplier architecture affects both our accuracy and area concerns and was considered early on in the architectural-design phase. There are pros and cons of using distributed arithmetic techniques [13] versus parallel multipliers [14], [15]. ROM table lookups can be done very fast as compared to a regular multiplier. Also, its accuracy is higher than regular multipliers as lookup-entries are precomputed and stored in a table. The only drawback is that, as a table lookup, the ROM size grows exponentially with input bit precision. Table II compares some of the preliminary ROM designs with the best parallel multiplier that was available to us. It was decided not to use a pipelined-fast-parallel multiplier because we did not want the pipelining issues to detract from illustrating the main concepts of the 2-D DCT/IDCT algorithm.

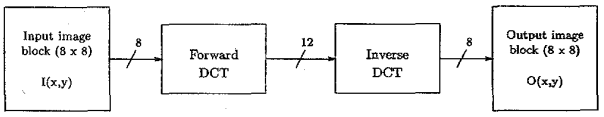


Fig. 3. Accuracy simulation block-outline.

TABLE III
FINITE PRECISION ARITHMETIC SIMULATION OF IIR STRUCTURE

# of ROM input bits	Internal bus precision	Average SNR	Peak SNR
12	12	21.1 dB	27.3 dB
12	16	37.8 dB	44.0 dB
16	16	38.0 dB	44.2 dB

The first row of Table II is a ROM which has one table lookup, while rows 2–3 ROM's are table lookups for two products. The last row is a nonpipelined general-purpose (variable-by-variable) parallel multiplier. We see that, barring a need for a 16-b wide input for the ROM, its size is smaller than the parallel multiplier. Our design of the ROM allows us to have two sets of table entries interdigitated with an area increase of only 12%. This is essential if we are to compute both the forward and inverse transforms using the same structure. Thus, based on the options we had, the choice of using ROM was made for our multiplier implementation—as not only is the ROM smaller than the multiplier, but it is over four times as fast. The timing simulations are with 2.0 μ technology parameters.

A. Finite Wordlength Simulations

For many video standards we need to ensure a minimum PSNR of 40 dB [16]. Fig. 3 illustrates the architectural simulations that help in quantifying the truncation noise in the system by computation of the PSNR. The blocks for forward and inverse DCT are C architectural models which model the finite precision arithmetic of the IIR structure for the 2-D DCT/IDCT computation, and helps in estimating the peak and average SNR. Simulations are performed for a set of different system parameters aimed at meeting minimum SNR criterion while at the same time, minimizing area and maximizing speed.

Several input images—LENA, SAILBOAT, AIRPLANE, and random data—have been used in these simulations. These images are 512 $\times$ 512 pixels, with each pixel being specified by 8-b word corresponding to 256 gray levels. As our image block size is 8 $\times$ 8 pixels, a single image yields 4096 blocks for which the 2-D DCT–IDCT is computed and the PSNR statistics collected. The 12-b DCT output has a 4.12 (integer.fractional) format. The output of the inverse DCT processor 8 b. The highlights of these simulations are presented in Table III.

As we can see, the minimum PSNR requirements are satisfied if we use a system bus precision of 16-b with a 12-b wide ROM input wordlength. A 16-b input ROM however, increases the area by 16 times, with only marginal improvement in PSNR.

We have also performed accuracy studies to quantify the max-peak-error, max-MSE, max-overall-MSE, max-mean-error, and max-overall-mean-error as outlined in CCITT Recommendation H.261-Annex 1. Results of architectural simulations with different wordlengths for our algorithm are presented in Table IV. "mdpi" is the ROM input precision, and "mdpo" is the internal wordlength. We see that, to meet the accuracy requirements, the internal bus precision certainly has to be increased. Another approach to take would be to use the normal form implementation—lattice structure which has much better numerical properties—of (9) and (10).

TABLE IV
CCITT H.261: ANNEX A—INVERSE TRANSFORM ACCURACY SPECIFICATION

Statistics	mdpi=12, mdpo=16	16,16	20,20	CCITT
Max Peak Error	6	3	1	1
Max MSE	14	2.56	0.624	0.06
Average MSE	2.634	0.532	0.017	0.02
Max Mean Error	3.679	1.553	0.064	0.015
Avg Mean Error	0.248	0.295	0.011	0.0015

IV. VLSI DESIGN AND IMPLEMENTATION

The regularity, modularity, and local interconnection property of this architecture lends itself to efficient VLSI implementations. To achieve a high-speed design (with minimum area) we use the full-custom approach. All submodules have been designed with careful regard to area and speed issues. Particular design care is taken to ensure that the critical path modules such as ROM lookup and adder are optimized. A highly hierarchical and modular strategy is employed in the chip design. By employing such a design strategy, we not only reduce the design time and effort but have also improved reliability.

A. Design and Simulation Tools/Methodology

The VLSI layout editor MAGIC is used to implement the full-custom 2-D DCT/IDCT chip. The various submodules needed for the chip design—ROM lookup, adder, latch, delay, multiplexer, and invertor—are laid out first. These submodules are characterized and their functionality is verified before they are used as macro-cells. These macro-cells are instantiated and used in the higher-level hierarchies such as "1-D Channel" and "2-D Channel" modules, which, in turn, are just larger macro-cells at the next higher level. This hierarchical approach minimizes our design effort considerably, as we need to only consider tiling, and place-and-route for various modules at the current level.

Crystal, and primarily, Spice, are used to perform the timing simulations. Crystal, a semi-interactive program, is used to estimate the critical path in a submodule between a specified set of input and output vectors. For a given change in the input vector, worst case times/delays are propagated to the output vectors. This gives us information about the critical paths. Using this as a starting point, Spice is used to perform a detailed timing analysis of critical paths of various macrocells. In our design, the slowest modules turned out to be the ROM lookup and carry lookahead adder.

Functionality verification is done using IRSIM, which is an event-driven logic-level simulator. The logic-level simulations

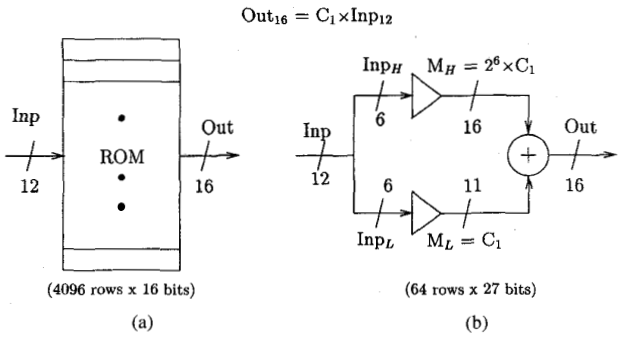


Fig. 4. ROM design strategy. (a) Naive implementation. (b) Our ROM implementation.

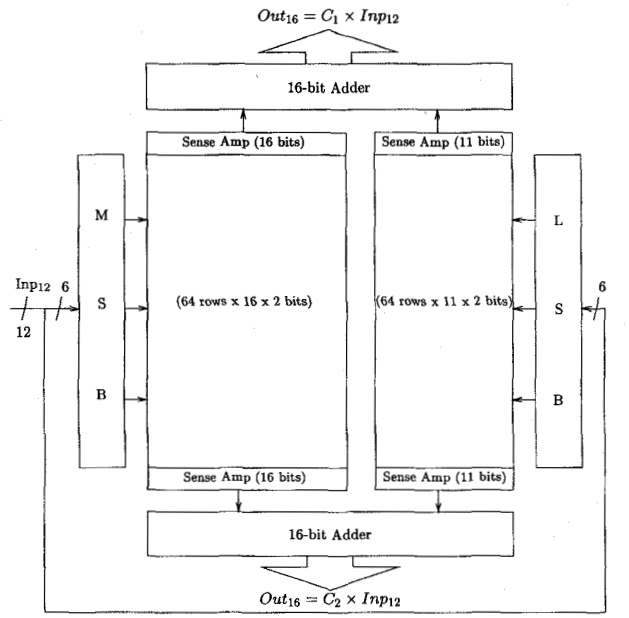


Fig. 5. ROM multiplier schematic.

are performed at a much higher level of circuit abstraction, treating the transistors as switches which are either ON or OFF. IRSIM is used to test not only the functionality of all macrocells, but is also used to perform logic-level simulations and verify the functionality at all hierarchy levels—starting from the bottommost, like that of the macrocells, to intermediate levels such as 1-D/2-D channel modules, then to the topmost level, namely the entire 2-D DCT/IDCT chip.

B. Distributed Arithmetic

This is perhaps one of the most critical submodules designed in this project. The ROM is optimized both for speed and area. The area optimization is especially critical, as the ROM is arrayed 34 times in the complete chip and is a significant portion of the chip area. Even a small reduction in ROM area would improve our chip area statistics considerably. While speedwide, ROM's are superior to multipliers, their area increases exponentially with the input word size.

The optimal system design, satisfying accuracy requirements and minimizing area, required a 16-b internal bus with

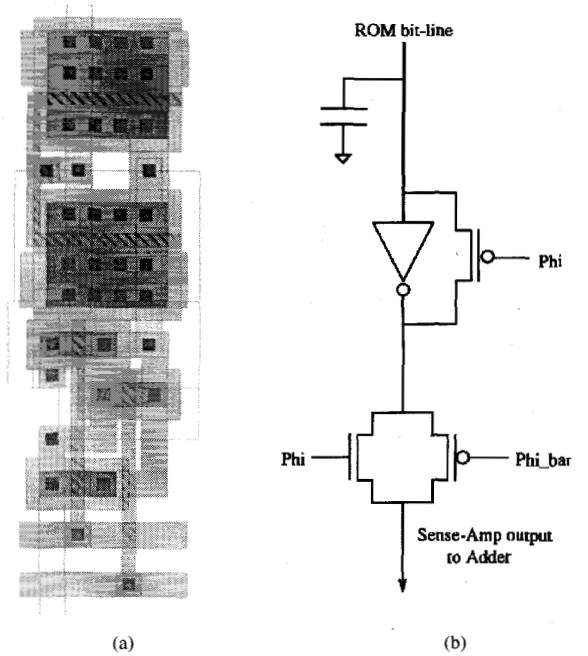


Fig. 6. Sense-Amp: (a) physical layout of SA ($26\lambda \times 94\lambda$), (b) corresponding circuit schematic.

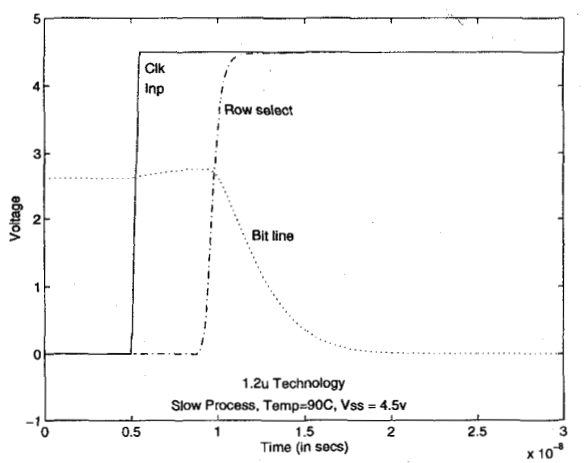


Fig. 7. ROM lookup table: spice timing simulation ($1.2\,\Phi\mu$).

a 12-b ROM input wordlength. A straightforward implementation of this ROM lookup table would need 2^{12} (or 4096) rows of 16-b words, which is too large to be implemented. However, by using partial sums method, as illustrated in Fig. 4, we can reduce the size of the lookup table to 2^6 (or 64) rows, but this requires two separate lookup tables along with a fast adder to combine the high and low order sums. Also, our ROM structure should have the capability of computing the two products, for both forward and inverse transforms.

The 12-b input is split into two 6-b words, Inp_L and Inp_H. The multiplication is effected in the following manner. The output, $\mathrm{Out} = C_1 \times \mathrm{Inp}$, is computed as:

$$\mathrm{Inp} = 2^6 \times \mathrm{Inp}_H + \mathrm{Inp}_L.$$
$$\mathrm{Out} = 2^6 \times C_1 \times \mathrm{Inp}_H + C_1 \times \mathrm{Inp}_L.$$

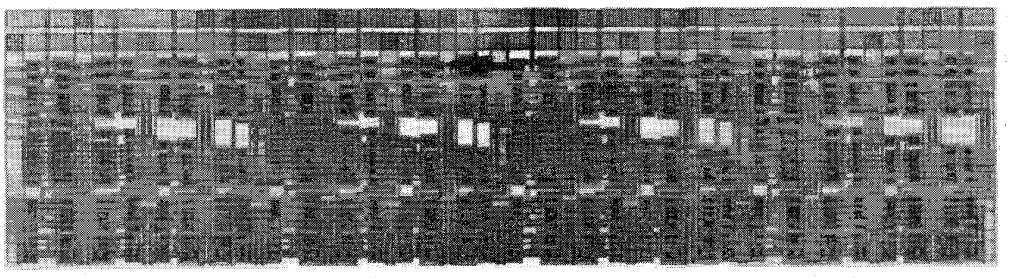


Fig. 8. Physical layout of the 16-b carry-lookahead–ripple Adder. Module dimension is $1348\lambda \times 355\lambda$.

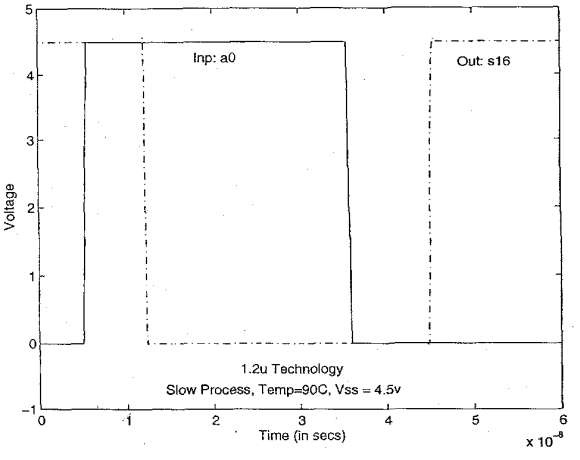


Fig. 9. 16-b Adder: spice timing simulation (1.2 μ).

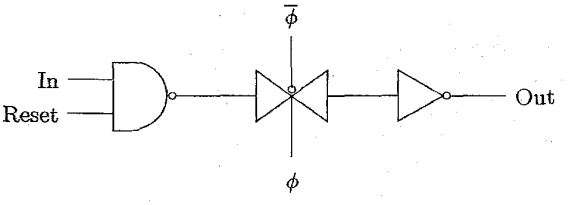


Fig. 10. Schematic of half-latch with reset control circuitry.

The two sub-products are precomputed with sufficient accuracy and storing in the ROM lookup table. The output is formed by adding the sign-extended lower order product to the higher order product. This is illustrated in Fig. 4(b). We need two tables, each with 2^6 or 64 rows only. It turns out that we need to store 16 b for the upper precomputed product and 11 b for the lower product. The lower and higher order ROM entries are added with the proper shift, taking into account the 2's complement representation.

1) Design Details: Like most ROM's, the bit lines are precharged during the first phase. During the evaluate phase, as per the stored bit-sequence, lines are selectively discharged. The main components of the ROM are tree-based row decoder, memory cells, and sense-amp. The ROM schematic is shown in Fig. 5.

6-64 Decoder: The 6-b input lines are decoded and the appropriate ROM row select line is selected. Instead of a straightforward 6-b decoder implementation, we use two 3-b decoders and an array of 64 AND gates. This reduce the layout complexity and also results in shorter access time.

ROM Core and Sense-Amp: It's an n-channel transistor connected between Vdd or GND which encodes a logic low or high. In our ROM table there are totally $(16+11) \times 2 \times 64$ or 3456 unit cells. The pitch of the unit cell is half of the pitch of the sense-amp, allowing for two sets of interdigitated lookup tables with minimal increase in area. The sense-amp shown in Fig. 6(b) precharges the bit-line to about 2.5 volts. In the evaluate phase, the bit line is pulled up to Vdd or down to GND, as the case may be, and the output is available at the transmission-gate.

2) ROM Implementation: The first ROM is designed in the regular manner—design the individual cells like sense-amp, 3-b decoders, 6-b decoders, and 0/1 unit cells in their various orientation and at UP/DOWN positions. Once a complete ROM is assembled (with dummy coefficients), it is broken up into subunits/cells, and their locations and arraying information noted. Using a perl script, new ROM's are assembled by switching the various subunits/cells in the appropriate locations. The crucial part is the encoding of the interdigitated coefficients. The sine or cosine coefficients are computed using double-precision floating point arithmetic on the Sun workstation, and given as input to the perl script. These numbers are converted to 2's complement binary, and routines called to place the "1-unitcell" or "0-unitcell" depending on the bit-value at that particular location.

The size of the basic ROM structure which encodes two multiplier coefficients is $1292\lambda \times 1854\lambda$. If we include the adders to combine the high and low order products, the ROM/adder assembly measures $2226\lambda \times 1854\lambda$.

3) Timing: Spice is used to compute the propagation delay in the ROM lookup table. The ROM netlist is extracted with layout-parasitics using 1.2 μ technology. The worst-corner (temp = 90; Vss = 4.5 v; slow-process) spice simulation for the ROM lookup-table is shown in Fig. 7. The input and clock arrive at 5 ns. The "Row select" line is the output of the 6-b decoder which selects one of the words in the ROM table. The "Row select" line charges to 50. The bit line is initially charged to about 2.6 V, and is discharged to 0.45 V at 14.6 ns. The access time is 9.6 ns.

C. Other Submodules

Various other macrocells needed in our implementation—Adder, half-latch, delay, multiplexer, and inverter are described here.

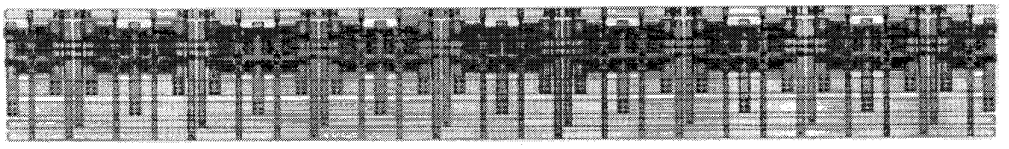


Fig. 11. Physical layout of half-latch with reset. Module dimension is $938\lambda \times 127\lambda$.

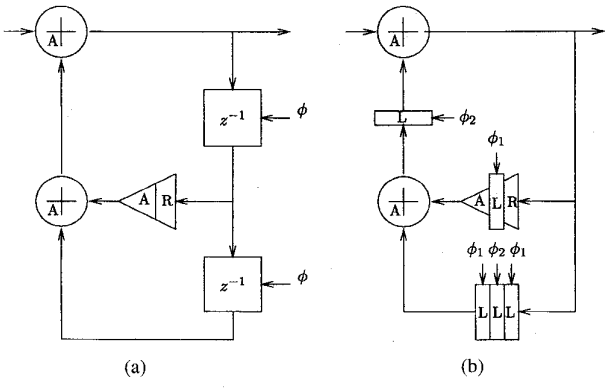


Fig. 12. Clock speedup.

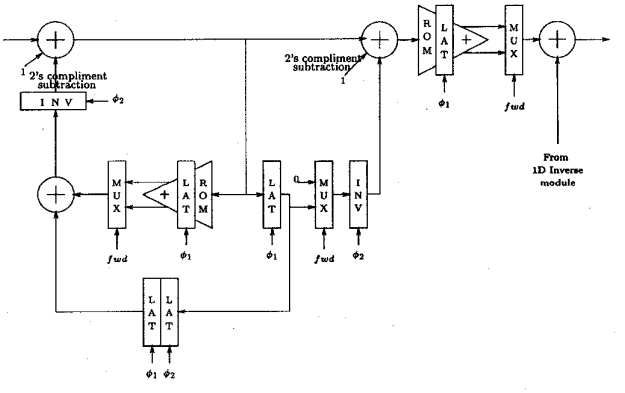


Fig. 13. 1-D IIR SFG with clocks and latches.

1) Adder: The Adder is the other critical path module. To meet our timing requirements of 50 MHz, the adder has to be faster than 10 ns (as explained in Section IV-D). A regular ripple-carry adder would have been too slow. A good compromise between area minimization and speed was the choice of using a 4-b carry-lookahead-adder (CLA). Four CLA's were connected in ripple fashion to build a 16-b fast adder [17], [18]. The adder module shown in Fig. 8 has 704 transistors and measures $1348\lambda \times 355\lambda$ units.

Timing: As the carry-in is always hardwired to 0 or 1, the longest path is from the In_{LSB} to Out_{MSB}. Simulations performed with 1.2 μ technology parameters for the slow-corner indicate a max-propagation delay of 9.1 ns. The spice timing simulation is shown in Fig. 9.

2) Half-Latch with Reset: The schematic for the half-latch is shown in Fig. 10, and the VLSI layout in Fig. 11. It is 16 b wide corresponding to the internal bus precision. Depending on its location in signal flow graph (see Fig. 13), it latches on either ϕ_1 or ϕ_2. A reset control is also provided. To design the latch, a mirrored 2-b slice is laid out, which is tested with Irsim and Spice for functionality and timing. The size of the output invertor is made sufficiently large to allow adequate driving of expected output load in the module where it will be used. Local distribution of all control signals are taken into account at the 1-b design stage itself. The 2-b slice is arrayed eight times to form the final module.

3) Delay, Delay7, and Delay8: These are shift-registers which act as delay-units for one, seven, and eight cycles. The basic design philosophy was to use what is already available. To build these shift-registers, the half-latch (described in the previous subsection) is arrayed as required. The control/clock signals required by this module are "Reset," ϕ_1, and ϕ_2. The "delay" module measures $938\lambda \times 217\lambda$, the "delay7" measures $1028\lambda \times 1549\lambda$, and the "delay8," $1028\lambda \times 1771\lambda$.

4) Multiplexers: The multiplexers are used at the output of the ROM (see Fig. 13) and at other locations in the SFG which change depending on whether the forward or the inverse transform is being computed. The size of the multiplexer is $1271\lambda \times 119\lambda$.

D. Clocking

The propagation delays of ROM and adder helps us decide retiming and clocking issues. Fig. 12 illustrates the implementation of the 1-D kernel SFG using a single-phase clock (with static latches) or a two-phase non-overlapping clock (with dynamic latches). From both speed and area viewpoint, the two-phase clocking scheme turns out to be a better choice.

A two-phase clock permits us to use dynamic latches which are simpler to design and more compact than static latches. Two-phase clocks give us the flexibility to retime the SFG such that the delays in the various critical-path segments are equalized. This is illustrated in Fig. 12(a) and (b). In Fig. 12(a), the propagation delay between any two subsequent latches is (3A + R, 2A), where A and R are the propagation delays of adder and ROM. The maximum delay between two subsequent latches is the critical path and will determine the fastest possible clock rate. In Fig. 12(a), it is (3A + R). Assuming that the adder and ROM delay are about the same, the critical path is retimed as shown in Fig. 12(b). The maximum delay now is either (A + R) or 2A; which means that the critical path delay is halved, and thus the maximum clocking rate is doubled.

E. 1-D/2-D Channel Modules

The macrocells which have been described so far are put together to form the 1-D channel module at the next higher hierarchy. The channel module shown in Fig. 13 implements the SFG of the 1-D kernel. This module, depending on the value stored in the ROM, computes the appropriate DCT/IDCT transform coefficient. Magic [16] instantiates every occurrence

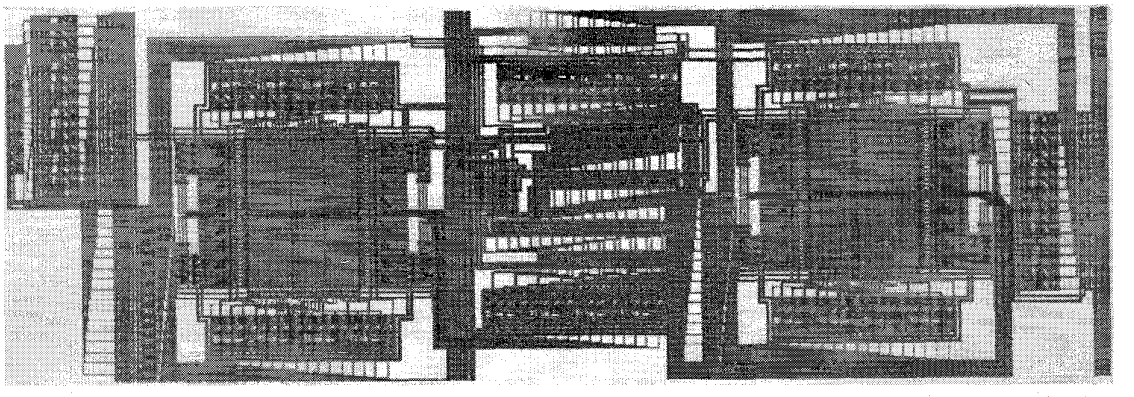


Fig. 14. VLSI layout of 1-D channel. Module dimensions are $2742\lambda \times 8029\lambda$.

of these modules, resulting in much faster layout and extraction of the module.

The signal flow graph implemented by this module is shown in Fig. 13. All but multiplier M_3 (needed only for inverse transform as indicated in Fig. 1) and the associated multiplexers and latches are included in this module. Since only one set of M_3 and its associated circuitry is needed for the entire 1-D module, it is designed separately.

The physical layout of the 1-D channel is shown in Fig. 14. All of the eight channel modules are identical except that they instantiate different ROM tables. The circuit has been laid out in a manner that facilitates easy modular development. Inter-module connections are brought to the edges of the blocks where they get connected with the other modules' wire-segments when tiled. By adopting such a methodology, we save considerably in design time and effort, and at the same time, if the modules-pitches are matched, we save in interconnection area requirement also. The power rails, input/output, and other important control signals are routed from top to bottom in each module. As they are tiled vertically, the routing of all signals is done automatically. We only have to concern ourselves with feeding these signals to the entire 1-D module, either from the top or the bottom, as local distribution of these signals is already taken care of in the design of the channel-module.

The eight 1-D channel modules are tiled one over the other to form the complete 1-D DCT. To compute the IDCT, the additional M_3 and its associated delay units is separately attached to the 1-D module, below Channel 7.

The 2-D channel-module is designed along similar lines as the previous 1-D channel-module. The SFG in Fig. 15 is almost the same, except that 8-block delay units are present in both loops. This facilitates computation of eight blocks of data in a time-displaced parallel fashion. The physical layout of this module measures $2784\lambda \times 10672\lambda$.

F. Circular Shift Array

The circular shift array (CSA) was the last module designed. It takes the eight 16-b words' channel coefficients generated by the 1-D module and feeds them serially to the 2-D module. The CSA serves another important function—of storing the

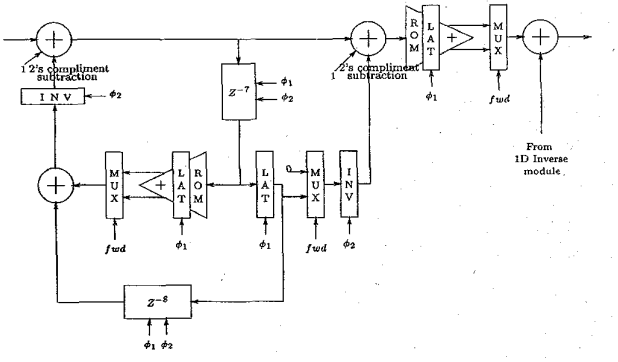


Fig. 15. 2-D IIR SFG.

1-D IDCT coefficients of the first row required for the inverse computation at the second stage.

There are several control signals for the CSA—to read data from the 1-D module, to latch it in, and shift it out serially to the 2-D module, to latch in data to help in computing of the inverse, and to hold it until required. It is during the time the 1-D module is being reset that the 1-D channel outputs are latched into the CSA. At the same time, the 2-D DCT is also being computed. In the clock period when the 1-D channel modules are being reset, the 2-D channel modules are not clocked. This is done to ensure synchronicity. For the same reason, the CSA's second set of shift registers which circulate the first row of 1-D DCT coefficients, is also not clocked in that period.

G. Control Signals

Routing of the control signals is a fairly important issue as there are quite a few control signals that need to be distributed to various clocked modules like latches and delays, and multiplexers. The basic idea is to distribute the master control signals to the high-level cells like 1-D/2-D channel modules and use a buffer to generate the local control signals, which are then distributed to all modules within that particular submodule. This is essentially a multi-level tree distribution of the control signals. By employing such a scheme, we are not only able to minimize skew, but also to improve rise and fall

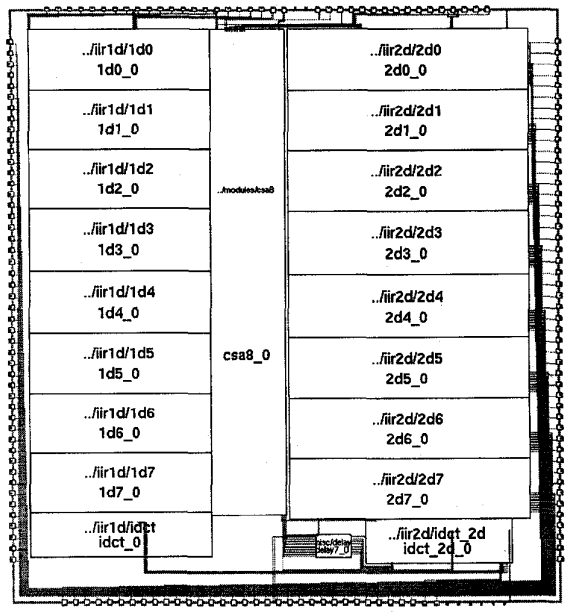


Fig. 16. Floorplan of 2-D DCT/IDCT chip.

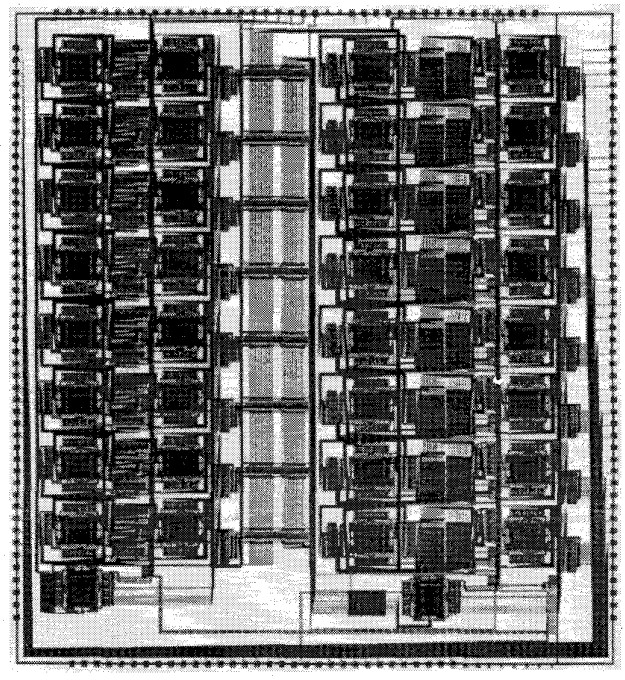


Fig. 17. Two dimensional DCT/IDCT chip. Chip measures 24550λ $\times$ 27094λ. Area is 240 mm^2.

times. This scheme is particularly relevant to the clock signal distribution.

The "rst" control signal is to be distributed to all those modules that are clocked as they need to be reset between blocks. The "fwd" signal which determines whether the forward or inverse DCT is computed is routed to all the multiplexers. Those modules that require these control signals also need the complement (which is generated at the local buffer). It is not necessary to route the complement of the control signals on a global chip scale. Routing of these control signals to each bit slice is built into the sub-module design.

H. 2-D DCT/IDCT Chip

Using all the cells—1-D, 2-D, and CSA—already described, the entire chip is assembled. The floor plan of the chip is shown in Fig. 16. In the floorplan, one can identify the various cells that have been mentioned earlier in this description. The physical layout of the 2-D DCT/IDCT chip is shown in Fig. 17. The chip-statistics are tabulated in Table V.

V. Comparisons

Some designs published in the literatures are compared to our design. As can be seen in Table VI, the designs in [5] and [8] are for low-bit rate CODEC's that operate at about 15 MHz. Both designs use the butterfly architecture resulting in little flexibility in transform size N. Transposition is also required, leading to higher latency. Both the designs are based on the distributed arithmetic implementation. The DCT/IDCT chip in [6] uses silicon compiler to design the whole system for implementation in 0.8 μ triple metal technology. In this design, a booth multiplier is used instead of distributed arithmetic. Transposition is employed even here. This chip can operate at 50 MHz, but the bit throughput rate is unknown. Table VI outlines the architectures of these four designs.

TABLE V
2D-DCT/IDCT Chip Statistics

Technology	1.2μ CMOS N-well
Die Dimensions	24550λ x 27094λ
Chip Area	240 mm^2
# of Transistors	320,000
Speed	50 MHz
Data rate	50 MSamples/s
Latency	72 cycles

TABLE VI
Architectural Comparisons

	Sun-Chen [5]	Fujiwara et al [8]	Miyazaki et al [6]	Srinivasan-Liu
Function	16 x 16 DCT	8 x 8 DCT/IDCT	8 x 8 DCT/IDCT	8 x 8 DCT/IDCT
Technology	2.0μ	1.0μ	0.8μ triple metal	1.2μ double-metal
Size	8.3 x 8.1 mm^2	10.7 x 10.2 mm^2	12.8 x 12.6 mm^2	14.7 x 16.2 mm^2
Transistors	73,000	156,000	180,000	320,000
Pads	25 active pads	68 PLCC pkg.	72 PGA	176 (Active pads ~38)
Max. Speed	15.1 MHz	15 MHz	50 MHz	50 MHz
Structure	Butterfly (irreg)	Irregular	Irregular	Regular (modular)

In comparison, our design and implementation of the 2-D DCT/IDCT chip in 1.2 μ CMOS technology results in a clocking rate of 50 MHz, corresponding to a data throughput rate of 50 MSamples/s. The modularity of the architecture tremendously simplifies the design/verification effort and the scalability of the algorithm gives us the flexibility in transform size N. In fact a 16 $\times$ 16 DCT/IDCT system can be designed and implemented by simply extending the 1-D/2-D filter banks to 16 channel modules and reprogramming the multiplier (ROM) coefficients. It may seem that our design has a lot of pins, but this is to aid in debugging—the number of active

pins is only 38, and a FIFO buffer can be placed at the output to reduce the pin count. Unlike the butterfly architecture (used in [5] and [8]) which needs global communication, our design does not have long nets running the width of the chip—as all channel modules are independent of each other. We would also like to point out that by using state-of-the-art technology (0.5–0.8 μ) and with minor tweaks to the architectural realization, much higher-speeds, better-accuracy, or significant area/transistor-count reduction can be achieved. Also, customization (i.e., reduction) in the number-of-bits allocation to the higher-order channels can result in further area minimization.

We would also like to reference Cucchi and Fratti's [19] work. Their 2-D DCT/IDCT chip has a throughput capability of 40 MHz. It has 57 000 transistors, occupying a 218-pad die area of 8.6 $\times$ 8.6 mm^2 fabricated in 1 μ technology.

VI. Conclusion

In this paper, we have presented a VLSI implementation of a high-performance high-speed 2-D DCT/IDCT chip. It is a full-custom implementation employing a highly modular and hierarchical design strategy. Distributed arithmetic is used for fast and compact multipliers. Nonoverlapping, two-phase clocking scheme leads to faster and more compact layout of the kernel. Architectural simulations are conducted for choosing system parameters that ensure adequate accuracy while minimizing chip area. The 2-D DCT/IDCT chip dimensions are 24550λ $\times$ 27094λ and its area is 240 mm^2 based on 1.2 μ technology. The pin-count is 176, and the chip has over 320 000 transistors. Timing simulations performed using Spice indicate a clock frequency of 50 MHz, corresponding to a data throughput rate of 50 MSamples/s. We have shown that VLSI design based on the class of time-recursive algorithms and architectures can easily meet the high-speed requirements. In comparison to the existing designs, our approach offers many advantages that can be further explored for even higher performance.

References

[1] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, and Applications*. New York; Academic, 1990.

[2] G. K. Wallace, "The JPEG still picture compression standard," *Commun. ACM*, vol. 34, pp. 31–44, Apr. 1991.

[3] F. Kretz and D. Nasse, "Digital television: Transmission and coding," *Proc. IEEE*, vol. 73, pp. 575–591, Apr. 1985.

[4] G. Tonge, "Image processing for higher definition television," *IEEE Trans. Circuits Syst.*, vol. CAS-34, pp. 1385–1398, November 1987.

[5] M.-T. Sun, T.-C. Chen, and A. M. Gottlieb, "VLSI implementation of a 16 $\times$ 16 discrete cosine transform," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 610–617, April 1989.

[6] T. Miyazaki, T. Nishitani, M. Edahiro, I. Ono, and K. Mitsuhashi, "DCT/IDCT processor for HDTV developed with DSP silicon compiler," *J. VLSI Signal Processing*, no. 5, pp. 151–158, 1993.

[7] M. Vetterli and A. Ligtenberg, "A discrete Fourier-cosine transform chip," *IEEE J. Select. Areas Commun.*, vol. SAC-4, pp. 49–61, Jan. 1986.

[8] H. Fujiwara, M. Liou, and M. Sun, "An all-ASIC implementation of a low bit-rate video codec," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, pp. 123–134, June 1992.

[9] D. Slawecki and W. Li, "DCT/IDCT processor design for high-data rate image coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, pp. 135–146, June 1992.

[10] K. J. R. Liu and C. T. Chiu, "Unified parallel lattice structures for time-recursive Discrete Cosine/Sine/Hartley transforms," *IEEE Trans. Signal Processing*, vol. 41, pp. 1357–1377, Mar. 1993.

[11] C. T. Chiu and K. J. R. Liu, "Real-time parallel and fully pipelined two-dimensional DCT lattice structures with applications to HDTV systems," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, pp. 25–37, Mar. 1992.

[12] K. J. R. Liu, C. T. Chiu, R. K. Kologotla, and J. F. JaJa, "Optimal unified architectures for the real-time computation of time-recursive discrete sinusoidal transforms," Tech. Res. Rep., ISR, University of Maryland at College Park, no. 66, 1992.

[13] S. A. White, "Application of distributed arithmetic to digital signal processing: a tutorial review," *IEEE ASSP Mag.*, pp. 4–19, July 1989.

[14] G. Ma and F. J. Taylor, "Multiplier policies for digital signal processing," *IEEE ASSP Mag.*, pp. 6–20, Jan. 1990.

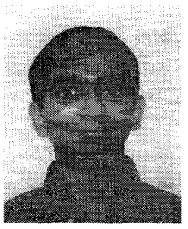
[15] C. Stearns and P. Ang, "Yet another multiplier architecture," in *IEEE Custom Integrated Circuits Conf.*, no. 24, pp. 6.1–6.4, 1990.

[16] R. N. Mayo, M. H. Arnold, W. S. Scott, D. Stark, and G. T. Hamachi, *DECWRL/Livermore Magic Release*. DEC and WRL, Res. Rep. 90/7, 1990.

[17] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design, A Systems Perspective*. Reading, MA: Addison-Wesley, 1988.

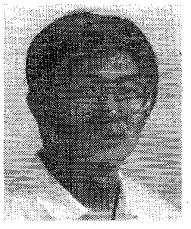
[18] L. A. Glasser and D. W. Dobberpuhl, *The Design and Analysis of VLSI Circuits*. Reading, MA: Addison-Wesley, 1985.

[19] S. Cucchi and M. Fratti, "A novel architecture for VLSI implementation of the 2-D DCT/IDCT," in *ICASSP*, pp. V-693–V-696, 1992.



Vishnu Srinivasan (M'95) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Madras, India, in 1991. He received the M.S. degree in 1993, also in electrical engineering, from The University of Maryland, College Park.

He was a Research Assistant at the Institute for Systems Research at UMCP from 1992 through 1993. He is currently employed at Crystal Semiconductor Corporation, Austin, TX, as a Design Engineer in the echo canceller group. His technical interests include VLSI architectures, algorithms and implementation, video compression, real-time signal processing, and mixed-signal integrated circuit design.



K. J. Ray Liu (S'86–M'86–SM'93) received the B.S. degree from the National Taiwan University in 1983, and the Ph.D. degree from the University of California, Los Angeles, in 1990, both in electrical engineering.

Since 1990 he has been with Electrical Engineering Department and Institute for Systems Research of University of Maryland at College Park, where he is an Associate Professor. His research interests span all aspects of high performance computational signal processing including parallel and distributed processing, fast algorithm, VLSI, and concurrent architecture, with application to image/video, radar/sonar, communications, and medical and biomedical technology.

Dr. Liu was the recipient of the National Science Foundation Young Investigator Award in 1994, and the IEEE Signal Processing Society's 1993 Senior Award. He received the George Corcoran Award for outstanding contributions to electrical engineering education at the University of Maryland. Dr. Liu is an Associate Editor of IEEE TRANSACTIONS ON SIGNAL PROCESSING and is a member of Design and Implementation of Signal Processing Systems Technical Committee of IEEE Signal Processing Society.