Reconfigurable Scalable Architecture for Discrete Cosine Transform

C. Abinaya & S. M. BalaMurugan
1P.G. Student, Department of Electrical and Electronics Engineering, G.K.M.CET
2Assistant Professor, Department of Electrical and Electronics Engineering, G.K.M.CET

Abstract
An approximate kernel for the discrete cosine transform (DCT) of length four comes from the 4 point DCT outlined by the High efficiency Video coding (HEVC) standard, and used that for the computation of DCT and inverse DCT (IDCT) of power of 2 lengths. There are 2 reasons to think about the DCT of length four because the basic module. Firstly, it permits to compute DCTs of length four, 8, 16, and thirty two prescribed by HEVC. Moreover, the DCTs generated by 4 point DCT not solely involve lower complexity however additionally supply higher compression performance. Comparing to existing method, DCT offer better Compression performance. The proposed method can perform HEVC compliant video coding. Unified forward and inverse rework architecture is additionally planned wherever the hardware complexness is reduced by sharing of hardware between DCT and IDCT computation. The planned approximation has nearly an equivalent arithmetic complexness and hardware demand as those of recently planned connected ways, but involves considerably less error energy.

KEYWORDS- DCT; HEVC; IDCT.

1. INTRODUCTION

The discrete cosine transform (DCT) is popularly utilized in image and video compression. Since the DCT is calculatedly accurate, many algorithms are planned within the literature to work out it with accurate. Recently, vital work has been done to derive approximate of 8-point DCT for reducing the computational complexness. There is vital scope for reduction of power consumption in video codec by algorithmic rule and design level performance. In algorithmic rule level, procedure complexness will be considerably reduced by appropriate approximation. Algorithm architecture co design on the opposite hand will give vital reduction in space, computation time, and therefore the general energy consumption in video secret writing and decipherment. It’s found that the distinct circular function rework (DCT), inverse DCT (IDCT), and filtering operation will tolerate some errors, and so can enable some approximation. For mobile and time period applications, number DCT is needed to be enforced as hardware accelerator. The HEVC standard permits to use DCT of lengths of \( N = \text{four}, 8, 16, \) and \( 32 \). Therefore, a simple hardware implementation would need a separate hardware section for the DCTs of different lengths. To cut back the silicon space, it’s needed to implement the DCTs of various lengths during a reconfigurable hardware which might be designed to calculate the DCT of any of the specified lengths. There are 2 main problems encountered during mapping of DCT algorithms to reconfigurable hardware architectures. But the prevailing DCT algorithms don’t give the most effective of all the above 3 necessities. A number of the prevailing strategies are. A unit deficient in terms of quantity ability, generalization for higher sizes, and orthogonal. We intend to maintain orthogonal in the approximate DCT for 2 reasons. Firstly, if the transform is orthogonal, we are able to continuously notice its inverse, and the kernel matrix of the inverse rework is obtained by simply transposing the kernel matrix

\[
\hat{C}_N = P_N \begin{pmatrix} 1 & 0 \\ 0 & S_N \end{pmatrix} \begin{pmatrix} I_N \ 0 & J_N \ I_N \end{pmatrix}
\]


or

\[
\hat{C}_4 = 32 \times R \left( C_4(i,j) + 32 \times \text{sgn}(C_4(i,j)) \right) \\
= \begin{bmatrix} 64 & 64 & 64 & 64 \\ 64 & -36 & -64 & -64 \\ 64 & -36 & 64 & -36 \\ 64 & 64 & -36 & -64 \\ 64 & -36 & 64 & -64 \\ 64 & -36 & 64 & -64 \\ 64 & -36 & 64 & -36 \end{bmatrix}
\]

of the forward rework. This feature of inverse rework may well be wont to cipher the forward and inverse DCT by similar computing structures.
Moreover, in case of orthogonal transforms, similar quick algorithms area unit applicable to each forward and inverse transforms.

2. APPROXIMATED FOUR POINT DCT

A. PROPOSED DCT APPROXIMATION

For input vector \( X = \{x[0], x[1], \ldots\}^T \) and its corresponding output \( Y = \{y[0], y[1], \ldots\}^T \) and is represented by \( Y = CNX \), where \( CN \) denotes the DCT Kernel matrix. The DCT is derived by recursive Sparse decomposition of matrix kernel. The Splitting Procedure can be applied to the DCT of any power of 2 length, higher than 4, but we restrict the present study to 8, 16, and 32. This means that DCT approximation of higher sizes can be made available for software experimental implementation also. The 4-point integer DCT kernel of HEVC is given as. To achieve full scalability and reduce the computational complexity of DCT, we need to approximate \((CN/2)\) and \((SN/2)\).

B. PROPOSED ARCHITECTURE FOR FOUR POINT DCT

The 4 point approximate integer DCT kernel given by where \( a(0) = x(0) + x(3) \); \( a(1) = x(1) + x(2) \); \( b(0) = x(0) - x(3) \); and \( b(1) = x(1) - x(2) \).

C. PROPOSED ARCHITECTURES FOR APPROXIMATE DCT OF HIGHER LENGTHS

Using the proposed scheme, the 8-point approximate DCT can be obtained

\[
\begin{bmatrix}
  t(0) \\
  t(1) \\
  t(2) \\
  t(3)
\end{bmatrix}
= \begin{bmatrix}
  64 & 64 & 64 & 64 \\
  64 & 32 & -32 & -64 \\
  64 & -64 & 64 & -64 \\
  -32 & -64 & -64 & 32
\end{bmatrix}
\begin{bmatrix}
  s(0) \\
  s(1) \\
  s(2) \\
  s(3)
\end{bmatrix}
\]

where \( t(0) = y(0), t(1) = y(2), t(2) = y(4), t(3) = y(6) \) in upper DCT unit \( s(0) = x(0) + x(7); s(1) = x(1) + x(6) \) and \( s(2) = x(2) + x(5); s(3) = x(3) + x(4) \) and \( t(0) = y(1), t(1) = y(3), t(2) = y(5), t(3) = y(7) \) in lower DCT units \( s(0) = x(0) - x(7); s(1) = x(1) - x(6) s(2) = x(2) - x(5); s(3) = x(3) - x(4) \).

D. UNIFIED INVERSE AND FORWARD RECONFIGURABLE DCT ARCHITECTURE

For mobile applications wherever devices ought to support playback video together with video capture and recording, forward and inverse transforms ought to be enforced within the same device. Some works are done recently to appreciate each forward and inverse work within the same device. The objective of unified architectures is to maximally share the hardware by the forward and therefore the inverse transforms.

![Unified design for approximate 32](image)

**E. SCALING IN FORWARD TRANSFORM**

**Table I**

<table>
<thead>
<tr>
<th>Description</th>
<th>Scale Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>First forward transform stage</td>
<td>( 2^{0+M/2} )</td>
</tr>
<tr>
<td>After the first forward transform stage ( (S_{T1}) )</td>
<td>( 2^{-M} )</td>
</tr>
<tr>
<td>Second forward transform stage</td>
<td>( 2^{0+M/2} )</td>
</tr>
<tr>
<td>After the second forward transform stage ( (S_{T2}) )</td>
<td>( 2^{-M} )</td>
</tr>
<tr>
<td>Total scaling for the forward transform</td>
<td>( 2^{15-B-M} )</td>
</tr>
</tbody>
</table>

**F. SCALING USED IN INVERSE TRANSFORM**

**Table II**

<table>
<thead>
<tr>
<th>Description</th>
<th>Scale Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>First inverse transform stage</td>
<td>( 2^{0+M/2} )</td>
</tr>
<tr>
<td>After the first inverse transform stage ( (S_{T1}) )</td>
<td>( 2^{-7} )</td>
</tr>
<tr>
<td>Second inverse transform stage</td>
<td>( 2^{0+M/2} )</td>
</tr>
<tr>
<td>After the second inverse transform stage ( (S_{T1}) )</td>
<td>( 2^{-(15-8-B)} )</td>
</tr>
<tr>
<td>Total scaling for the inverse transform</td>
<td>( 2^{-(15-8-B)} )</td>
</tr>
</tbody>
</table>

3. RESULT AND DISCUSSION FOUR POINT DCT
A. AREA SUMMARY FOR 4 PT DCT

TABLE III

<table>
<thead>
<tr>
<th>LOGIC UTILIZATION</th>
<th>USED</th>
<th>AVAILABLE</th>
<th>UTILIZED</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO OF SLICE LUT’S</td>
<td>1408</td>
<td>5720</td>
<td>24%</td>
</tr>
<tr>
<td>NO OF ULLY USED LUT-FF PAIRS</td>
<td>0</td>
<td>1408</td>
<td>0%</td>
</tr>
<tr>
<td>NO OF BANNED IOB’S</td>
<td>880</td>
<td>102</td>
<td>862%</td>
</tr>
</tbody>
</table>

B. UNIFIED DESIGN

TABLE IV

<table>
<thead>
<tr>
<th>Device Utilization Summary (estimated values)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic Utilization</td>
</tr>
<tr>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Number of Slice LUTs</td>
</tr>
<tr>
<td>Number of full used LUT-FF pairs</td>
</tr>
<tr>
<td>Number of bonded IOBs</td>
</tr>
</tbody>
</table>

C. DELAY SUMMARY FOR FOUR POINT DCT

Timing Details:
All values displayed in nanoseconds (ns)
Timing constraint: Default path analysis
Total number of paths / destination ports: 252 / 32
Delay: 7.261ns (Levels of Logic = 4)
Source: u0<11> (PAD)
Destination: m0<31> (PAD)
Data Path: u0<11> to m0<31>
Gate Net
Cell: in->out fan out Delay Logical Name (Net Name)
IBUF: I->O 3 1.222 0.995 u0_11_IBUF (u0_11_IBUF)
LUT6: I1->O 4 0.203 0.912 Madd_m0_cy<5>|11 (Madd_m0_cy<5>)
LUT4: I1->O 23 0.205 1.153 m0<9>|1> (m0_9_OBUF)
OBUF: I->O 2.571 m0_31_OBUF (m0<31>)
Total 7.261ns (4.201ns logic, 3.060ns route)
(57.9% logic, 42.1% route)

D. DELAY SUMMARY FOR UNIFIED DESIGN

Timing Details:
All values displayed in nanoseconds (ns)
Timing constraint: Default path analysis
Total number of paths / destination ports: 58 / 29
Delay: 6.042ns (Levels of Logic = 3)
Source: invforw (PAD)
Destination: zz0<31> (PAD)
Data Path: invforw to zz0<31>
Gate Net
Cell: in->out fan out Delay Logical Name (Net Name)
IBUF: I->O 21 1.222 1.218 invforw_IBUF (invforw_IBUF)
LUT2: I0->O 9 0.203 0.829 Mmux_mux_1131 (mux_1<20>)
OBUF: I->O 2.571 zz0_31_OBUF (zz0<31>)
Total 6.042ns(3.996ns logic, 2.046ns route)
(66.1% logic, 33.9% route)

4. CONCLUSION

In this paper, we've projected a completely scalable HEVC compliant orthogonal number approximation of DCT of any power of two length, N>4, that gives versatile trade-off between space and time complexities, and would possibly merely be realized throughout reconfigurable hardware with terribly low reconfiguration overhead. The projected approximation provides better compressed image quality with similar arithmetic complexity and hardware consumption compared with earlier approximation schemes. A completely scalable and reusable architectures are also projected for the computation of approximate DCT where 8 point DCT structure are accustomed compute a pair of 4 point DCT exploitation. The reuse structure of 32-point DCT could be designed for parallel calculation of two 16-point DCTs or four 8 point DCTs or eight 4 point DCTs. The projected reuse Scalable design can help real-time writing for high-definition video sequences.

5. References