4.6. Discrete cosine transform

The two-dimensional discrete cosine transform (2-D DCT) is one of the most effective method in image data compression. In this sub-chapter an inner product algorithm for the 8x8 2-D DCT implementation is presented. The proposed direct 2-D inner product algorithm exploits redundancies down to the bit level, and results in minimal hardware complexity. The basic algorithm separates the computation to 8 subtransform, according to the different cosine function values. Utilizing the odd-even property of the DCT, every transformed coefficients are expressed as a 4 point size inner product operation. The inner product processors are realized by an efficient distributed vector multiplication arrangement. All of the numerical parameters are built in into the inner product processors, so the arithmetic complexity is partly transformed to the internal topology of the units. The selected globally parallel, locally serial implementation style is featured by basic serial processing elements and low communication cost. It is ideal for FPGA implementation, where the available chip area is a priori partitioned between logical and routing resources. The fully concurrent bit-serial pipeline architecture needs less than 1000 arithmetic primitives. Assuming 30MHz bitclock rate in the Xilinx FPGA, the available throughput is 1 million 2-D 8x8 DCT transform/sec.

4.6.1 The DCT in image coding

Transform coding has found many applications in digital signal processing, mainly in image compression. Among the possible different transforms the discrete cosine transform (DCT) is the most widely used, due to its good energy compaction property. For highly correlated signals, especially for the first order Markov process the DCT approaches the statistically optimal Karhunen-Loeve transform (KLT). The importance of the two-dimensional DCT is emphasized by the new standards also.

Since the introduction of the transform [Ahm74], many algorithms and VLSI architectures have been proposed to improve the computation speed and reduce the hardware complexity [Nar78], [Lee84], [Kam82]. These fast algorithms can be classified into four categories, according to [Liu93], as: matrix factorisation, recursive computation, indirect computation and systolic structure implementation. The matrix factorisation algorithms decompose the DCT into multiplications of many sparse matrices, so the

necessary arithmetic operations are dramatically reduced. The recursive computations use the lower order DCT output results to calculate the higher order ones. It should make clear that the recursive property of this algorithms are not refers to time recursive operation, it simply express the calculation method. Indirect algorithms are usually apply the existing fast algorithms of the discrete Fourier transform (DFT) or the Walsh-Hadamard transform. Systolic structure implementations are similar to systolic architectures of other transforms also. These arrangements are usually better for VLSI implementation than the above mentioned recursive algorithms, because they need no global communications.

As the DCT is orthogonal, the 2-D transform is usually accomplished by two separated 1-D DCT, first along the rows, after along the columns. This conventional method requires 2N 1-D DCT's for the computation of the 2-D DCT. For parallel hardware implementation of this method, a complicated matrix transposition architecture as well as 2N 1-D DCT modules are required. Thus for more efficient computation or parallel implementation of the 2-D DCT, the algorithms that work directly on the 2-D data set have been introduced [Kam82],[Duh90],[Cho91]. These algorithms are based on matrix factorisation or indirect computation using the polynomial transform FFT, and all work in the usual block processing mode. Although they reduce significantly the number of multiplications relative to the conventional method, their arithmetic complexity prohibits the direct hardware implementation.

The new algorithm proposed in this paper is based on the inner product calculation of the 2-D transform coefficients. To hold the arithmetic complexity of the parallel architecture at a reasonable level, serial arithmetic is used to perform the computation. A fully pipe-lined structure based on an efficient method for inner product calculations [Feh93] allows high throughput capability. The full hardware structure is built up from basic arithmetic primitives and doesn't make use of multipliers. The algorithm works in the usual block processing mode. The computation needs all of the input data samples and generates the output results at the same time. This can be classified as a parallel input, parallel output (PIPO) algorithm.

4.6.2 The inner product algorithm of the 2D DCT

The NxN size 2-D DCT is defined as

$$[X] = [C]\ [x]\ [C]^T \tag{4.6.1}$$

where [x], [X] and [C] are the input, output and transform matrices with size of NxN.

Definition of C is given in [Ahm74]. Most of the implementations of the 2-D DCTs are based on this transposition method [Liu93]. These architectures in case of parallel hardware implementation needs 2N 1-D DCT modules and a complicated matrix transposition unit.

Substituting the transform matrix elements, the immediate computation of [X] output data matrix from the [x] input is given by Eq. 4.6.2.,

$$X(u,v) = \frac{2}{N} c(u)\, c(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} x(i,j) \cos\frac{(2i+1)u\pi}{2N} \cos\frac{(2j+1)v\pi}{2N} \qquad \textbf{(4.6.2)}$$

$$u,v,i,j = 0,1,\ldots,N-1; \qquad c(u), c(v) = \begin{cases} \sqrt{(2)} & u,v = 0 \\ 1 & u,v \neq 0 \end{cases}$$

where x(i,j) and X(u,v) represent the (NxN) arrays in the data and DCT domain. Eq. 4.6.2. also can be written in matrix form.

$$X_L = [F]\, x_L \qquad \textbf{(4.6.3)}$$

where X_L and x_L are linear sequences rearranged in lexicographic order of 2-D data and transformed array x_L={x(0,0), x(0,1), ... x(0,N-1), x(1,0), ..., x(N-1,N-1)}, X_L={X(0,0), X(0,1), ... X(0,N-1), X(1,0), ..., X(N-1,N-1)}. The [F] (N^2,N^2) matrix is the forward 2-D transform matrix, its elements are determined by Eq. 4.6.2. as shown in Eq. 4.6.4.

$$F(m,n) = \frac{2}{N} c(Q(m,N))\, c(R(m,N)) \cos\frac{(2Q(n,N)+1)Q(m,N)\pi}{2N} \cos\frac{(2R(n,N)+1)R(m,N)\pi}{2N} \qquad \textbf{(4.6.4)}$$

where m=0,1,...,N^2-1, n=0,1,...,N^2-1, and Q(n,N), Q(m,N), R(n,N) and R(m,N) are the quotient and remainder of the integer division operation of m/N, n/N, and stand for the original i,u row and j,v column indices, respectively. According to Eq. 4.6.4. the components of X_L can be calculated as an NxN size inner product operation. Analysis of the 2-D transform matrix shows high degree of symmetry and redundancy between the matrix elements. Application of the basic trigonometric identity of Eq. 4.6.5. leads more efficient computation of the inner products.

$$\cos a \cdot \cos b = \frac{1}{2}\{\cos(a+b) + \cos(a-b)\} \qquad \textbf{(4.6.5)}$$

The elements of the 2-D transformation matrix [F] can be written as the sum of two term using Eq. 4.6.5. The same conversion can be applied to the c(Q(m,N))c(R(m,N)) multiplier constant values also. When Q(m,N) and/or R(m,N) are equal to zero, then the c constants take the value of $\sqrt{2}/2$. This value is equal with $\cos((N/2)\pi/2N)$, so these elements of [F] has the same form as the others.

For the F(m,n) matrix elements in a given row, the Q(m,N) and R(m,N) are constant values, even or odd numbers. so the multiplier factors of $\pi/2N$ can have only even or odd values in the inner product. Exploiting the symmetry and periodicity of the cosine function the N^2 size inner products have only N/2 (or less) numerically different multiplier factors.

The transformation matrix [F] is separable to the sum and difference matrices, according to Eq. 4.6.5.

$$X_L = ([F^+] + [F^-])\,x_L = [F^+]\,x_L + [F^-]\,x_L \qquad \textbf{(4.6.6)}$$

These separation can be continued until all the matrices contains only one Cx numerical constant value.

$$X_L = ([F0^+] + [F1^+] + \ldots + [F(N-1)^+])\,x_L + ([F0^-] + [F1^-] + \ldots + [F(N-1)^-])\,x_L \qquad \textbf{(4.6.7)}$$

In the [Fx+] and [Fx-] matrices, for a given m,n index pair, m,n=0,1,..N^2-1, there is maximum one nonzero element, so the matrices can be merged.

$$X_L = ([F0] + [F1] + \ldots + [F(N-1)])\;x_L = \sum_{k=0}^{N-1} [Fk]\;x_L \qquad \textbf{(4.6.8)}$$

Eq. 4.6.8. already defines the algorithm, but a more better form available, using the following notation for the multiplier factors.

$$[Fx] = Cx \cdot [Tx], \quad where\ Cx = \cos\frac{x\pi}{2N}\ ;\ x = 0,1,..N-1 \qquad \textbf{(4.6.9)}$$

Removal of the Cx multipliers from the [Fx] gives the [Tx] matrices, x=0,1,..N-1. These matrices have only +1, -1, and 0 elements. The computation of the 2-D DCT consist of two phases, as shown in Eq. 4.6.10. and Fig. 4.6.1. At first, in the pre-addition phase, the lexicographical input data is arranged according to the pattern of the nonzero elements of the [Tx] matrices. The pre-additions need only simple butterfly stages on the x_L input data vector components, and provides the necessary data combinations for the final, inner product processors.

$$X_L^{EVEN} = \sum_{k=0}^{N/2-1} C(2k)\ \{\ [T(2k)]\ x_L\ \}$$

$$X_L^{ODD} = \sum_{k=0}^{N/2-1} C(2k+1)\ \{\ [T(2k+1)]\ x_L\ \} \qquad \textbf{(4.6.10)}$$

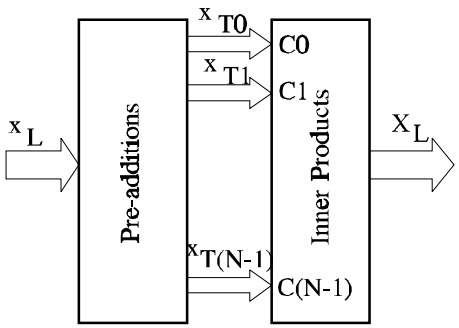

Figure 4.6.1: The computational steps of the algorithm

The first two level of the butterfly stages are simple and regular and common for all [Tx] x=0,1,...N-1 matrices. The $[L_N]$ NxN size butterfly matrix can be expressed in terms of the identity matrix $[I_{N/2}]$ and the opposite identity matrix $[I^*_{N/2}]$.

$$[L_N] = \begin{bmatrix} [I_{N/2}] & [I^*_{N/2}] \\ [I^*_{N/2}] & -[I_{N/2}] \end{bmatrix} \qquad \textbf{(4.6.11)}$$

If the temporary data vectors between the levels of butterflies are indexed by the letters of B, C, D, E, F, then

$$\begin{gathered} X_B = [A]\, x_L \quad ; \quad X_C = [B]\, X_B \quad ; \quad X_D = [C]\, X_C \\ [A] = [L_{64}] \\ [B] = diag \ \{ \ [L_8],[L_8],.....[L_8],[L_8] \ \} \\ [C] = diag \ \{ \ [L_4],[L_4],[L_4],.....[L_4],[L_4],[L_4],[L_4] \ \} \end{gathered} \qquad \textbf{(4.6.12)}$$

The signal flowgraph of the pre-addition phase for the following levels are shown in Fig. 4.6.2. and Fig. 4.6.3. for the even and the odd [Tx] matrices, respectively. On the figures the subtractions are not designated explicitly, but the lower horizontal line carries always the subtrahend. The flowgraphs show the appropriate connection points at the C and D level for these unique and the calculated $X_{Tx(i)}$ output signals of the pre-addition phase. It can be seen that many of the rows of the [Tx] matrices are share the same pattern, and the number of the different combinations are only N^2. The execution of the full pre-addition flowgraph for the 8x8 transformation needs 368 addition operation. The Tx(i) i=0,1,...N-1 notation at the output points of the pre-addition butterflies indicates the necessary inner product operations. To get the i-th transform domain data every Tx(i) combination of the input signals should be multiplied by the appropriate Cx value in the inner product processors, as given in Eq. 4.6.10. The architecture of the necessary inner product processors are discussed in the following section.

4.6.3 Internal structure of the inner product processors

Parallel hardware implementation of the direct 2-D transformation needs efficient arithmetic elements. To hold the hardware complexity at a reasonable level, the globally parallel, locally serial implementation style should be selected. This method has two

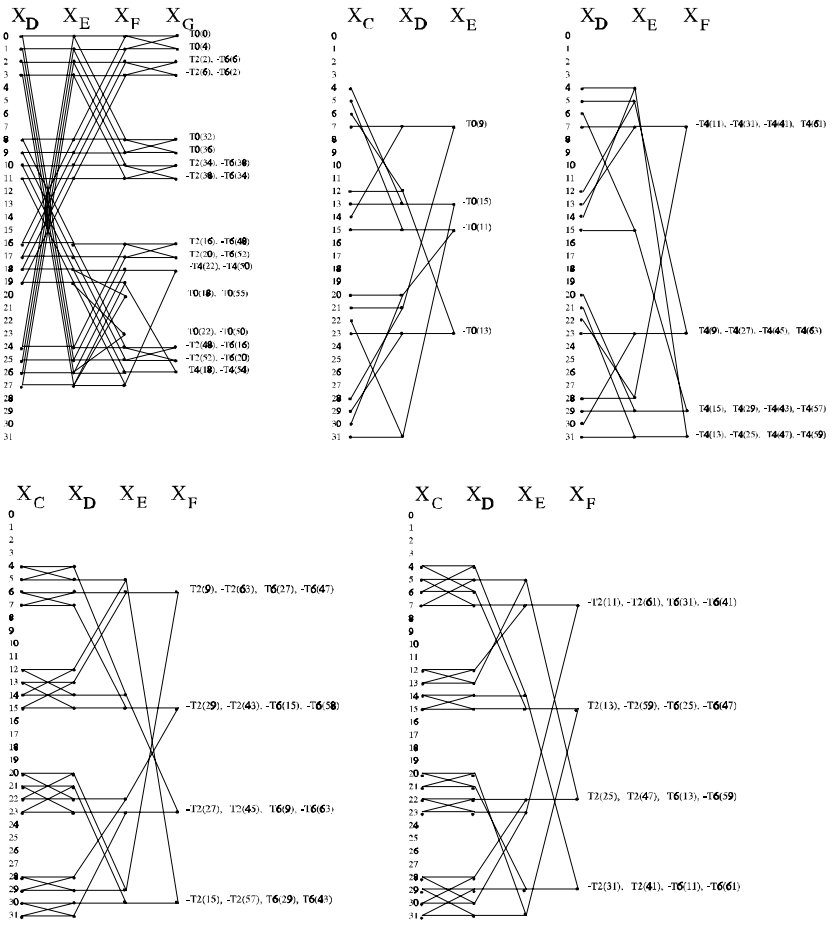

Figure 4.6.2: The signal flowgraphs of the even T matrices

advantage. The first is the simplicity of the basic serial arithmetic units, the second is the

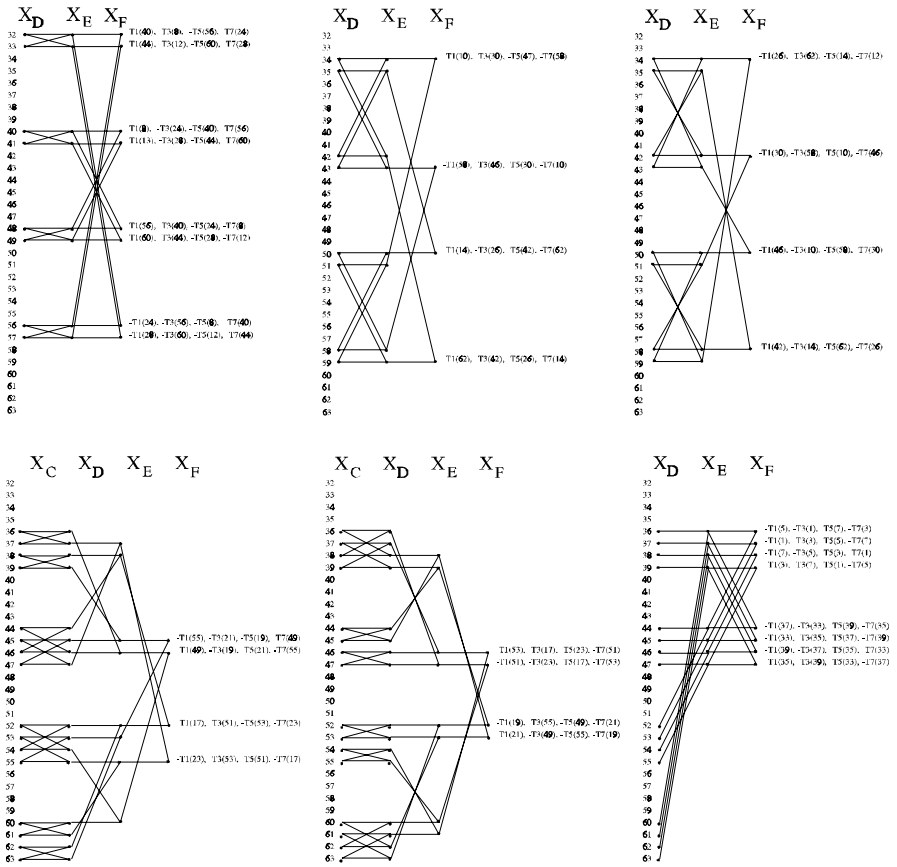

Figure 4.6.3: The signal flowgraphs of the odd T matrices

very low communication cost. Both of these are important to reduce the necessary resources.

There are different methods to realize the inner product processors. The most widely used arrangement in the DCT processors is the memory look-up table based vector multiplier [Rao90]. This processors contains memories to store the precomputed partial product sums and use parallel arithmetic units internally for the accumulation. The are some drawbacks of this arrangement. In some technology memory modules are difficult to realize (for example in FPGA devices), and the parallel arithmetic elements needs long basic execution cycles.

Recently a new efficient inner product processor arrangement have been proposed for the realization of fixed coefficient vector multiplications by serial arithmetic [Feh93]. This arrangement contains basic arithmetic primitives only, all of the operations are realized by pipe-lined serial adders, subtractors. Efficiency is achieved by using the Canonical Signed Digit code (CSD) of the fixed coefficients of the operations and by exploiting the available digit pattern coincidences between the coefficients CSD codes. The amount of reduction is depend on the size of the inner product operation, the actual numerical values and the CSD codes of the coefficients. Typically the method provide more than 30% reduction in the number of arithmetic elements and more than 70% in the number of the delay elements [Feh93].

The maximum size of the inner products necessary to calculate the X_L transformed domain coefficients of the 8x8 2-D DCT are 4. The multiplication factors are the odd or even elements of the Cx, x=0,1,...N-1 series, where Cx is defined by Eq. 4.6.9. The numerical values and the CSD codes of the coefficients of the 2-D DCT are given in Eq. 4.6.12., and internal structure of the inner product processors are shown in Fig. 4.6.4.

The inner product processors assume 12 bit number representations of the Cx coefficients. Simulation results have shown, that it is satisfies the necessary computation accuracy.

4.6.4 FPGA implementation of the 2D DCT algorithm

Logic complexity of the available Field Programmable Gate Arrays (FPGAs) are relatively low for computation intensive DSP applications. Implementation of complex algorithms, such as the 2-D DCT on FPGA elements needs special precautions. The limitations of the available logic resources requires direct mapping of the arithmetic primitives to the FPGA basic logic cells. In case of serial arithmetic the optimal mapping is the one to one correspondence between the signal flow graph components and the

$$
\begin{array}{lllllllllllllllll}
C0 & = & 1_{10} & = & 1 & . & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
C1 & = & 0.9808_{10} & = & 1 & . & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 1 & 0 \\
C2 & = & 0.9239_{10} & = & 1 & . & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
C3 & = & 0.8315_{10} & = & 1 & . & 0 & -1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\
C4 & = & 0.7071_{10} & = & 1 & . & 0 & -1 & 0 & -1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
C5 & = & 0.5556_{10} & = & 0 & . & 1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & -1 \\
C6 & = & 0.3827_{10} & = & 0 & . & 1 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
C7 & = & 0.1951_{10} & = & 0 & . & 0 & 1 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0
\end{array}
\tag{4.6.13}
$$

hardware logic resources. Using SRAM based FPGAs, the relatively large size of the basic logic cells are ideal to implement the pipe-lined serial arithmetic primitives. All the necessary logic are contained in one cell. This provides single level logic delays, high utilization of internal combinational and storage resources and minimum number of external connection points, reducing the routing complexity. As the routing resources of the FPGAs are designed for general usage of the high fan-in logic cells, in case of serial arithmetic most of the routing channels remains available to realize the complex topology of the signal flow graph of the given algorithm.

The hardware cost of the 2-D DCT is consist of two part. The first is the pre-addition module, the second is the inner product processor. The cost of the pre-addition module is proportional with the size of the transformation and is given by Eq. 4.6.14.

$$C_{PA} \cong N^2 \cdot \log_2 N^2 \tag{4.6.14}$$

The cost of the inner product processors for the selected 12 bit precision can be calculated from Fig. 4.6.4. The processors of the X_L transformed domain data components are represents different complexity. The cost of all modules are summarized and listed in Table 1. It shows, that the total number of arithmetic primitives is less than 1000.

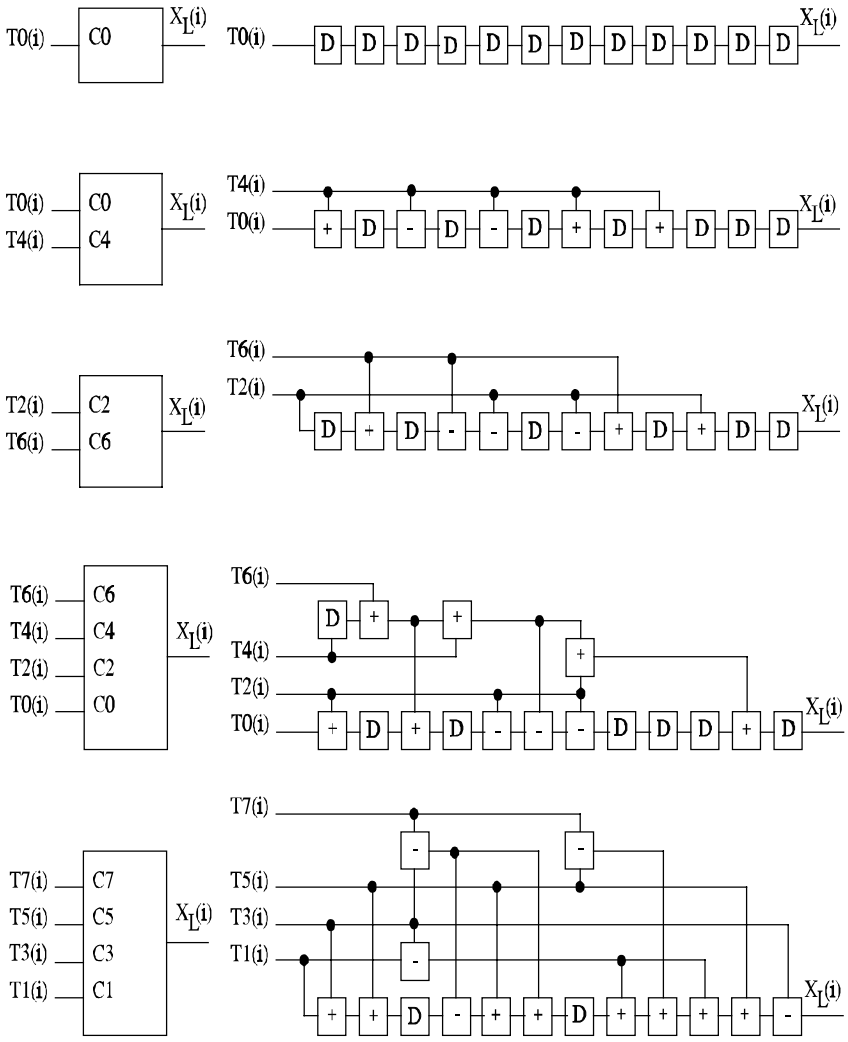

Figure 4.6.4: The signal flow graph of the 8x8 2-D DCT inner product processors

	Pre-add.	C1-C7 IPPs	C0 delay	C0,C4 IPPs	C2,C6 IPPs	C0-C6 IPPs	Total
+/-	368	416	0	20	48	144	996
Delay	0	64	48	28	48	112	300

The execution time of the computation using the conventional LSB first computation scheme is $B_x+B_c+\log_2 N^2$ bit clock cycle, where B_x and B_c are the number of bits in the input data and Cx coefficient representation. The available bit clock rate with the Xilinx 4000 series FPGAs is about 30-40 MHz. Assuming the usual B_x=9 and B_c=12 bit values, one transformation needs less than 30 clock cycle, so the available thoughput is 1 million transformation/sec.

Bibliography

[Ahm74] N. Ahmed, T. Natarajan, K. R. Rao," Discrete cosine transform," IEEE Trans. Computers, Vol. C-23, pp. 90-93, Jan. 1974.

[Ait91] D. Ait-Boudaoud, M. K. Ibrahim,"Efficient Implementation of Bit-Level Signal Processors", Proc. of IEEE ISCAS, 1991

[Bla85] R. E. Blahut, "Fast Algorithms for Digital Signal Processing,"Addison-Wesley, 1985

[Bur77] C. S. Burrus,"Digital Filter Structures Described by Distributed Arithmetic," IEEE Trans. on Circuits and Systems, Vol. CAS-24, No.12, pp. 674-680. Dec. 1977.

[Cho91] N. I. Cho, S. U. Lee,"Fast algorithms and implementation of 2-D discrete cosine transform," IEEE Trans. Circuits and Systems, Vol. 38. pp. 297-305, March, 1991.

[Den85] P. Denyer, D. Renshaw, VLSI signal processing: A bit-serial Approach. Addison-Wesley, 1985

[Diet90] C. B. Dietrich, et. al.,"A Complete Audio Decoder IC for U.S. Television Stereo Using Serial Arithmetic", Proc. of IEEE CICC 1990

[Duh90] P. Duhamel, C. Guillemot,"Polynomial transform computation of 2-D DCT," in Proc. of IEEE ICASSP 1990. pp. 1515-1518. Apr. 1990.

[Feh91] B. Fehér,"Field-programmable gate array implementation of the recursive Walsh-Hadamard transformation," presented at the Int. Conf. on DSP Appl. and Techn. Berlin, Germany, Oct. 28-31, 1991.

[Feh92a] B., Feher, "Resonator Based Digital Filters Using Field Programmable Gate Array Elements, Proc. of IEEE ASIC, 1992.

[Feh92b] B., Fehér, T., Lázár, "Digital FIR Filter Structure with XILINX Programmable Logic Circuit", (in Hungarian) Mérés- és Automatika, Vol. 40. No. 5. pp. 294-298. Dec. 1992.

[Feh93a] B. Fehér," Efficient Synthesis of Distributed Vector Multipliers," Microprocessors and Microprogramming, Vol. 38. pp. 345-350, Sept, 1993

[Feh93b] B., Fehér, "Coefficient Dependent Synthesis of FIR Digital Filters", Proc. of the 4th Mideuropean Conference on Custom/Application Specific Integrated Circuits, 1993

[Feh94] B., Feher et. al. "Parallel Implementation of teh Statistic Feature method" Proceedings of uP94.The 8th Symposium on Microcomputer and Microprocessor Applications Budapest, October, 1994. pp.211-219.

[Hou87] H. S. Hou, "A fast recursive algorithms for computing the discrete cosine transform," IEEE Trans. Acoust. Speech, Signal Processing, Vol. ASSP-35, pp. 1455-1461, Oct. 1987.

[Hsi90] Hsieh, Hung-Cheng, et. al.,"Third-generation architecture boosts speed and density of field programmable gate arrays," IEEE Conference on Custom Integrated Circuits, 1990

[Hun90] H., Hung-Cheng, et. al.,"Third-generation architecture boosts speed and density of field programmable gate arrays," Proc. of IEEE CICC, 1990

[Kam82] F. A. Kamangar, K. R. Rao, "Fast algorithms for the 2-D discrete cosine transform," IEEE Trans. Computers, Vol. C-31, pp. 899-906, Sept, 1982.

[Lee84] B. G. Lee, "A new algorithm to compute the discrete cosine transform," IEEE Trans. Acoust. Speech, Signal Processing, Vol. ASSP-32, pp. 1243-1245, Dec. 1984.

[Liu93] K. J. R. Liu, C. T. Chiu,"Unified parallel structures for time-recursive discrete cosine/sine/Hartley transform," IEEE Trans. on Signal Processing, Vol. 41. No. 3. pp. 1357-1377. March 1993.

[Lyo76] R. F. Lyon,"Two's complement pipeline multipliers," IEEE Trans. Communications, pp. 418-425, Apr. 1976.

[Mas91] D. L. Maskell, "Hardware Implementations for Digital Filters with Low Coefficient Sensitivity," IEEE Int. Symp. on CAS, 1991. vol. 1, pp.504-507.

[Nar78] N. J. Narashimha, A. M. Peterson, "On the computation of the discrete cosine transform," IEEE Trans. Commun. Vol. COM-26, pp. 934-936, June 1978.

[Pec86] G. Péceli,"A common structure for recursive discrete trans forms," IEEE Trans. Circuits Syst., vol. CAS-33 no. 10. pp. 1035-1036, Oct. 1986.

[Pec89] G. Péceli,"Resonator-Based Digital Filters," IEEE Trans. Circuits Syst., vol. CAS-36, no. 1. pp. 156-159, Jan 1989.

[Pec90] G. Péceli, B. Fehér,"Digital filters based on recursive Walsh-Hadamard transformation," IEEE Trans. Circuits Syst., vol. CAS-37, no. 1. pp. 150-152, Jan. 1990.

[Pel74] A. Peled, B. Liu,"A new hardware realization of digital filters," IEEE Trans. Acoustics, Speech, Signal Processing, vol. ASSP-22, no. 6, pp. 456-462. Dec. 1974.

[Rao90] K. R. Rao, P. Yip,"Discrete cosine transform," Chapter 5., Academic Press, San Diego, 1990.

[Rei60] G. W. Reitwiesner, "Binary Arithmetic. Advances in Computers," vol. 1, pp. 232-308. Academic Press, 1960.

[Vet85] Vetterli, M: Fast 2-D Discrete Cosine Transform. Proc. of IEEE ICASSP, 1985.

[Xil91] Xilinx XNF Netlist Specification, Xilinx Inc., Version 4.00a-10/18/1991.