GENERALIZED APPROACH FOR THE REALIZATION OF DISCRETE COSINE TRANSFORM USING CYCLIC CONVOLUTIONS

Yuk-Hee CHAN and Wan-Chi SIU

Department of Electronic Engineering
Hong Kong Polytechnic

Abstract

A general solution is proposed to realize the discrete cosine transform of any length via cyclic convolutions in this paper. This algorithm is not optimal in minimizing any measure of computational complexity, but it involves some regular forms that are most suitable for the realization using technologies and structures which are well suited for doing convolutions, such as the distributed arithmetic and the systolic array. On the other hand, this algorithm is much more flexible than any available DCT algorithm as it can be applied to realize DCT/IDCT with any length.

Introduction

It is well known that one can use the chirp-z transform to realize a DFT with any length through convolutions. This algorithm is not optimal in minimizing any measure of computational complexity, but it is useful and efficient in the realization of DFT with variable length as it is much more flexible compared with other DFT algorithms for certain applications.

It is obvious that similar advantages can be obtained for the realization of the DCT if there exists a similar general solution to the DCT[1]. Unfortunately, one can not apply the chirp-z transform directly as that in the DFT case since the kernels of the two transforms are completely different. Possible solution can be achieved by firstly inverting the DCT into a DFT[2,3] and then applying the chirp-z transform directly. However, this approach involves complex number operations and seems a little bit circuitous. Hence, it is necessary for us to look for a more straight-forward solution which involves real operations solely.

The aim of this paper is to suggest such a general solution for the realization of the DCT and to propose a simple structure to ease the implementation of this general solution.

Basic Algorithm

The definition of the DCT on a sequence $\{y(i):i=0,1..N\text{-}1\}$ is given as

$$Y(k) = \sum_{i=0}^{N-1} y(i)\cos\left[\frac{\pi(2i+1)k}{2N}\right] \qquad \text{for } k=0,1..N\text{-}1 \quad (1)$$

In ref.[4], we have shown that by defining another sequence $\{x(i):i=0,1..N\text{-}1\}$ as

$$x(N\text{-}1) = y(N\text{-}1)$$

$$x(i) = y(i)\text{-}x(i+1) \qquad \text{for } i=0,1...N\text{-}2 \quad (2),$$

we have
$$Y(k) = \left\{2T(k) + x(0)\right\}\cos\frac{k\pi}{2N} \qquad \text{for } k=0,1...N\text{-}1 \quad (3)$$

where
$$T(k) = \sum_{i=1}^{N-1} x(i)\cos\frac{ik\pi}{N} \qquad \text{for } k=0,1...N\text{-}1 \quad (4)$$

Instead of realizing eqn. 4 directly, we can realize another sequence $\{X(k)\}$ defined below and then compute $\{T(k)\}$ from $\{X(k)\}$ indirectly. First of all, we have to define the sequence $\{X(k)\}$:

$$X(k) = \sum_{i=1}^{N-1} x(i)\cos\frac{\pi}{2N}(k^2-2ik) \qquad \text{for any integer } k \quad (5)$$

Obviously, we have

$$X(-k) = X(2N-k) = \sum_{i=1}^{N-1} x(i)\cos\frac{\pi}{2N}(k^2+2ik) \qquad \text{for any integer } k \quad (6)$$

Then, $\{T(k)\}$ can be obtained by

$$T(k) = \frac{1}{2}\left(X(k)+X(2N-k)\right)\sec\frac{k^2\pi}{2N} \qquad \text{for } k=0,1...N\text{-}1 \quad (7)$$

In particular, $\{X(k)\}$ can be rewritten as

$$X(k) = \sum_{i=1}^{N-1} x(i)\cos\left[(i-k)^2-i^2\right]\frac{\pi}{2N}$$

$$= \sum_{i=1}^{N-1} h(i)\cos\left[\frac{(i-k)^2\pi}{2N}\right] + \sum_{i=1}^{N-1} g(i)\sin\left[\frac{(i-k)^2\pi}{2N}\right] \qquad \text{for any integer } k \quad (8)$$

where
$$h(i) = x(i)\cos\frac{i^2\pi}{2N} \qquad \text{for } i=1,2...N\text{-}1 \quad (9)$$

$$g(i) = x(i)\sin\frac{i^2\pi}{2N} \qquad \text{for } i=1,2...N\text{-}1 \quad (10)$$

To obtain the DCT of the sequence $\{y(i)\}$, we have to obtain the sequence $\{T(k):k=0,1..N\text{-}1\}$. That means we have to compute $\{X(k):k=0,1...N\text{-}1,N+1,N+2...2N\}$. However, as $T(0)=\sum_{i=1}^{N-1} x(i)$, all we need to do is to calculate $\{X(k):k=1,2..N\text{-}1,N+1,N+2..2N\text{-}1\}$. In other words, from eqn. 8, we have to compute the following two liner convolutions:

$$G(k) = \sum_{i=1}^{N-1} g(i)\sin\left[\frac{\pi}{2N}(i-k)^2\right] \qquad \text{for } k\in\{1,2...2N\text{-}1\}\backslash\{N\} \quad (11)$$

$$H(k) = \sum_{i=1}^{N-1} h(i)\cos\left[\frac{\pi}{2N}(i-k)^2\right] \qquad \text{for } k\in\{1,2...2N\text{-}1\}\backslash\{N\} \quad (12)$$

In such case, one can realize a DCT with any length N by using two N-length linear convolutions while the overheads are at most 4(N-1) multiplications.

0-7803-0946-4/93 $3.00 © 1993 IEEE

Further Deduction

Sometimes it is preferable to compute the DCT by cyclic convolutions instead of linear convolutions due to the efficiency of the realization especially when dedicated hardware structure can be used. In such case, we can make some modifications on eqn.7 such that we can make use of cyclic convolutions. Obviously, from eqns.7-12, we have

$$T(k) = \frac{1}{2}\left[H(k)+H(2N-k)+G(k)+G(2N-k)\right]\sec\left[\frac{k^2\pi}{2N}\right]$$

$$\text{for } k=0,1...N-1 \quad (13)$$

As $T(0) = \sum_{i=1}^{N-1} x(i)$, we can compute this item directly through simple addition. For other items of the sequence $\{T(k)\}$, we can first compute two sequences, $\{H(k)+H(2N-k)\}$ and $\{G(k)+G(2N-k)\}$. That is, for the sequence $\{H(k)+H(2N-k)\}$, we compute

$$H(k)+H(2N-k) = \sum_{i=1}^{2N-1} h'(i)\cos\left[\frac{(i-k)^2\pi}{2N}\right]$$

$$\text{for } k=1,2...N-1 \quad (14)$$

where $\{h'(i):i=0,1..2N-1\}$ is defined as

$$h'(i) = \begin{cases} h(i) & \text{if } i=1,2,...N-1 \\ 0 & \text{if } i=N \\ h(2N-i) & \text{if } i=N+1,N+2,...2N-1 \end{cases} \quad (15)$$

Similarly, by defining another sequence $\{g'(i):i=0,1..2N-1\}$ as

$$g'(i) = \begin{cases} g(i) & \text{if } i=1,2,...N-1 \\ 0 & \text{if } i=N \\ g(2N-i) & \text{if } i=N+1,N+2,...2N-1 \end{cases} \quad (16),$$

we have

$$G(k)+G(2N-k) = \sum_{i=1}^{2N-1} g'(i)\sin\left[\frac{(i-k)^2\pi}{2N}\right]$$

$$\text{for } k=1,2...N-1 \quad (17)$$

Note that both $\{\cos\left[n^2\frac{\pi}{2N}\right]: n\in Z\}$ and $\{\sin\left[n^2\frac{\pi}{2N}\right]: n\in Z\}$ (where **Z** is the set of integers) are cyclic with a period of 2N, that means eqns. 14 and 17 are both in cyclic convolution form after appending $h'(0)=0$ and $g'(0)=0$ to sequences $\{h'(i)\}$ and $\{g'(i)\}$ respectively. Note that it is not necessary to compute the whole cyclic convolution as only N-1 items of $\{G(k)+G(2N-k)\}$ and $\{H(k)+H(2N-k)\}$ are required. On the other hand, this approach can save 2(N-1) additions in realizing eqn. 7 compared with the approach using linear convolutions.

Example

Let us use a 6-point DCT to clarify our proposal. Suppose the input sequence is given as $\{y(i):i=0,1..5\}$. Then, obviously, we have $Y(0) = \sum_{i=0}^{5} y(i)$. For the other DCT coefficients, we first compute the sequence $\{x(i)\}$ from $\{y(i)\}$ with eqn. 2:

$$\begin{aligned} x(5) &= y(5) \\ x(4) &= y(4)-y(5) \\ x(3) &= y(3)-y(4)+y(5) \\ x(2) &= y(2)-y(3)+y(4)-y(5) \\ x(1) &= y(1)-y(2)+y(3)-y(4)+y(5) \\ x(0) &= y(0)-y(1)+y(2)-y(3)+y(4)-y(5) \end{aligned}$$

Then, based on eqns. 9,10,14-17, we can realize the following two cyclic convolutions to get sequences $\{G(k)+G(12-k)\}$ and $\{H(k)+H(12-k)\}$:

$$\begin{bmatrix} H(1)+H(11) \\ H(2)+H(10) \\ H(3)+H(9) \\ H(4)+H(8) \\ H(5)+H(7) \end{bmatrix} =$$

$$\begin{bmatrix} c(1) & c(0) & c(1) & c(4) & c(9) & c(16) & c(1) & c(12) & c(1) & c(16) & c(9) & c(4) \\ c(4) & c(1) & c(0) & c(1) & c(4) & c(9) & c(16) & c(1) & c(12) & c(1) & c(16) & c(9) \\ c(9) & c(4) & c(1) & c(0) & c(1) & c(4) & c(9) & c(16) & c(1) & c(12) & c(1) & c(16) \\ c(16) & c(9) & c(4) & c(1) & c(0) & c(1) & c(4) & c(9) & c(16) & c(1) & c(12) & c(1) \\ c(1) & c(16) & c(9) & c(4) & c(1) & c(0) & c(1) & c(4) & c(9) & c(16) & c(1) & c(12) \end{bmatrix} \bullet$$

$$\begin{bmatrix} 0 & h(1) & h(2) & h(3) & h(4) & h(5) & 0 & h(5) & h(4) & h(3) & h(2) & h(1) \end{bmatrix}^T,$$

$$\begin{bmatrix} G(1)+G(11) \\ G(2)+G(10) \\ G(3)+G(9) \\ G(4)+G(8) \\ G(5)+G(7) \end{bmatrix} =$$

$$\begin{bmatrix} s(1) & s(0) & s(1) & s(4) & s(9) & s(16) & s(1) & s(12) & s(1) & s(16) & s(9) & s(4) \\ s(4) & s(1) & s(0) & s(1) & s(4) & s(9) & s(16) & s(1) & s(12) & s(1) & s(16) & s(9) \\ s(9) & s(4) & s(1) & s(0) & s(1) & s(4) & s(9) & s(16) & s(1) & s(12) & s(1) & s(16) \\ s(16) & s(9) & s(4) & s(1) & s(0) & s(1) & s(4) & s(9) & s(16) & s(1) & s(12) & s(1) \\ s(1) & s(16) & s(9) & s(4) & s(1) & s(0) & s(1) & s(4) & s(9) & s(16) & s(1) & s(12) \end{bmatrix} \bullet$$

$$\begin{bmatrix} 0 & g(1) & g(2) & g(3) & g(4) & g(5) & 0 & g(5) & g(4) & g(3) & g(2) & g(1) \end{bmatrix}^T$$

where $c(n) = \cos\left[n\frac{\pi}{12}\right]$, $s(n) = \sin\left[n\frac{\pi}{12}\right]$, $h(n) = x(n)c(n^2)$ and $g(n) = x(n)s(n^2)$. From eqn. 13, we have

$$\begin{bmatrix} T(1) \\ T(2) \\ T(3) \\ T(4) \\ T(5) \end{bmatrix} = \begin{bmatrix} 1/2c(1)\times(G(1)+H(1)+G(11)+H(11)) \\ 1/2c(4)\times(G(2)+H(2)+G(10)+H(10)) \\ 1/2c(9)\times(G(3)+H(3)+G(9)+H(9)) \\ 1/2c(16)\times(G(4)+H(4)+G(8)+H(8)) \\ 1/2c(25)\times(G(5)+H(5)+G(7)+H(7)) \end{bmatrix}$$

and finally we can obtain the other DCT coefficients through the realization of eqn. 3:

$$\begin{bmatrix} Y(1) \\ Y(2) \\ Y(3) \\ Y(4) \\ Y(5) \end{bmatrix} = \begin{bmatrix} (2T(1)+x(0))\times c(1) \\ (2T(2)+x(0))\times c(2) \\ (2T(3)+x(0))\times c(3) \\ (2T(4)+x(0))\times c(4) \\ (2T(5)+x(0))\times c(5) \end{bmatrix}$$

If N is small, the overhead of 4(N-1) multiplications will dominate the computational effort required and therefore the implementation of this algorithm may be inefficient. Nevertheless, when N gets larger, this overhead will become less and finally neglectable. In such case, the implementation of this algorithm can be very efficient as it mainly involves two cyclic convolutions only, which requires no complicated data management and routing algorithm.

Inverse DCT

The inverse Discrete Cosine Transform (IDCT) of data $\{Y(k):k=0,1...N-1\}$ is given by the following:

$$y(i) = \sum_{k=0}^{N-1} Y(k)\cos\left[\frac{\pi(2i+1)k}{2N}\right] \qquad \text{for } i=0,1...N-1 \quad (18)$$

Obviously, if one can rewrite eqn.18 such that it can be realized through the formulation in the form of eqn.4, eqn.18 can also be realized using convolutions. In fact, this can be readily achieved by defining another sequence:

$$y'(i) = y(i) + y(i-1) = \sum_{k=1}^{N-1}\left\{2\,Y(k)\cos\frac{k\pi}{2N}\right\}\cos\frac{ik\pi}{N} + 2Y(0)$$

$$\text{for } i=0,1...N-1 \quad (19)$$

Where y(-1) is defined to be equal to y(0).

We have $y'(0) = 2y(0)$ and therefore $\{y(i):i = 0,1...N\text{-}1\}$ can be obtained from $\{y'(i):i = 0,1...N\text{-}1\}$ through N-1 additions. Note that eqn.19 is in the form of eqn.4. Hence, by applying the same technique used in above sections, one can realize an IDCT with two linear convolutions or two cyclic convolutions.

Hardware Implementation

This algorithm suggests a straight-forward approach to implement an unified DCT/IDCT hardware system. Figures 1 and 2 show the flow diagrams for DCT and IDCT respectively. One can see that the switch from DCT to IDCT or vice versa of such a system implies the change of the order of the two hardware structures, namely, the recursive adder and the convolution module only. Note that the multiplier in the output stage is disabled in IDCT mode as the multiplication of the $c(k)$ value is embedded into the convolution module as shown in figure 2. This is not only able to reduce the computation time, but also reduces the computational error arose during multiplications.

The convolution module shown in figures 1 and 2 includes two convolvers. The distributed arithmetic technique[5] is applied here due to its simple structure. From eqns.14 and 17, we know that we have to realize two 2N-point cyclic convolutions. That means one has to construct two tables for the realization of the two equations if the distributed arithmetic technique is applied. In particular, we need a table of $\sum_{n=0}^{2N-1} b_n \cos\left[\frac{n^2\pi}{2N}\right] : b_n \in \{0,1\}$ for realizing eqn.14 and another table of $\sum_{n=0}^{2N-1} b_n \sin\left[\frac{n^2\pi}{2N}\right] : b_n \in \{0,1\}$ for realizing eqn.17. Hence, the basic size of each table will be 2^{2N} words. However, one can make a further simplification to reduce the table size required.

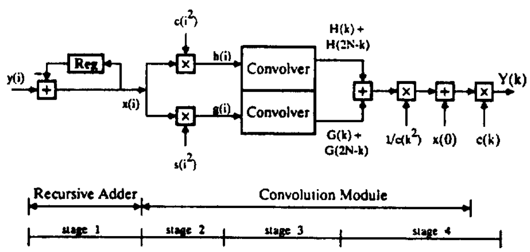

Fig 1. Flow diagram for the implementation of the DCT.

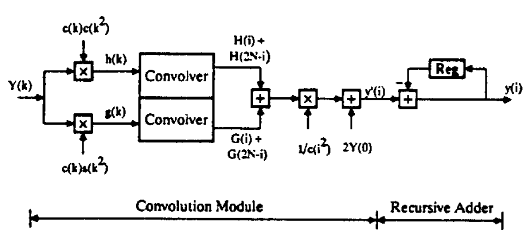

Fig 2. Flow diagram for the implementation of the IDCT.

By substituting $i = j+k$ into eqn.14, we have

$$H(k)+H(2N-k) = \sum_{j=1-k}^{2N-1-k} h'(j+k)\cos\left[\frac{j^2\pi}{2N}\right] \quad \text{for } k=1,2....N\text{-}1 \quad (20)$$

As $h'(0)=0$, eqn.20 can then be rewritten as

$$H(k)+H(2N-k) = \left\{ h'(k)+h'(N-k)\cos\left[\frac{N\pi}{2}\right] \right\} + \sum_{j=1}^{N-1} h''(j,k)\cos\left[\frac{j^2\pi}{2N}\right] \quad \text{for } k=1,2....N\text{-}1 \quad (21)$$

$$\text{where} \quad h''(j,k) = \begin{cases} h'(j+k)+h'(k-j) & \text{if } 0<j\le k \\ h'(j+k)+h'(2N-j+k) & \text{if } k<j<N \end{cases} \quad (22)$$

To realize eqn.21, only a table of $\sum_{n=1}^{N-1} b_n \cos\left[\frac{n^2\pi}{2N}\right] : b_n \in \{0,1\}$ is required. In such case, the basic table size can be reduced to 2^{N-1} words. Storage elements required can then be reduced to one 2^{N+1}th of the original approach. Similar simplification can be used to realize eqn.17.

Figure 3 shows the structure of such a convolver. In conventional distributed arithmetic structure, the address for accessing the ROM table is generated from a circular buffer[5]. In the initial state, data are loaded into the circular buffer. Then, a specific bit of each datum will be selected and combined with appropriate bits from other data to form an address to access the ROM table. Fetched data are shifted and accumulated to form

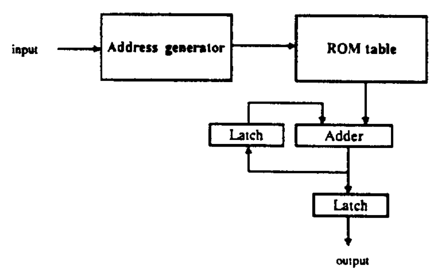

Fig 3a. Convolver constructed with distributed arithmetic.

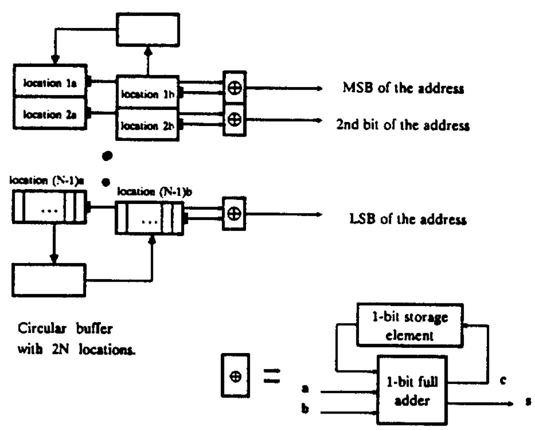

Fig 3b. Modified circular buffer used as the address generator in the convolver.

partial results until the final result comes out. Whenever an output is obtained, all data in the circular buffer will circulate by one step and the above procedure will be repeated until all outputs are obtained. For the structure proposed, the address is generated from a modified circular buffer as shown in figure 3b instead of a typical circular buffer.

Numerical Stability

From eqn.7, we know that the present approach involves the multiplication of the $\sec(k^2\pi/2N)$ term for each value of k. This will cause numerical instability of some results as $\cos(k^2\pi/2N)$ may equal to zero for specific values of k. In particular, as $\cos(k^2\pi/2N)=0$ implies $\langle k^2\rangle_{2N}=N$ $(0<k<N)$ and vice versa, it means one has to face this problem if $N\in\{(2m-3)n^2 : m,n=2,3,4...\}$. Fortunately, this problem can be easily resolved without too much additional effort especially when the distributed arithmetic technique is applied.

Obviously, if $\cos\frac{k_0^2\pi}{2N}=0$, we have $\sin\frac{k_0^2\pi}{2N}=1$ or -1. Moreover, we have

$$\sum_{i=1}^{2N-1} h'(i)\sin\frac{(i-k_0)^2\pi}{2N} - \sum_{i=1}^{2N-1} g'(i)\cos\frac{(i-k_0)^2\pi}{2N} = 2\sin\frac{k_0^2\pi}{2N}T(k) \tag{23}$$

Hence, by making a small modification, one can still make use of the structure proposed in the previous sections. In fact, only multiplexers are required to add to the convolver such that one can swap the data table fetched during the computation of the value of $T(k_0)$ as shown in figure 4. In general, the two convolvers perform the same operations as those described in the previous section. For the computation of other values of T(k), the circular buffer containing $\{h(i)\}$ generates the address to access the table of $\sum_{n=1}^{N-1} b_n \cos\left[\frac{n^2\pi}{2N}\right] : b_n\in\{0,1\}$ while the circular buffer containing $\{g(i)\}$ generates the address to access the table of $\sum_{n=1}^{N-1} b_n \sin\left[\frac{n^2\pi}{2N}\right] : b_n\in\{0,1\}$. In other words, no special treatment is required for this normal case. However, for the computation of the value of $T(k_0)$, we swap the data table such that the address generated from the circular buffer containing $\{g(i)\}$ and the circular buffer containing $\{h(i)\}$ are respectively used to access the table of $\sum_{n=1}^{N-1} b_n \cos\left[\frac{n^2\pi}{2N}\right] : b_n\in\{0,1\}$ and the table of $\sum_{n=1}^{N-1} b_n \sin\left[\frac{n^2\pi}{2N}\right] : b_n\in\{0,1\}$ instead. Then the outputs of the two convolvers will be $\sum_{i=1}^{2N-1} h'(i)\sin\frac{(i-k_0)^2\pi}{2N}$ and $\sum_{i=1}^{2N-1} g'(i)\cos\frac{(i-k_0)^2\pi}{2N}$ in this cycle. For the present design, the value of $1/c(k_0^2)$ is assigned to be $\sin(k_0^2\pi/2N)$ $(=1$ or $-1)$ and the value of $\sum_{i=1}^{2N-1} g'(i)\cos\frac{(i-k_0)^2\pi}{2N}$ is negated before being fed into stage 4 of the system. The same implementation structure as shown in figure 1 can then be used to obtain results of these special cases. Note that this trick can also be used to eliminate numerical instability in the case of $\cos(k^2\pi/2N)\approx 0$. Similar modification can be applied to the realization of the IDCT.

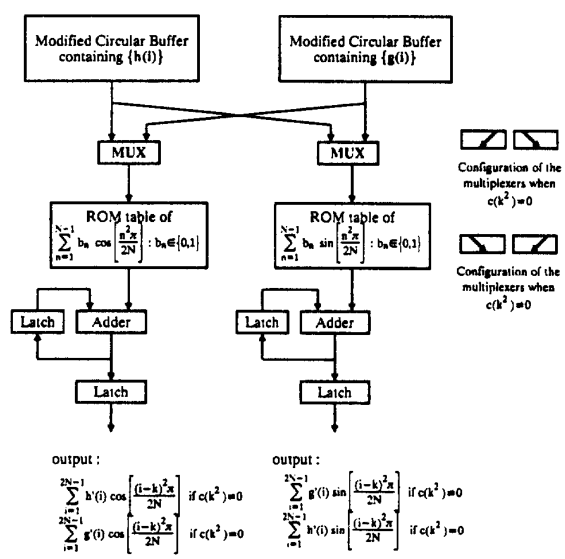

Fig 4. Convolvers swapping data table to avoid numerical instability by using multiplixers.

Conclusion

In this paper, we propose a general solution for the realization of the discrete cosine transform and the inverse discrete cosine transform. This algorithm converts the DCT/IDCT into convolution form such that one can easily implement it with technologies that are well suited for doing convolutions. This is an efficient and effective approach as it can avoid complicated data routing and data management. For example, there is no address generation problem and a simple pipeline structure can be applied to achieve parallel processing. Compared with conventional algorithms, this algorithm is much more flexible as it can be applied to realize DCT/IDCT with any length. On the other hand, this algorithm suggests an efficient approach to realize an unified hardware for the implementation of both the DCT and the IDCT. The realization of both the DCT and the IDCT can rely on the same convolution module.

References

[1] N.Ahmed, T.Natarajan and K.R.Rao, "Discrete cosine transform," IEEE Trans., 1974, C-23, pp.90-94.

[2] J.Makhoul, "A fast cosine transform in one and two dimensions," IEEE trans., Vol.ASSP-28, No.1, Feb 1980, pp.27-34.

[3] M.T.Heideman, "Computation of an odd-length DCT from a real-valued DFT of same length," IEEE trans. Vol.SP-40, No.1, Jan 1992, pp.54-61.

[4] Y.H.Chan and W.C.Siu, "Algorithm for prime length discrete cosine transform," IEE Electronics letters, Vol.26, No.3, Feb 1990, pp.206-208.

[5] S.A.White, "Applications of distributed arithmetic to digital signal processing: A tutorial review," IEEE ASSP Mag., Vol.6, No.3, July 1989, pp.4-19.