Implementation of Multi Standard Transform For High Throughput Core Supporting Mpeg/H.264/Vc-1 Based on CSDA

T.Shivalingappa
PG Scholar,
Department of ECE,
Intellectual Institute of Technology, AP, India.

K.Madan Mohan
Assistant Professor,
Department of ECE,
Intellectual Institute of Technology, AP, India.

Abstract:

Transforms are widely used in video and image applications. proposes a low-cost high-throughput Multi standard transform (MST) core, which can support MPEG- 1/2/4 (8 × 8), H.264 (8 × 8, 4 × 4), and VC-1 (8 × 8, 8 × 4, 4×8, 4×4) transforms. Common sharing distributed arithmetic (CSDA) combines factor sharing and distributed arithme-
tic sharing techniques, efficiently reducing the number of adders for high hardware-sharing capability. With eight parallel computation paths, the proposed MST core has an eightfold operation frequency throughput rate. Measurements show that the proposed CSDA MST core achieves a high-throughput rate the proposed CSDA combines the FS and DA methods. The FS method is adopted first to identify the factors that can achieve higher capability in hardware resource sharing. The DA method is used to find the shared coefficient based on the results of the FS method. The adder-tree circuits will be followed by the proposed CSDA circuit.

Index Terms:

Common sharing distributed arithmetic (CSDA), discrete cosine transform (DCT), integer transform, multistandard transform (MST).

1 INTRODUCTION:

Transforms are widely used in video and image applications. Several groups, such as The International Organiza-
tion for Standardization (ISO), International Telecommunica-
ion Union Telecommunication Standardization Sector (ITU-T), and Microsoft Corporation, have developed various transform dimensions and coefficients, corresponding to different applications. Numerous researchers have worked on transform core designs, including discrete cosine transform (DCT) and integer transform, using distributed arithmetic (DA) factor sharing (FS) and matrix decomposition methods to reduce hardware cost.

The inner product can be implemented using ROMs and accumulators instead of multipliers to reduce the area cost. Yu and Swartzlander present an efficient method for reducing ROMs size with recursive DCT algorithms. Although ROMs are likely to scale much better than other circuits with shrinking technology nodes, several ROM-free DA architectures have recently emerged. Shams et al. employ a bit-level sharing scheme to construct the adder-based butterfly matrix, called new DA (NEDA). To improve the throughput rate of the NEDA method, high-throughput adder trees are introduced in.

Chang et al. use a delta matrix to share hardware resources using the FS method. They derive matrices for multi standards as linear combinations from the same matrix and delta matrix, and show that the coefficients in the same matrix can share the same hardware resources by factorization. To further reduce the area, Qi et al. present optimization strategies for FS and adder sharing (AS) for multi-standard (MST) applications. Fan and Su use the matrix decomposition method to establish the sharing circuit. Matrices for VC-1 transformations can be decomposed into several small matrices, a number of which are identical for different points transforms. Hardware resources can be shared.

Moreover, other previous works on hardware resource sharing are presented. Recently, reconfigurable architectures have been presented as a solution to achieve a good flexibility of processors in field-programmable gate array (FPGA) platform or application-specific integrated circuit (ASIC), such as AsAP, Ambric, MORA, and Smart Cell. Although these reconfigurable architectures have the feature of flexibility, the pure ASIC design can be recommended for a fixed customer application suitably. Because of the same properties in DCT and integer transform applied to Moving Picture Experts Group (MPEG) and Windows Media Video 9 (WMV-9/VC-1), many MST cores are presented. Hwangbo and Kyung introduce a fully supported transform core for the H.264 standard, including 8 × 8 and 4 × 4 transforms.
2COMMON SHARING DISTRIBUTED ALGORITHM:

Mathematical derivation of proposed Common sharing distributed Algorithm. To gain better resource sharing of inner product operation, The proposed CSDA method combines Factor Sharing and Distributed methods. To find FS and DA methods are described in the following.

A.Mathematical derivation of Factor Sharing:

The factor sharing method shares the same factor in different coefficients of applied input. Consider two different elements s1 and s2 with same input X as an example.

\[ S1 = C1 \times X, \quad S2 = C2 \times X \]

Assuming the same factor Fs can be found in coefficients C1 and C2, S1 and S2 can be rewritten as follows.

\[ S1 = (F_{s1}k_1 + F_{d1})X \]
\[ S2 = (F_{s2}k_2 + F_{d2})X \]

Where \( k_1 \) and \( k_2 \) indicate the weight position of the sharing factor Fs in C1 and C2 respectively. \( F_{d1} \) and \( F_{d2} \) denote the remainder coefficients after extracting sharing factor Fs for C1 and C2 respectively.

\[ F_{d1} = C1 \times F_{s2} \]
\[ F_{d2} = C2 \times F_{s2} \]

b.Mathematical derivation of Common sharing distributed arithmetic:

The inner product for a general multiplication and accumulation can be written as

\[ Y = A^T \times X = \sum_{i=1}^{L} A_i \times X_i \]

where \( X_i \) is input data, \( A_i \) is N-bit CSD co efficient. It can be expressed as follow.

\[
\begin{bmatrix}
A_{1,0} & A_{2,0} & \cdots & A_{L,0} \\
A_{1,1} & A_{2,1} & \cdots & A_{L,1} \\
\vdots & \vdots & \ddots & \vdots \\
A_{1,(N-1)} & A_{2,(N-1)} & \cdots & A_{L,(N-1)}
\end{bmatrix}
\begin{bmatrix}
X_1 \\
X_2 \\
\vdots \\
X_L 
\end{bmatrix}
\]

\[ = \begin{bmatrix} 2^0 & 2^{-1} & \cdots & 2^{-(N-1)} \end{bmatrix} \begin{bmatrix} Y_0 \\
Y_1 \\
\vdots \\
Y_{(N-1)} \end{bmatrix} \]

Where \( Y_1 = \sum_{i=1}^{L} A_i \times X_i \), \( A_{i,j} \) is (-1, 0, 1) and

j=0, ..., (N-1). If \( Y_j \) can be calculated by adding or subtracting \( X_i \) with \( A_{i,j} \neq 0 \). The product \( Y \) can then be obtained by shifting and adding every non-zero \( Y_j \).

C.1-D Common Sharing Distributed arithmetic-MST:

Based on the proposed CSDA algorithm, the coefficients for MPEG-1/2/4, H.264, and VC-1 transforms are chosen to achieve high sharing capability for arithmetic resources. To adopt the searching flow, software code will help to do the iterative searching loops by setting a constraint with minimum nonzero elements. In this paper, the constraint of minimum nonzero elements is set to be five. After software searching, the coefficients of the CSD expression, where 1 indicates −1. Note that the choice of shared coefficient is obtained by some constraints. Thus, the chosen CSDA coefficient is not a global optimal solution. It is just a local or suboptimal solution. Besides, the CSD codes are not the optimal expression, which have minimal nonzero bits. However, the chosen coefficient of CSD expression can achieve high sharing capability for arithmetic resources by using the proposed CSDA algorithm. More information about CSDA coefficients for MPEG-1/2/4, H.264, and VC-1 transforms.

![Fig.1. Architecture of proposed 1-D CSDA-MST](image)
After the SBF module, the CSDA_E and CSDA_O execute and by feeding input data a and b, respectively. The CSDA_E calculates the even part of the eight-point transform, similar to the four-point transform for H.264 and VC-1 standards.

Within the architecture of CSDA_E, two pipeline stages exist (12-bit and 13-bit). The first stage executes as a four-input butterfly matrix circuit, and the second stage of CSDA_E then executes by using the proposed CSDA algorithm to share hardware resources in variable standards.

Odd part common sharing distributed arithmetic circuit:

Similar to the CSDA_E, the CSDA_O also has two pipeline stages. Based on the proposed CSDA algorithm, the CSDA_O efficiently shares the hardware resources among the odd part of the eight-point transform and four-point transform for variable standards.

It contains selection signals of multiplexers (MUXs) for different standards. Eight adder trees with error compensation (ECATs) are followed by the CSDA_E and CSDA_O, which add the nonzero CSDA coefficients with corresponding weight as the tree-like architectures. The ECATs circuits can alleviate truncation error efficiently in small area design when summing the nonzero data all together.

d. Proposed CSDA Algorithm:

The proposed CSDA combines DA and FS methods. By expand the coefficients matrix at bit level The Factor sharing method first shares the same factor in each coefficient ,the distributed method is then applied to share the same combination of Input among each coefficient position. The proposed CSDA algorithm in matrix inner product can explain as follows.

\[
\begin{bmatrix}
Y_1 \\ Y_2
\end{bmatrix} =
\begin{bmatrix}
C_{11} & C_{12} \\ C_{21} & C_{22}
\end{bmatrix}
\begin{bmatrix}
X_1 \\ X_2
\end{bmatrix}
\]

Where the coefficients C11 ~ C22 are all five bits CSD numbers

- C11=[1 -1 1 0 0]
- C12=[1 -1 0 0 1]
- C21=[1 1 -1 0 0]
- C22=[0 1 -1 0 0]

Fig.5. shows the proposed CSDA sharing flow. The shared factor Fs in four coefficients is [1 -1] and C11 ~ C22 can use instead of [1 -1] ,with the the corresponding position under the FS method. The Distributed Arithmetic is applied to share the same position for the input, and the DA shared coefficient DA1=(X1+X2)Fs. Finally ,the matrix inner product in above equation can be implemented by shifting and adding every nonzero weight position. Fig. Shows the coefficient searching flow of the proposed Common Sharing Distributed Arithmetic algorithm.
2-D common sharing distributed arithmetic-MST core: This section provides a discussion of the hardware resources and system accuracy for the proposed 2-D CSDA-MST core and also presents a comparison with previous works. Finally, the characteristics of the implementation into a chip are described.

**TMEM:**

The TMEM is implemented using 64-word 12-bit dual-port registers and has a latency of 52 cycles. Based on the time scheduling strategy and result of the time scheduling strategy, the 1st-D and 2nd-D transforms are able to be computed simultaneously. The transposition memory is an 8×8 register array with the data width of 16 bits and is shown in Fig.7.

**a. Mathematical derivation of eight-point and four-point transforms:**

We introduce the proposed 2-D CSDA-MST core implementation. Neglecting the scaling factor, the one dimensional (1-D) eight-point transform can be defined as follows.

Because the eight-point coefficient structures in H.264, MPEG-1/2/4, and VC-1 standards are the same, the eight-point transform for these standards can use the same mathematical derivation. According to the symmetry property, the 1-D eight point transform in (7) can be divided into even and odd two four-point transforms, $Z_e$ and $Z_o$, as listed in (8) and (9), respectively.

$$C = \begin{bmatrix} Z_0 \\ Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \\ Z_5 \\ Z_6 \\ Z_7 \end{bmatrix} = C_i,$$

$$C_i = \begin{bmatrix} c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 \\ c_1 & c_3 & c_5 & c_7 & c_7 & c_5 & c_3 & c_1 \\ c_2 & c_6 & c_2 & c_6 & c_2 & c_6 & c_2 & c_6 \\ c_3 & c_7 & c_1 & c_3 & c_3 & c_1 & c_3 & c_7 \\ c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 & c_4 \\ c_5 & c_1 & c_3 & c_5 & c_7 & c_7 & c_5 & c_3 \\ c_6 & c_6 & c_2 & c_6 & c_2 & c_6 & c_2 & c_6 \\ c_7 & c_7 & c_3 & c_5 & c_7 & c_7 & c_5 & c_3 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}$$

$$Z_e = \begin{bmatrix} Z_0 \\ Z_1 \\ Z_2 \\ Z_3 \end{bmatrix} = \begin{bmatrix} c_{eo} \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \end{bmatrix}$$

$$Z_o = \begin{bmatrix} Z_4 \\ Z_5 \\ Z_6 \\ Z_7 \end{bmatrix} = \begin{bmatrix} c_{eo} \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \end{bmatrix}$$

Where

$$a = \begin{bmatrix} x_0 + x_1 \\ x_1 + x_2 \\ x_2 + x_3 \\ x_3 + x_4 \end{bmatrix}, \quad b = \begin{bmatrix} x_0 - x_1 \\ x_1 - x_2 \\ x_2 - x_3 \\ x_3 - x_4 \end{bmatrix}.$$  

The even part of the operation in (9) is the same as that of the four-point VC-1 and H.264 transformations. Moreover, the even part $Z_e$ can be further decomposed into even and odd parts: $Z_{ee}$ and $Z_{eo}$.
b. Architecture of the Proposed 2-D CSDA-MST Core:

The architecture of the 1-D eight-point MST core is shown in Fig.1, which consists of a selected butterfly (SBF) module, an even part CSDA (CSDA_E), an odd part CSDA(CSDA_O), eight error-compensated error trees (ECATs), and a permutation module. The following example serves to clarify the operation of the proposed CSDA algorithm. When the MPEG-1/2/4 standard is conducted, Z0 can be obtained by executing $c_4 A_0 + c_4 A_1$.

$$Z_0 = c_4 A_0 + c_4 A_1$$

Evaluation and odd parts: $Z_{even}$ and $Z_{odd}$

- The even part of the operation in (9) is the same as that computed by the CSDA_O efficiently shares the hardware resources among the odd part of the eight-point transform in (9) and four-point transform in (9) for variable standards. The proposed system by reducing the Area of converting one dimensional to two dimensional core designs. And also will reduce the total number of adders by using the CSDA Common Sharing Distributed Arithmetic method in the proposed system.

4. SIMULATION RESULTS:

The Multi standard transform core written in verilog, compiled and simulation using modelsim. The circuit simulated and synthesized. The simulated result for Multi standard transform core.

10. CONCLUSION:

The CSDA-MST core can achieve high performance, with a high throughput rate and low-cost VLSI design, supporting MPEG-1/2/4, H.264, and VC-1 MSTs. By using the proposed CSDA method, the number of adders and MUXs in the MST core can be saved efficiently. Measured results show the CSDA-MST core with a throughput rate of 1.28 G-pels/s, which can support $(4928 \times 2048@24$ Hz) digital cinema format with only 30 k logic gates. Because visual media technology has advanced rapidly, this approach will help meet the rising high-resolution specifications and future needs as well.

REFERENCES:


