Reconfigurable CORDIC-Based Low-Power DCT Architecture Based on Data Priority

Shaik Farheen Anjum[1],Mrs. Meenakshi[2]

[1]M.Tech Student (VLSI), Dept. of ECE, Gokaraju Rangaraju Institute of Engineering and Technology, Bachupally, Qutbullapur, Medchal,Telangana, India.

[2]Professor, Head of ECE Dept., Gokaraju Rangaraju Institute of Engineering and Technology, Bachupally, Qutbullapur, Medchal,Telangana, India.

Abstract— this paper presents a low-power coordinate rotation digital computer (CORDIC)-based reconfigurable discrete cosine transform (DCT) architecture. The main idea of this paper is based on the interesting fact that all the computations in DCT are not equally important in generating the frequency domain outputs. Considering the importance difference in the DCT coefficients, the number of CORDIC iterations can be dynamically changed to efficiently tradeoff image quality for power consumption. Thus, the computational energy can be significantly reduced without seriously compromising the image quality. The proposed CORDIC-based 2-D DCT architecture is implemented and experimental results show that our reconfigurable DCT achieves power savings ranging over the CORDIC-based Loeffler DCT at the cost of minor image quality degradations.

I INTRODUCTION

WITH THE explosive growth of multimedia services running on portable applications, the demand for lowpower implementations of complex signal processing algorithms is tremendously increasing. The most significant part of multimedia systems are the applications involving image and video processing, which are very computationally intensive and thus should be implemented with low cost because of the limited battery lifetime of portable devices. Many previous research efforts are focused on reducing power dissipation of image and video applications. Especially, low-power design of discrete cosine transform (DCT) has been of particular interest, since DCT is one of the most computationally intensive operations in video and image compression, and it is widely adopted in many standards such as JPEG, MPEG, and H.264.

Since first proposed in 1959 , coordinate rotation dig- ital computer (CORDIC) has been widely used to calculate the trigonometric functions in signal processing applications, such as QR decomposition, fast Fourier transform, singular value decomposition, and so on. Since CORDIC can be simply implemented with the iterative oper- actions of additions and shifts, it has been widely used for the multiplier less low-power DCT architectures

CORDIC (for Coordinate Rotation Digital Computer),also known as Volder's algorithm, is a simple and efficient algorithm to calculate hyperbolic and trigonometric functions, typically converging with one digit (or bit) per iteration. It is therefore also a prominent example of digit-by-digit algorithms. CORDIC and closely related methods known as pseudo-multiplication and pseudo-division or factor combining are commonly used when no hardware multiplier is available (e.g. in simple microcontrollers and FPGAs),

II EXISTING SYSTEM

A. CORDIC Architecture

The basic principal of CORDIC is to iteratively rotate a vector using a rotation matrix [8], which is represented as follows:

$$\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} x_{i-1} - \sigma_i 2^{1-i} y_{i-1} \\ y_{i-1} + \sigma_i 2^{1-i} x_{i-1} \\ z_{i-1} - \sigma_i \alpha_i \end{bmatrix} \quad (1)$$

In the CORDIC architecture, the amplitude and argument of a given vector can be calculated using the vectoring mode, while the sine and cosine values of the given angle are obtained with the rotation mode [28]. The hardware architecture of the CORDIC iteration is shown in Fig.1, which is referred as a crossing-architecture in the following.

1) Lookahead CORDIC Approach:

In the CORDIC equation shown in (1), to calculate the output of the current stage, the results from the previous stage iterations should be computed first.. An example of four-iteration step lookahead CORDIC [25]–[27] is shown in (2). It is noteworthy that if the signbits σk , (k = 1,..., 4) are known ahead, the following stage iterations can be directly computed using the input vectors of the present stage iteration without computing the intermediate results:

$$\begin{bmatrix} x_4 \\ y_4 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} -\sigma_1\sigma_2 2^0 \\ -\sigma_1\sigma_3 2^{-1} \\ -\sigma_1\sigma_4 2^{-2} \\ +\sigma_1\sigma_2\sigma_3\sigma_4 2^{-3} \end{bmatrix} & \begin{bmatrix} -\sigma_1 2^0 \\ +\sigma_1\sigma_2\sigma_3 2^{-1} \\ +\sigma_1\sigma_2\sigma_4 2^{-2} \\ +\sigma_1\sigma_3\sigma_4 2^{-3} \end{bmatrix} \\ \begin{bmatrix} +\sigma_1 2^0 \\ -\sigma_1\sigma_2\sigma_3 2^{-1} \\ -\sigma_1\sigma_2\sigma_4 2^{-2} \\ -\sigma_1\sigma_3\sigma_4 2^{-3} \end{bmatrix} & \begin{bmatrix} -\sigma_1\sigma_2 2^0 \\ -\sigma_1\sigma_3 2^{-1} \\ -\sigma_1\sigma_4 2^{-2} \\ +\sigma_1\sigma_2\sigma_3\sigma_4 2^{-3} \end{bmatrix} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad (2)$$

2) Scale-Factor in CORDIC Operations:

In the CORDIC operation, the magnitude of the rotated vector is scaled and accumulated after every iteration according to the following equation:

$$K_i = \frac{1}{\sqrt{1 + 2^{2(1-i)}}}. \quad (3)$$

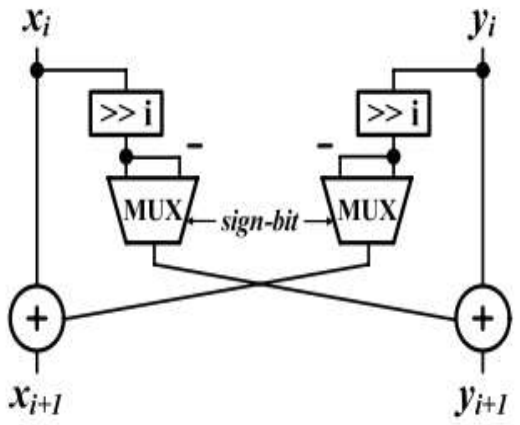


Fig.1. Hardware architecture of CORDIC iteration.

After a series of iterations, the accumulated Ki value in (3) is converged to a constant as follows:

$$K(n) = \prod_{i=1}^{n} K_i = \prod_{i=1}^{n} \frac{1}{\sqrt{1 + 2^{2(1-i)}}} \Rightarrow \lim_{n\to\infty} K(n) \approx 0.60725\ldots \quad (4)$$

where n is the number of iterations.

B. CORDIC-Based DCT Architecture

The 2-D DCT process is decomposed into an 1-D DCT (row DCT) followed by another 1-D DCT (column DCT), which is expressed as the following equation:

$$Y = TxT^T = T(Tx^T)^T \quad (5)$$

where x and Y are 8 × 8 size of image data matrix and 2-D DCT transformed output matrix, respectively. T is the 8 × 8 1-D DCT basis matrix. The 2-D DCT process with separable 1-D DCT is shown in Fig.2. The 8 × 8 1-D DCT transform is expressed as

$$X(k) = \frac{c(k)}{2} \sum_{i=0}^{7} x(i) \cos\left(\frac{(2i+1)k\pi}{16}\right)$$

where

$$k = 0, 1, 2, \ldots, 7$$

$$c(k) = \begin{cases} 1/\sqrt{2} & k = 0 \\ 1 & \text{otherwise} \end{cases} \quad (6)$$

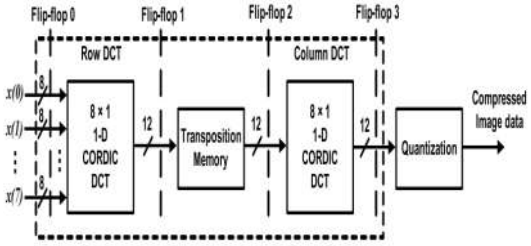


Fig. 2. 8 × 8 2-D DCT processor with separable 1-D DCT

The rearranged 1-D DCT equation is now represented as vector rotation matrix together with the consecutive CORDIC iterations as shown in Fig..3. Now, DCT can be implemented using only shifters and adders without multiplier [13]. Please note that the sign-bits and the scale-factor are known ahead since the input angles of CORDIC module are given as the DCT bases

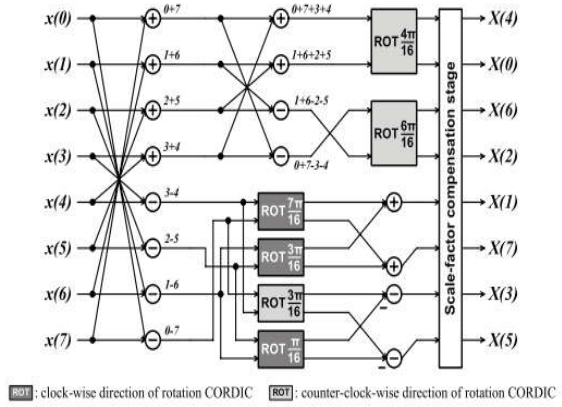


Fig.3. Hardware architecture of CORDIC-based 1-D DCT.

III PROPOSED SYSTEM

PRIORITY-BASED LOW-POWER DCT ARCHITECTURE USING LOOKAHEAD CORDIC APPROACH

A. Data Priority Considered Look ahead CORDIC Architecture

In the conventional CORDIC structure shown in Fig. 1, due to the crossing-data path, changing the number of iterations for two separate CORDIC data paths is not feasible. To assign different number of iterations to the two CORDIC data paths, we adopt the look ahead CORDIC approach in the proposed DCT architecture. As shown in (2), the three-step look ahead CORDIC can be expressed as follows:

$$\begin{bmatrix} x_3 \\ y_3 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} -\sigma_1\sigma_2 2^0 \\ -\sigma_1\sigma_3 2^{-1} \end{bmatrix} & \begin{bmatrix} -\sigma_1 2^0 \\ +\sigma_1\sigma_2\sigma_3 2^{-1} \end{bmatrix} \\ \begin{bmatrix} +\sigma_1 2^0 \\ -\sigma_1\sigma_2\sigma_3 2^{-1} \end{bmatrix} & \begin{bmatrix} -\sigma_1\sigma_2 2^0 \\ -\sigma_1\sigma_3 2^{-1} \end{bmatrix} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}. \quad (9)$$

Assuming that if the CORDIC results require four iterations for x whereas three iterations are needed for y, as shown in (2) and (9), the look ahead CORDIC equation for both results can be expressed as follows, which means that we can separately calculate the two CORDIC outputs:

$$\begin{bmatrix} x_4 \\ y_3 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} -\sigma_1\sigma_2 2^0 \\ -\sigma_1\sigma_3 2^{-1} \\ -\sigma_1\sigma_4 2^{-2} \\ +\sigma_1\sigma_2\sigma_3\sigma_4 2^{-3} \end{bmatrix} & \begin{bmatrix} -\sigma_1 2^0 \\ +\sigma_1\sigma_2\sigma_3 2^{-1} \\ +\sigma_1\sigma_2\sigma_4 2^{-2} \\ +\sigma_1\sigma_3\sigma_4 2^{-3} \end{bmatrix} \\ \begin{bmatrix} +\sigma_1 2^0 \\ -\sigma_1\sigma_2\sigma_3 2^{-1} \end{bmatrix} & \begin{bmatrix} -\sigma_1\sigma_2 2^0 \\ -\sigma_1\sigma_3 2^{-1} \end{bmatrix} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}. \quad (10)$$

Fig.4 presents the difference between the conventional crossing CORDIC architecture and the look ahead-based approach. Table I shows the iterations executed at ith stages and the corresponding rotation direction σ (sign-bits). For example, to rotate the vector by π/16, only the ith iterations (i = 0, 1, 3, 10) are executed and the rest of the iterations can be skipped for power savings. The look ahead algorithm for π/16 CORDIC rotator can be written as follows:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & -\sigma_{10} \cdot 2^{-10} \\ \sigma_{10} \cdot 2^{-10} & 1 \end{bmatrix} \begin{bmatrix} 1 & -\sigma_3 \cdot 2^{-3} \\ \sigma_3 \cdot 2^{-3} & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & -\sigma_1 \cdot 2^{-1} \\ \sigma_1 \cdot 2^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & -\sigma_0 \cdot 2^{0} \\ \sigma_0 \cdot 2^{0} & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad (11)$$

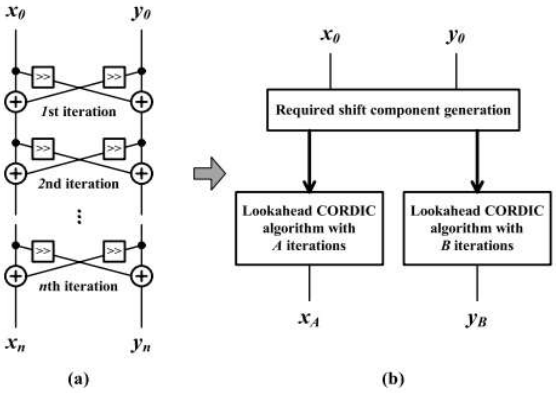


Fig.4. Differences between (a) crossing-architecture and (b) look ahead approach-based architecture of CORDIC module.

TABLE I

REQUIRED ITERATIONS AND DIRECTIONS FOR VECTOR ROTATION

(+: CLOCKWISE DIRECTION, *: COUNTER-CLOCKWISE DIRECTION)

Angle	Required Iterations	Directions (Sign-Bits)
$\pi/16^{+}$	$i = 0, 1, 3, 10$	$\sigma = -1, +1, +1, +1$
$3\pi/16^{+}$	$i = 1, 3, 10$	$\sigma = -1, -1, -1$
$3\pi/16^{*}$	$i = 1, 3, 10$	$\sigma = +1, +1, +1$
$4\pi/16^{*}$	$i = 0$	$\sigma = +1$
$6\pi/16^{*}$	$i = (90^{\circ}), 2, 3, 5, 7$	$\sigma = -1, +1, +1, +1, -1$
$7\pi/16^{+}$	$i = 0, 1, 3, 10$	$\sigma = -1, -1, -1, -1$

B. Proposed Low-Power CORDIC-Based DCT Architecture

As mentioned in the last part of Section III-A, considering the data priorities in DCT coefficient, high shift-term of the look ahead CORDIC can be carefully removed, which has the same effect with the less number of CORDIC iterations. Because the less number of CORDIC iterations means the CORDIC with low computational complexity, a low-power CORDIC-based DCT architecture can be derived and its detailed implementation is as follows. Fig. 3.2(a) shows the hardware architecture of the proposed CORDIC-based DCT. Inside the CORDIC module, the lookahead CORDIC is derived using the parameters in Table I.

The scale-factors are also specified in Table II. An example of the look ahead CORDIC algorithm for 7π/16 rotation and the corresponding scale-factors are presented in the equations shown in Fig. 5(b). To reduce the number of iterations, the high shift-terms are removed as presented in Section III-A, the implementation of which is specified in the solid lines of Fig. 5(b). We further reduce the less important components considering the data priorities in DCT coefficients. In Fig. 5(b), a CORDIC output, Kx , is more important than Ky as it is used later for X(1), whereas Ky is needed for the higher frequency component, X(7). Thus, the high shift-terms for y and Ky are further removed, which is expressed as the dotted lines in Fig. 5(b). In the proposed hardware architecture, all the shift components for each of look ahead CORDIC algorithm and the scale-factors are pre computed using the look ahead CORDIC equations. In Fig. 5(c), the numbers in the circle represent the shift operation, and the black color circle means the 2's complement elements of the shifted component, which are used for subtract operations. The dotted line in Fig. 5(c) represents the omitted computations, thus, the two results in look ahead CORDIC modules have the different number of terms, which leads to power savings owing to the smaller number of iterations

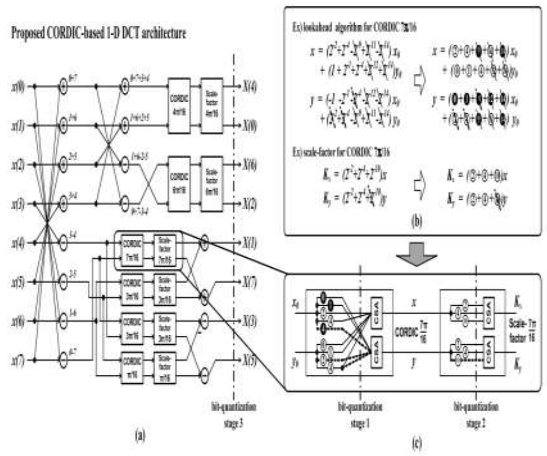


Fig. 5(a) Hardware architecture of the proposed low-power CO55RDIC-based 1-D DCT. (b) An example of look ahead CORDIC algorithm (7π/16) and (c) its hardware architecture.

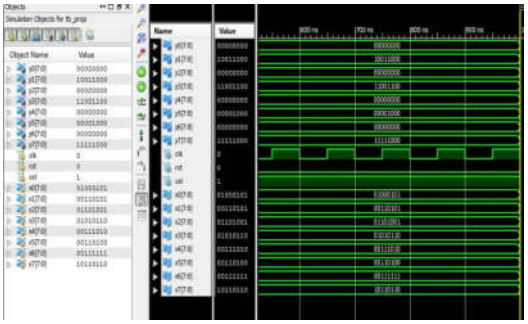
IV RESULTS

Proposed results:

Design summary:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	319	4656	6%
Number of Slice Flip Flops	66	9312	0%
Number of 4 input LUTs	585	9312	6%
Number of bonded IOBs	131	232	56%
Number of GCLKs	1	24	4%

Simulation results:



Extension results:

Design summary:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	234	4656	5%
Number of 4 input LUTs	421	9312	4%
Number of bonded IOBs	131	232	56%
Number of GCLKs	1	24	4%

Simulation results:

V CONCLUSION

In the conventional DCT architecture, all the computations are not equally important in generating the frequency domain outputs. This paper presented a low-power CORDIC based DCT architecture, where the importance differences in DCT coefficients were efficiently exploited to allocate the numbers of CORDIC iterations and internal data bit-widths. Look ahead CORDIC architectures were effectively used to get over the inherent data-dependencies in the conventional crossing-architecture of CORDIC. The proposed reconfigurable CORDIC-based DCT architecture can dynamically change the tradeoff modes with the power savings compared with the CORDIC-based Loeffler DCT architecture.

REFERENCES

[1] T. Liu, T. Lin, S. Wang, and C. Lee, "A low-power dual-mode video decoder for mobile applications," IEEE Commun. Mag., vol. 44, no. 8, pp. 119–126, Aug. 2006.

[2] M. Parlak and I. Hamzaoglu, "Low power H.264 deblocking filter hardware implementations," IEEE Trans. Consum.Electron., vol. 54, no. 2, pp. 808–816, May 2008.

[3] A. Bahari, T. Arslan, and A. T. Erdogan, "Low-power H.264 video compression architectures for mobile communication," IEEE Trans. Circuits Syst. Video Technol., vol. 19, no. 9, pp. 1251–1261, Sep. 2009.

[4] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," IEEE Trans. Comput., vol. 23, no. 1, pp. 90–93, Jan. 1974.

[5] G. K. Wallace, "The JPEG still picture compression standard," IEEE Trans. Consum. Electron., vol. 38, no. 1, pp. 18–34, Feb. 1992.

[6] D. L. Gall, "MPEG: A video compression standard for multimedia applications," Commun. ACM, vol. 34, no. 4, pp. 46–58, Apr. 1991.

[7] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 560–576, Jul. 2003.

[8] J. E. Volder, "The CORDIC trigonometric computing technique," IRE Trans. Electron. Comput., vol. 8, no. 3, pp. 330–334, Sep. 1959.

[9] A. Maltsev, V. Pestretsov, R. Maslennikov, and A. Khoryaev, "Triangular systolic array with reduced latency for QR-decomposition of complex matrices," in Proc. IEEE Int. Symp. Circuits Syst., May 2006, pp. 385–388.