A Parallel Reconfigurable Architecture for DCT of Lengths
N=32/16/8

Parthiban K G* & Sabin.A.B**

*Professor, M.P. Nachimuthu M. Jaganathan Engineering College, Erode, India
**PG Scholar, M.P. Nachimuthu M. Jaganathan Engineering College, Erode, India

ABSTRACT:

Discrete Cosine Transform (DCT) algorithm is very effective due to its symmetry and simplicity. It is good replacement of Fast Fourier Transform (FFT) due to consideration of real component of image data. This paper introduces an orthogonal approximation for the 32-point DCT, using recursive sparse matrix decomposition and makes use of the symmetries of DCT basis vectors. The proposed transformation matrix contains only ones and zeros. Bit shift operations and multiplication operations are absent. The approximate transform of DCT is obtained to meet the low complexity requirements. The proposed image compression algorithm is comprehended using Matlab code. A fully scalable reconfigurable parallel architecture for the computation of approximate DCT of a 32-point DCT or for parallel computation of two 16-point DCTs or four 8-point DCTs with a marginal control overhead based on the proposed algorithm.

Index terms- Discrete Cosine Transform, Fast Fourier Transform (FFT), Matlab

INTRODUCTION

THE DISCRETE COSINE TRANSFORM (DCT) expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies. DCTs are important to numerous applications in science and engineering, from lossy compression of audio (e.g. MP3) and images (e.g. JPEG) (where small high-frequency components can be discarded), to spectral methods for the numerical solution of partial differential equations. The use of cosine rather than sine functions is critical for compression, since it turns out (as described below) that fewer cosine functions are needed to approximate a typical signal, whereas for differential equations the cosines express a particular choice of boundary conditions. DCT is mainly used in Image and Video compression. In particular, a DCT is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers.

Like any Fourier-related transform, DCTs express a function or a signal in terms of a sum of sinusoids with different frequencies and amplitudes. Like the Discrete Fourier Transforms (DFT), a DCT operates on a function at a finite number of discrete data points. The obvious distinction between a DCT and a DFT is that the former uses only cosine functions, while the latter uses both cosines and sines (in the form of complex exponentials). However, this visible difference is merely a consequence of a deeper distinction: a DCT implies different boundary conditions from the DFT or other related transforms. The Fourier-related transforms that operate on a function over a finite domain, such as the DFT or DCT or a Fourier series, can be thought of as implicitly defining an extension of that function outside the domain. That is, once you write a function f(x) as a sum of sinusoids, you can evaluate that
sum at any $x$, even for $x$ where the original $f(x)$ was not specified. The DFT, like the Fourier series, implies a periodic extension of the original function. A DCT, like a cosine transform, implies an even extension of the original function.

Image data compression has been an active research area for image processing over the last decade and has been used in a variety of applications. Image and video data compression refers to a process in which the amount of data used to represent image and video is reduced to meet a bit rate requirement (below or at most equal to the maximum available bit rate), while the quality of the reconstructed image or video satisfies a requirement for a certain application and the complexity of computation involved is affordable for the application.

Recently, two new transforms have been proposed for 8-point DCT approximation: Cintra have proposed a low-complexity 8-point approximate DCT based on integer functions and Potluri have proposed a novel 8-point DCT approximation that requires only 14 additions. On the other hand, Bouguezel have proposed two methods for multiplication-free approximate form of DCT. The first method is for length $N=8, 16$ and $32$, and is based on the appropriate extension of integer DCT. Cintra have proposed a new 16X16 matrix also for approximation of 16-point DCT and have validated it experimentally.

II. IMAGE AND VIDEO COMPRESSION

Image and video data compression refers to a process in which the amount of data used to represent image and video is reduced to meet a bit rate requirement (below or at most equal to the maximum available bit rate), while the quality of the reconstructed image or video satisfies a requirement for a certain application and the complexity of computation involved is affordable for the application. Image compression system has three main blocks (i) a transform (usually DCT on 8x8 blocks), (ii) a quantizer, (iii) a lossless (Entropy) coder; each tries to throw away information which is not essential to understand the image but costs bits.

The DCT transform throws away correlations, if we make a plot of the value of a pixel as a function of one of its neighbors, we can see that the pixels are highly correlated (i.e. most of the time they are very similar), and this is just a consequence of the fact that surfaces are smooth. The advantage of working in the frequency domain is that our visual system is less sensitive to distortion around edges. The transition associated with the edge masks our ability to perceive the noise.

Fig.1 Image Compression Technique

Compression is a reversible conversion (encoding) of data that contains fewer bits. This allows a more efficient storage and transmission of the data. The inverse process is called decompression (decoding). Software and hardware that can encode and decode are called decoders. Both combined form a codec and should not be confused with the terms data container or compression algorithms. Lossless compression allows a 100% recovery of the
original data. It is usually used for text or executable files, where a loss of information is a major damage. These compression algorithms often use statistical information to reduce redundancies. Huffman-Coding and Run Length Encoding are the two popular examples allowing high compression ratios depending on the data. Using lossy compression does not allow an exact recovery of the original data. Lossy compression allows higher compression ratios than lossless compression.

III. EXISTING METHOD

The 8-point discrete cosine transform (DCT) is a key step in many image and video processing applications. This particular block length is widely adopted in several image and video coding standards, such as JPEG, MPEG-1, MPEG-2, and H.261. This is mainly due to its good energy compaction properties. This 8-point DCT is used as the basis for the proposed system for 16, 32 and 64 point DCTs. This correspondence introduced an approximation algorithm for the DCT computation based on matrix polar decomposition. This method could outperform the BAS-2008 method in high and low-compression ratios scenarios. Moreover, this method possesses constructive formulation based on the round-off function. Therefore, generalizations are more readily possible. This existing transformation matrix contains only zeros and ones, multiplications and bit shift operations are absent.

![Fig. 2 Signal Flow Graph of DCT-8](image)

Most of the existing algorithms for approximation of the DCT target only the DCT of small transform lengths such as 16-point and 32-point is not possible and some of them are non-orthogonal. If the transform is orthogonal, we can always find its inverse and the kernel matrix of the inverse transform is obtained by just transposing the kernel matrix of the forward transform.

As specified in the recently adopted HEVC, DCT of different lengths such as N=8, 16, 32 are required to be used in video coding applications. Therefore, a given DCT architecture should be potentially reused for the DCT of different lengths instead of using separate structures for different lengths. Here proposing such reconfigurable DCT structures which could be reused for the computation of DCT of different lengths. The proposed reconfigurable architecture for the implementation of approximated 32-point DCT is shown in Fig. 3. It consists of four computing units, two 16-point approximated DCT units, output permutation unit and a 32-point input adder unit that generates a(i) and b(i), i ∈ [1:15]. The input to the first 16-point DCT approximation unit is fed through 16 MUXes that select either [a(0), a(1), . . . , a(14), a(15)] or [X(0),X(1), . . . ,X(14), X(15)], depending on whether it is used for 32-point DCT calculation or 16-point DCT calculation. Similarly, the input to the second 16-point DCT unit (Fig. 4.3) is fed through 16 MUXes that select either [b(0),b(1), . . . ,b(14), b(15)] or
[X(0),X(1), . . . ,X(14), X(15)], depending on whether it is used for 32-point DCT calculation or 16-point DCT calculation.

Then outputs from the 16-point adder units and output from the 32-point adder units are fed to another section of MUXes in which, input to the first 8-point DCT approximation unit is fed through 8 MUXes that select either [a(0), a(1), . . . , a(7)] or [m(0),m(1), . . . ,m(7)], depending on whether it is used for 16-point DCT calculation or 8-point DCT calculation. Then the MUXes for the next 8-point DCT approximation unit selects either [a(8), a(9), . . . , a(15)] or [m(8),m(9), . . . ,m(15)], depending on whether it is used for 16-point DCT calculation or 8-point DCT calculation. Similarly, the input to the third and fourth 8-point DCT units(Fig. 3) are fed through 16 MUXes, in which first 8 MUXes select either [b(0),b(1),…,b(7)] or [m(0),m(1),……….. m(7)] ), depending on whether it is used for 16-point DCT calculation or 8-point DCT calculation, and the next 8 MUXes select either [b(8),b(9), . . . , b(15)] or [m(8),m(9), . . . ,m(15)], depending on whether it is used for 16-point DCT calculation or 8-point DCT calculation. On the other hand, the output permutation unit uses thirty 3-input MUXes to select and re-order the output depending on the size of the selected DCT.

Sel32 is used as control input for the first 32 MUXes used in the 32/16 computation unit, sel16 is used as control input for the second 32 MUXes used in the 16/8 computation unit to select inputs and to perform permutation according to the size of the DCT to be computed. Specifically, Sel32=1 & Sel16=1, enables the computation of 32-point DCT, Sel32=0 & Sel16=1, enables the computation of a pair of 16-point DCTs in parallel and finally when Sel32=0 & Sel16=0 enables the computation of four 8-point DCTs in parallel. Consequently, the architecture of Fig. 4.3 allows the calculation of a 32-point DCT or two 16-point DCTs in parallel or four 8-point DCTs in parallel.
IV. PROPOSED SYSTEM
For increasing the Maximum Operating Frequency (MOF), reducing the number of registers and LUTs used, proposing a parallel pipelined structure in which each set of input values (first 32 inputs) are first computed using the first 32-point block, at the same time all the different 32 inputs are computed parallel so that achieving more MOFs and increased speed. Parallel reconfigurable structure for the computation of DCTs of lengths N= 32/16/8 are shown in the Fig.4 given below.

V. SIMULATION RESULTS
The input to the first 16-point DCT approximation unit is fed through 16 MUXes that select either [a(0), a(1), . . . , a(14), a(15)] or [X(0), X(1), . . . , X(14), X(15)], depending on whether it is used for 32-point DCT calculation or 16-point DCT calculation. Similarly, the input to the second 16-point DCT unit is fed through 16 MUXes that select either [b(0), b(1), . . . , b(7)] or [X(0), X(1), . . . , X(14), X(15)], depending on whether it is used for 32-point DCT calculation or 16-point DCT calculation.
The output of 32-Point DCT transform when both the select signal is ‘1’ shown in Fig. 5.3 & 5.4. Outputs are given by F0, F1, F2, F3 . . . . . , F15 . . . , F30, F31. The energy of the image is compacted towards the first row of the output register. Also most of the remaining values are zero (0), so that image can be easily transmitted.
Fig. 5.4 Output of Proposed 32-point DCT Transform when “Sel32=1 & Sel16=1” [F16 to F31]

The output of 32-point DCT transform when Sel32=0 & Sel16=1 is shown in Fig. 5.5 & 5.6. The computation of two 16-point DCTs in parallel occurs in which most of the remaining values are zero (0), so that the image can be easily transmitted.

Fig. 5.5 Output of Proposed 32-point DCT Transform when “Sel32=0 & Sel16=1” [F0 to F15]
Fig. 5.6 Output of Proposed 32-point DCT Transform when “Sel32=0 & Sel16=1” [F16 to F31]

The output of 32-point DCT transform when Sel32=0 & Sel16=0 is shown in Fig.5.7 & 5.8. The computation of four 8-point DCTs in parallel occurs in which most of the remaining values are zero (0), so that image can be easily transmitted.

Fig. 5.7 Output of Proposed 32-point DCT Transform when “Sel32=0 & Sel16=1” [F0 to F16]
VI. REDUCED DELAY

Proposed design involves nearly 7% less area for 16-point DCT, 6% less area for 32-point DCT, also will have a 5% less area for 64-point DCT on comparing with existing systems like BDCT.

VII. COMPARISON OF EXISTING AND PROPOSED SYSTEMS

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>EXISTING SYSTEM</th>
<th>PROPOSED SYSTEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation complexity is low</td>
<td>22 adders for 8-point. 72 adders for 16-point 160 adders for 32-point</td>
<td>22 adders for 8-point 60 adders for 16-point 152 adders for 32-point</td>
</tr>
<tr>
<td>Delay is less</td>
<td>22 adders for 8-point 72 adders for 16-point 160 adders for 32-point</td>
<td>22 adders for 8-point 60 adders for 16-point 152 adders for 32-point</td>
</tr>
<tr>
<td>MOF</td>
<td>215.3 for 8-point 167.0 for 16-point 136.4 for 32-point</td>
<td>500.9 for 8-point 496.2 for 16-point 490.4 for 32-point</td>
</tr>
</tbody>
</table>
VIII. CONCLUSION AND FUTURE SCOPE

In this paper, I have proposed a generalized recursive algorithm to obtain orthogonal approximation of DCTs of larger lengths where the computation of 32-point DCT is configured for parallel computation of two 16-point DCTs and four 8-point DCTs. Future enhancement can be to propose a fully scalable reconfigurable architecture for approximate DCT computation where the computation of 64-point DCT could be configured for parallel computation of two 32-point DCTs and four 16-point DCTs and also can be configured for the parallel computation of eight 8-point DCTs.

REFERENCES


