A Fully Parallel Approximate CORDIC Design

Linbin Chen and Fabrizio Lombardi
Electrical and Computer Engineering Department
Northeastern University
Boston, USA
lombardi@ece.neu.edu

Jie Han
Electrical and Computer Engineering Department
University of Alberta
Edmonton, Canada
jhan8@ualberta.ca

Weiqiang Liu
College of Electronic Information and Engineering
Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China
liuweiqiang@nuaa.edu.cn

***Abstract*— This paper proposes a new approximate scheme for a coordinate rotation digital computer (CORDIC) design; this scheme is based on modifying the existing Para-CORDIC architecture with multiple approximations. These approximations make possible a relaxation of the CORDIC algorithm itself, such that a fully parallel approximate CORDIC (FPAX-CORDIC) scheme is designed. This scheme avoids the memory register of Para-CORDIC and makes fully parallel the generation of the rotation direction. A comprehensive analysis and the evaluation of the error introduced by the approximations together with different circuit-related metrics are pursued using HSPICE as simulation tool. The error analysis of this paper combines existing figures of merit for approximate computing (such as the Mean Error Distance (MED)) with CORDIC-specific parameters; a good agreement between expected and simulated error values is found. As an application to image processing, the Discrete Cosine Transformation (DCT) is investigated by utilizing the proposed approximate FPAX-CORDIC architecture with different accuracy requirements. The results confirm the viability of the proposed scheme.**

Index Terms*— *Inexact computing, CORDIC, Error distance, Power dissipation

I. Introduction

Modern computers rely heavily on fast arithmetic computation to solve complex problems with a high degree of accuracy. However, accurate hardware implementations for arithmetic processing often incur large overheads due to increase of circuit complexity, delay and power consumption. These overheads are more evident at the nanometric scales in which computation encounters physical limitations due to the reduced feature size. The paradigm of inexact computing relies on relaxing fully precise and completely deterministic computation to balance often contradicting figures of merit, such as power consumption and performance [1]. The tradeoffs that are available for inexact computing are very complex once arithmetic processing is considered at a higher level than just a single operation (such as addition or multiplication). Many inexact or approximate adders, multipliers and dividers have been proposed in the technical literature [2-4]; however, these designs are considered and compared often with respect to the implications of an approximation to the operation itself and its ability to deliver an output of acceptable accuracy (such as image processing or filtering) [5, 6].

This paper addresses a different scenario in which approximation is still implemented in hardware, but it is considered as part of an algorithm, namely for a coordinate rotation digital computer (CORDIC) implementation [7, 8].

CORDIC is an iterative algorithm for the calculation of the rotation of a 2-dimensional vector in different coordinate systems; the benefits of the CORDIC algorithm are that only additions and shift operations are employed. A hardware implementation of the CORDIC algorithm usually employs both a finite number of iterations and a finite level of precision. Therefore, the objective of this paper is to design, evaluate and assess an approximate CORDIC design. Multiple approximations are utilized, thus relaxing the CORDIC algorithm itself for a fully parallel execution. Power dissipation and accuracy can be reduced when properly selecting the parameters (such as the so-called error control parameter p) of the proposed scheme. A comprehensive evaluation of p and different circuit metrics including complexity and power dissipation are presented. An error analysis that combines traditional figures of merit (such as the Mean Error Distance (MED) [3]) with CORDIC specific parameters is analytically pursued. For computing the Discrete Cosine Transformation (DCT), the FPAX-CORDIC reduces power consumption while still generating computational results with very modest inaccuracy.

II. Review

The basic iterative equations of the CORDIC algorithm for radix-2 in circular coordinate systems are as follows:

$$\begin{aligned} x_{i+1} &= x_i - \sigma_i y_i 2^{-i}, \\ y_{i+1} &= y_i + \sigma_i x_i 2^{-i}, \\ \theta_{i+1} &= \theta_i - \sigma_i \alpha_i, \\ \sigma_i &\in \{-1, 1\};\ i = 1, 2, \ldots, n. \end{aligned} \tag{1}$$

$$K = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}}, \tag{2}$$

in which the direction of the iterative rotation $\sigma_i = sign(\theta_i)$ and K is the scaling constant that represents the increases in magnitude of the vector (x_i, y_i) in every iteration. The CORDIC algorithm can be used to compute trigonometric functions. For example, let $x_1 = K, y_1 = 0,$ and $\theta_1 = \theta$, the final output is given by $x_{N+1} = \cos(\theta), y_{N+1} = \sin(\theta)$.

Eq. (1) shows that the performance bottleneck is the sequential calculation of σ_i. The operations in each stage can be executed only after the corresponding stage has selected the correct rotation direction. If the direction is found in all stages and can be parallelized or pre-computed, the corresponding CORDIC rotations in the microrotation stage can also be concurrently executed. Different solutions have been proposed in the technical literature [9-13] for the parallel execution of σ_i through the θ-path. In [14], the so-called Para-CORDIC parallelizes the generation of the rotations direction *i.e.*, σ_i from the binary value of the input angle θ by employing a binary to bipolar representation (BBR) and a microrotation

angle recoding (MAR) technique. The two's complement N+1 bit binary representation of the input angle θ is given by $(-b_0) + \Sigma_{j=1}^{N} b_j 2^{-j}$, where $b_j \in \{0,1\}$ is assumed to be in the range $|\theta| \leq \pi/4$. The input angle θ is divided into the higher part θ^H and the lower part θ^L:

$$\begin{aligned}\theta &= \theta^L + \theta^H \\ &= (-b_0) + \Sigma_{j=1}^{l-1} b_j 2^{-j} + \Sigma_{j=l}^{N} b_j 2^{-j}\end{aligned} \tag{3}$$

In (3), l is the smallest index value such that $2^{-l} - \tan^{-1} 2^{-l} < 2^{-N}$. It has been shown in [15] that $l = \lceil (N - \log_2 3)/3 \rceil$. Next, a brief treatment of the BBR and MAR methods to predict the σ_i for θ^L and θ^H respectively, is provided.

1) Binary to Bipolar Representation (BBR)

The BBR method converts the first l-1 bits of the input angle (*i.e.*, θ^L) and obtains the corresponding rotation directions (σ_1 *to* σ_l). The binary value $b_j \in \{0,1\}$ is converted to the corresponding bipolar representation $r_k \in \{-1,1\}$ as follows:

$$\begin{aligned}\theta^L &= (-b_0) + \Sigma_{j=1}^{l-1} b_j 2^{-j} \\ &= (-b_0) + \Sigma_{j=1}^{l-1} [2^{-j-1} + (2b_j - 1)2^{-j-1}] \\ &= \Sigma_{i=1}^{l} r_i 2^{-i} - 2^{-l},\end{aligned} \tag{4}$$

where,

$$\begin{aligned} r_1 &= 1 - 2b_0 \\ r_i &= 2b_{i-1} - 1, i = 2, 3, \dots, l \end{aligned}$$

The first l rotation direction (σ_1 *to* σ_l) is directly derived from (4) and the bipolar value of r_1 to r_l. Then, θ^L is written as follows:

$$\begin{aligned}\theta^L &= \Sigma_{i=1}^{l} r_i 2^{-i} - 2^{-l} \\ &= \Sigma_{(i=1)}^{l} \sigma_i \left\{ \Sigma_{j=1}^{n(i)} \tan^{-1}\left(2^{-s_i^j}\right) + e_i \right\} - 2^{-l},\end{aligned} \tag{5}$$

where, $\sigma_i = r_i$; $i = 1, 2, \dots, l$. In (5), 2^{-i} is expressed as the sum of arctangent values and an error term, *i.e.*, $\Sigma_{j=1}^{n(i)} \tan^{-1}\left(2^{-s_i^j}\right) + e_i$ (as discussed next in the MAR algorithm).

2) Microrotation Angle Recoding (MAR)

The decomposition of each positional binary weighting $2^{-i}, i = 1, 2, \dots, l-1$ into a combination of arctangent terms and a nonnegative error term e_i, yields the following expression:

$$\begin{aligned}2^{-i} &= \tan^{-1}(2^{-i}) + \Sigma_{j=2}^{n(i)} \tan^{-1}\left(2^{-s_i^j}\right) \\ &= \Sigma_{j=1}^{n(i)} \tan^{-1}\left(2^{-s_i^j}\right) + e_i,\end{aligned} \tag{6}$$

where, $s_i^1 = i,\ i = 1, 2, \dots, l-1$.

The above equation is generally known as MAR. $n(i)$ is the number of microrotations required in the MAR recording of 2^{-i}, and s_i^j is the shift sequences for $j = 1, 2, \dots, n(i)$ with the first shift $s_i^1 = s_i = i, i = 1, 2, \dots, l-1$. The detailed algorithm of MAR recording can be found in [14]. By combining (3) and (5-6), the corrected rotation angle $\hat{\theta}^H$ is given by

$$\hat{\theta}^H = \theta^H + \Sigma_{i=1}^{l-1} \sigma_i e_i - 2^{-l}. \tag{7}$$

Following the BBR for $\theta^L$and the MAR for each binary positional weight 2^{-i}, another BBR is applied to the binary representation of the corrected $\hat{\theta}^H$ as follows:

$$\begin{aligned}\hat{\theta}^H &= (-\hat{b}_{l-1})2^{-l+1} + \Sigma_{k=l}^{N} \hat{b}_k 2^{-k} \\ &= (-\hat{b}_{l-1}) + \Sigma_{k=l+1}^{N+1} \left[(2\hat{b}_{k-1} - 1)2^{k+2^{-l}} + 2^{-N-1} \right] \\ &= \Sigma_{i=l}^{N+1} \hat{r}_i 2^{-i} - 2^{-N-1},\end{aligned} \tag{8}$$

where,

$$\begin{aligned} \hat{r}_i &= 1 - 2\hat{b}_{i-1}, && i = l \\ \hat{r}_i &= (2\hat{b}_{i-1} - 1), && i = l+1, \dots, N+1 \end{aligned}$$

From (8) the last N-l+2 rotation direction ($\hat{\sigma}_l$ *to* $\hat{\sigma}_{N+1}$) is found directly from the bipolar value of $\hat{r}_l$ to $\hat{r}_{N+1}$.

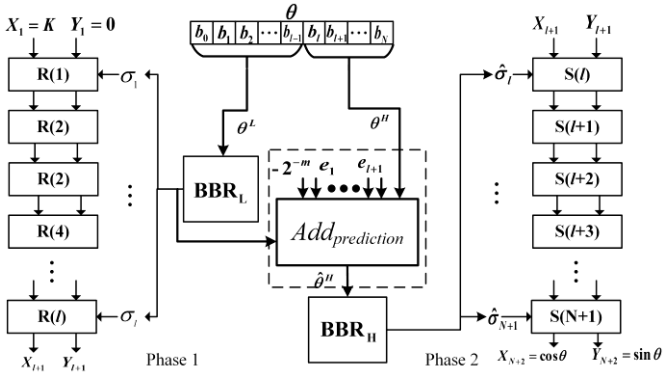


Fig. 1 The Para-CORDIC Architecture [14].

3) Para-CORDIC architecture

The Para-CORDIC rotation architecture is shown in Fig. 1. There are two BBRs in the para-CORDIC rotation. In Fig. 1, BBR_L and BBR_H are used to represent the operations in (4) and (8) *i.e.*, determine the rotation direction σ_1 to σ_l and $\hat{\sigma}_l$ to $\hat{\sigma}_{N+1}$, respectively. The operations in (7) are denoted as $Add_{prediction}$ (Fig. 1). The errors given by the e_i terms, are precomputed and stored in the memory register.

III. Proposed Fully Parallel Approximate CORDIC (FPAX-CORDIC)

A. FPAX-CORDIC algorithm and architecture

Although Para-CORDIC parallelizes the computation of σ_i in two phases (when calculating BBR_L and BBR_H), this algorithm cannot still independently execute the relation between Phase 1 and Phase 2. The generation of the rotation direction σ_i is still not fully parallel, *i.e.*, the relation between Phase 1 and Phase 2 must be further analyzed to allow a fully parallel generation of σ_i. Therefore, an approximate design is proposed for implementation to meet this requirement.

The relationship between Phase 1 and Phase 2 is considered next with respect to the error compensation mechanism in Para-CORDIC; the basic criterion consists of assessing the error following BBR in Phase 1. If the generated error can be tolerated, then this relationship can be completely eliminated to allow a fully parallel execution of BBR_L and BBR_H. Also in this latter case, there is no need of additional memory to store the error compensation terms e_i; moreover, this condition makes the operation of the circuit fully combinational, hence also improving its performance.

Consider Fig. 1, the block $Add_{prediction}$ is eliminated (denoted now as a dashed rectangle) as per the following analysis. The main function of the $Add_{prediction}$ block is to compensate the error introduced by the first BBR operation in

(7). In MAR for $2^{-i}, i = 1, 2, \dots, l-1$, the value of $\hat{\theta}^H$ must satisfy the following constraint:

$$\left|\hat{\theta}^H\right| < 2^{-(l-1)} \tag{9}$$

If this condition is met, then each binary weighting of 2^{-i} in the remaining angle can be approximated by $\tan^{-1}(2^{-i})$ within the accuracy allowed by the *N* fractional bits. As outlined in [14], if $\Sigma_{i=1}^{l-1} e_i < 2^{-l}$, then the value of $\left|\hat{\theta}^H\right|$ must satisfy the inequality of (9). 2^{-l} is the upper bound for $\Sigma_{i=1}^{l-1} e_i$; the higher order terms $\tan^{-1}(2^q), q > i$ in MAR of 2^{-i} can be found to allow $\Sigma_{i=1}^{l-1} e_i < 2^{-l}$. Thus, the number of microrotations (with the shift sequences $s_i^j, j = 1, 2, \dots, n(i)$) is directly controlled by $\Sigma_{i=1}^{l-1} e_i$, *i.e.*, the smaller $\Sigma_{i=1}^{l-1} e_i$ is, the larger $n(i)$ is [14]. The complete elimination of the $Add_{predition}$ block in Fig. 1 (so making CORDIC fully parallel) permits the inequality $\Sigma_{i=1}^{l-1} e_i < 2^{-N}$ to be satisfied. As shown in (7), if $\Sigma_{i=1}^{l-1} \sigma_i e_i = 0$, then $\hat{\theta}^H = \theta^H - 2^{-l}$ and error compensation is not required, *i.e.*, $Add_{prediction}$ is not required.

In general $\Sigma_{i=1}^{l-1} \sigma_i e_i = 0$ is not applicable, because it is only possible to make $\Sigma_{i=1}^{l-1} \sigma_i e_i$ approximately equal to 0. Let $\theta^E = \Sigma_{i=1}^{l-1} \sigma_i e_i$; as each e_i is not negative in MAR, then $|\theta^E| = \left|\Sigma_{i=1}^{l-1} \sigma_i e_i\right| \le \Sigma_{i=1}^{l-1} e_i$. So, let $\Sigma_{i=1}^{l-1} e_i \to 0$, hence $\theta^E \to 0$. For an input angle θ with an *N*-bit precision, $\Sigma_{i=1}^{l-1} e_i < 2^{-N}$; therefore, the error θ^E can be ignored for a *N*-bit precision.

An algorithm to find the high-order terms $\tan^{-1}(2^q), q > i$ and make $\Sigma_{i=1}^{l-1} e_i < 2^{-N}$ is rather intuitive, because it is rather similar to MAR [14]. However to meet the condition $2^{-N} \ll 2^{-l}$, the execution of an error free fully parallel CORDIC is significantly more restrictive than in Para-CORDIC. Therefore, the number of microrotation $n(i)$ in this case is more than in [14], *i.e.*, the complexity of the X/Y path is higher. When the θ error term $|\theta^E| < 2^{-p}$ is tolerated for a specific application, then $|\theta^E| \le \Sigma_{i=1}^{l-1} e_i < 2^{-p}$ where $p \in [l, \dots, N]$ is the so-called *error-tolerant parameter*. The proposed FPAX-CORDIC algorithm is presented in pseudo-code in Fig. 2.

For an N-bit input angle θ,
//Initial Values
Find $l = \lceil (N - log_2 3)/3 \rceil$
Perform Eq. (3) and $\theta^H - 2^{-l}$
Use the proposed theorem to find s_i^j *and* e_i $(\Sigma_{i=1}^{l-1} e_i < 2^{-p})$.
// Full Parallel Execution
Perform BBR (for the full range of the N-bit input angle θ)
From stage 1 to stage l
Perform R(i) using $\sigma_i = r_i, i = 1 \dots l$.
From stage (l+1) to stage (N+2)
Perform S(i) using $\sigma_i = r_l, i = l \dots N+1$

Fig. 2 FPAX-CORDIC Algorithm.

The architecture of the proposed FPAX-CORDIC is shown in Fig. 3. The two paths, namely the θ-path and the X/Y path, are discussed as follows.

θ-path: Consider Fig. 1 and Fig. 3; the operations of the data path in the conventional CORDIC are replaced by the BBR in the proposed FPAX-CORDIC. The delay and the hardware overhead of the two BBRs are negligible, because they perform a simple signal mapping and are parallel with X/Y-path (*i.e.*, 0 (1) is considered, as the subtraction (addition) signal).

X/Y-path: In the proposed architecture (Fig. 3), the number of total microrotations is denoted as $Rot(N) = \Sigma_{i=1}^{l-1} n(i) + N - l + 3$. $n(i)$ is directly related to $\Sigma_{i=1}^{l-1} e_i$, and therefore, it is also related to the error-tolerant parameter p. So, the area and delay are $\big(4N \times Rot(N)\big)A_{FA}$ and $\big(2 \times Rot(N)\big)T_{FA}$, i.e. it is assumed that each microrotation stage is implemented using a Binary Signed Digit Adder (BSDA) and the area and delay with a word size of *N* bits are $4N \times A_{FA}$ and $2T_{FA}$. A_{FA} and T_{FA} are the area and delay of a full adder.

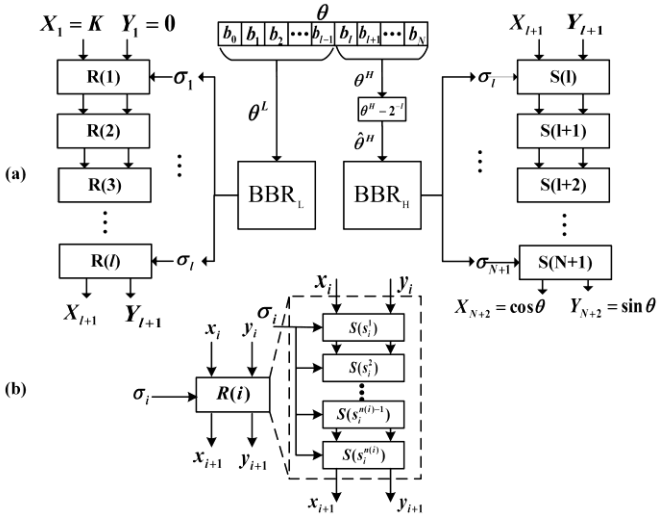


Fig. 3 (a) Architecture of proposed FPAX-CORDIC (b) Structure of R(i)

B. *Error Analysis of FPAX-CORDIC*

The approximation error of FPAX-CORDIC with respect to the real trigonometric function is controlled by the error-tolerant parameter *p*. As $|\theta^E| < 2^{-p}$ is ignored in θ, then

$$\sin\theta' = \sin(\theta - \theta^E) \tag{10}$$

Thus,

$$\begin{aligned} E_{sin}^{AX}(\theta, \theta^E) &= \sin\theta - \sin\theta' \\ &= (1 - \cos\theta^E)\sin\theta + \sin\theta^E \cos\theta \\ E_{cos}^{AX}(\theta, \theta^E) &= \cos\theta - \cos\theta' \\ &= (1 - \cos\theta^E)\cos\theta - \sin\theta^E \sin\theta \end{aligned} \tag{11}$$

If the rounding error is not considered, then the MED [3] can be derived by the integral of the approximation error for $\theta \in [0, 2\pi)$ as,

$$\begin{aligned} \text{MED} &= \int_0^{2\pi} E_{sin}^{AX}(\theta, \theta^E) d\theta = \int_0^{2\pi} E_{cos}^{AX}(\theta, \theta^E) d\theta \\ &= \int_0^{2\pi} |(1 - \cos\theta^E)\sin\theta + \sin\theta^E \cos\theta| d\theta \\ &= \begin{cases} \dfrac{\sin\theta^E - \cos\theta^E + 1}{\pi}, & E_{sin}^{AX}(\theta, \theta^E) \ge 0 \\ \dfrac{\cos\theta^E - \sin\theta^E - 1}{\pi}, & E_{sin}^{AX}(\theta, \theta^E) < 0 \end{cases} \end{aligned} \tag{12}$$

IV. EVALUATION

A 16-bit FPAX-CORDIC is implemented and simulated using HSPICE with the 45nm predictive technology model (PTM); the input θ uses 17 bit 2's complementary binary values to represent the angle in radian. The angle information is in the format U(1, 16), where bit 16 is the sign bit and the

bits [15:0] are the fractional parts. X/Y use a 17-bit 2's complementary binary number representation; its format is given by U(1, 16) inclusive of the sign bit. The BBR block are implemented using multiplexers and inverters. As σ_i is fixed a-priori, the X/Y path can be realized using a Binary Signed Digit Adder (BSDA), i.e. addition without propagation of a carry. The intermediate rotation results are represented using a Binary Signed-Digit (BSD) numbering system [16].

The simulated error $E_{sin}(\theta, \theta^E)$ versus p for the proposed 16-bit FPAX-CORDIC is plotted in Fig. 4. The black solid line is the mean error over the range of $p \in [5, 15]$; as p increases, the output error variation decreases to reach nearly zero. Eq. (12) has been plotted against the simulated results. Fig. 5 shows that the error equation provides a good estimate of the MED for the FPAX-CORDIX architecture, i.e. the simulated MED of the 16-bit FPAX-CORDIC as function of p. The MED drops rather fast at $p = 8$ and is nearly constant at $p > 10$; however, the simulated MED value is different from the estimated value. This is caused by the quantization of the rounding error at a less significant bit.

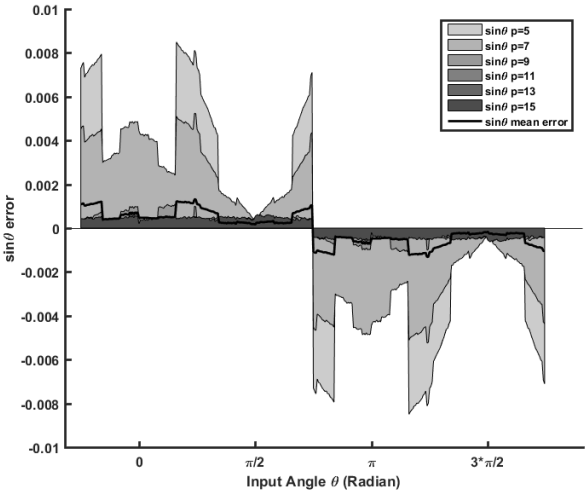


Fig. 4 Error of 16-bit FPAX-CORDIC versus parameter *p*.

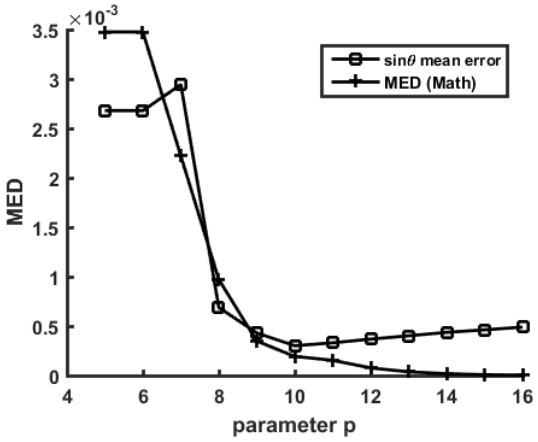
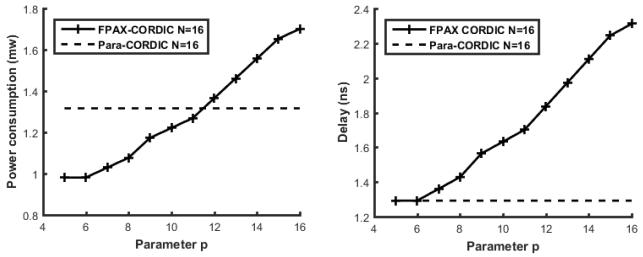


Fig. 5 MED of 16-bit FPAX-CORDIC versus parameter *p*.

The power and delay results are plotted in Fig. 6. When p<11, the power consumption of the FPAX-CORDIC is smaller than the Para-CORDIC. Because the value of p for FPAX-CORDIC is always larger than l, there is a larger number of microrotation stages in FPAX-CORDIC than in Para-CORDIC; thus the delay is longer than that of Para-CORDIC. The delay disadvantage can be addressed by ignoring (or truncating) some of the final rotation stages $S(i)$ (i=l, … N+1), when a slightly larger error can be tolerated in FPAX-CORDIC.

An approximate arithmetic design always has to deal with a trade-off between accuracy and power. As shown in Fig. 7, when p is changed, the power increases and the MED decreases. An abrupt decrease in the MED occurs when p=8 (and the power increases to 1.43 mw). Thus, p=8 is the most appropriate value for attaining low power and high accuracy design for a 16-bit design.



(a) (b)

Fig. 6 (a) Power consumption (b) Delay of 16-bit FPAX-CORDIC vs. parameter *p*.

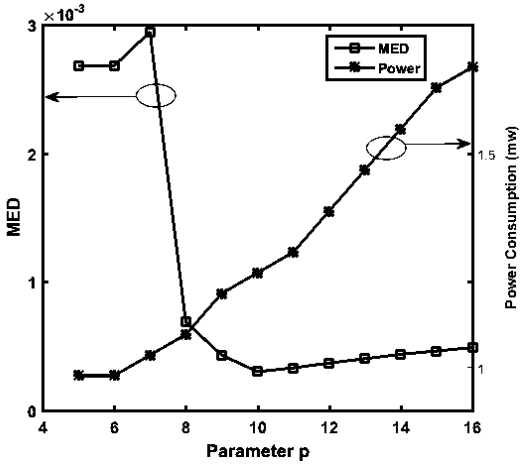


Fig. 7 Power and MED for 16-bit FPAX-CORDIC.

V. Application: Discrete Cosine Transform (DCT)

A wide range of algorithms (such as image enhancement in the spatial domain, frequency transform, image rotation, edge detection) can be implemented using CORDIC [17]. A discrete cosine transform (DCT) [18] expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies. Therefore, the focus of this section is on the 2-D 8×8 DCT, as widely used in image compression applications. As shown in Fig. 8, the source image (Lena) is divided into smaller 8×8 blocks; the DCT computation is applied to each 8×8 block. Throughout this section, the FPAX-CORDIC architecture uses a 17-bit U(1, 16) for the θ-path, 32-bit U(16, 16) for the X/Y-path (a real number consisting of 16 bits each for integer and fractional parts; each input image pixel is in the range [0, 255]).

Inexact 2-D 8×8 DCT based on FPAX-CORDIC: The DCT computation is implemented using the proposed FPAX-

CORDIC; the 2-D DCT is decomposed into a 1-D DCT (row-wise DCT) followed by another 1-D DCT (column-wise DCT). The process with the separable 1-D DCTs is shown in Fig. 9; the resulting DCT image is compared against the results of Para-CORDIC.

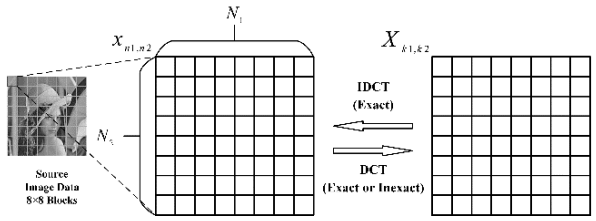


Fig. 8 2D DCT-IDCT process for image compression application.

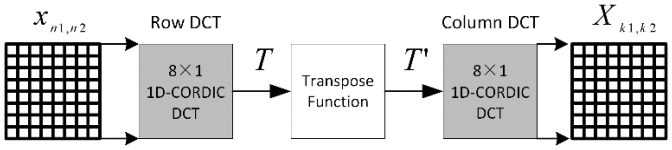


Fig. 9 Process for 2-D 8×8 DCT.

The 8×1 1-D DCT transform is given by [19]

$$X(k) = \frac{c(k)}{2}\sum_{i=0}^{7} x(i)\cos\frac{(2i+1)k\pi}{16} \quad (13)$$

where

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

$$c(k) = \begin{cases} \frac{1}{\sqrt{2}}, & k = 0 \\ 1, & k > 0 \end{cases}$$

By (13) and using the DCT and the trigonometric symmetric property, the 8×1 1-D DCT is as follows in matrix form:

$$\begin{bmatrix} X(4) \\ X(0) \end{bmatrix} = \frac{1}{2}\begin{bmatrix} c_4 & -s_4 \\ s_4 & c_4 \end{bmatrix}\begin{bmatrix} x(0)+x(7)+x(3)+x(4) \\ x(1)+x(6)+x(2)+x(5) \end{bmatrix}$$
$$\begin{bmatrix} X(6) \\ X(2) \end{bmatrix} = \frac{1}{2}\begin{bmatrix} c_6 & -s_6 \\ s_6 & c_6 \end{bmatrix}\begin{bmatrix} x(0)+x(7)-x(3)-x(4) \\ x(1)+x(6)-x(2)-x(5) \end{bmatrix}$$
$$\begin{bmatrix} X(1) \\ X(7) \end{bmatrix} = \frac{1}{2}\begin{bmatrix} c_7 & s_7 \\ -s_7 & c_7 \end{bmatrix}\begin{bmatrix} x(3)-x(4) \\ x(0)-x(7) \end{bmatrix} + \frac{1}{2}\begin{bmatrix} c_3 & s_3 \\ -s_3 & c_3 \end{bmatrix}\begin{bmatrix} x(1)-x(6) \\ x(2)-x(5) \end{bmatrix} \quad (14)$$
$$\begin{bmatrix} X(3) \\ X(5) \end{bmatrix} = \frac{1}{2}\begin{bmatrix} c_3 & -s_3 \\ s_3 & c_3 \end{bmatrix}\begin{bmatrix} x(0)-x(7) \\ x(3)-x(4) \end{bmatrix} - \frac{1}{2}\begin{bmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{bmatrix}\begin{bmatrix} x(2)-x(5) \\ x(1)-x(6) \end{bmatrix}$$

where $c_k = cos\left(\frac{k\pi}{16}\right) = s_m = \sin\left(\frac{m\pi}{16}\right), m = 8-k.$

The implementation of (14) for an 8×1 1D-CORDIC DCT is shown in Fig. 10. Six fixed-angle FPAX-CORDICs are used to complete the multiplication of the trigonometric terms in (14), because for an input $\theta \in [\frac{\pi}{4}, -\frac{\pi}{4}]$ to the FPAX-CORDIC design, all rotation angles are converted to this restricted range. In the architecture of the 8×8 DCT, the parameter p controls the accuracy of the FPAX-CORDIC modules. Thus, the accuracy of the inexact 2-D 8×8 DCT is affected in two respects: (1) for each 8×1 1D-CORDIC DCT (Fig. 10), six parallel FPAX-CORDIC modules can be configured by changing p; (2) the two 8×1 1D-CORDIC DCT are connected in series (Fig. 9), but they can be configured using different values of p. For simplicity, the value of p is kept constant for all FPAX-CORDIC modules in the following analysis.

Configurations of 8×8 DCT based on FPAX-CORDIC: Two features are considered for the six FPAX-CORDIC modules to further reduce the complexity of the proposed architecture for an 8×8 DCT and attain a better DCT accuracy.

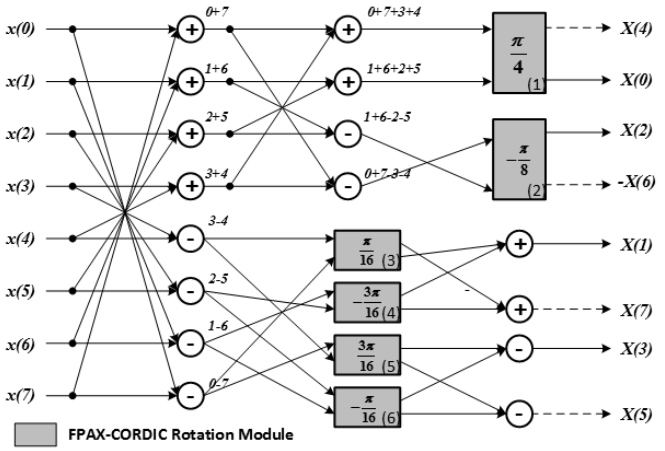


Fig. 10 8×1 1D-CORDIC DCT.

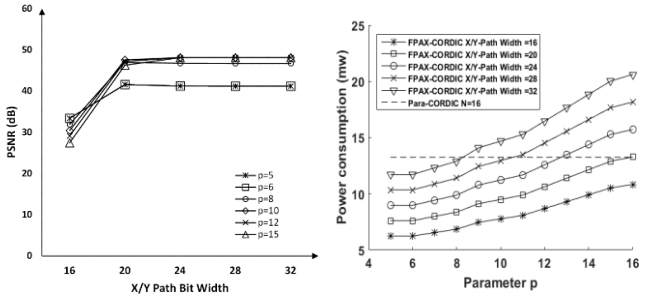
Non-equal precision path: When considering the reconstruction of the original input image $x(i)$ from the DCT compressed image $X(k)$, the high frequency components of $X(k)$ (for example $X(4)\sim X(7)$) have a small impact on the reconstructed image x(i). So, the X/Y path of the six FPAX-CORDIC modules in the 1D-CORDIC architecture can have non-equal precision. For example, the X-path of the FPAX-CORDIC modules (1) (3) and the Y-path of the FPAX-CORDIC modules (2) (4) (5) (6) can either have a poor precision (i.e., a smaller number of bits), or simply be ignored. The high frequency or non-critical outputs are shown in Fig. 10 with dashed arrows.

p sets: On the basis of unequal precision paths to further control the error for the low-frequency or critical outputs, the value of p for each individual FPAX-CORDIC module can be adjusted to meet the error specifications of an application. In each FPAX-CORDIC module, the output usually consists of a low-high frequency component pair. For example, for module (1), the output pair is $(X(4), X(0))$; as the accuracy of $X(4)$ is reduced, p affects solely $X(0)$. Similar conditions may also apply to the other five FPAX-CORDIC modules.

Image test: Fig. 11 shows an example of the simulated error impact of different configurations of FPAX-CORDIC in an inexact DCT application. The original image $x(i,j)$ is transformed using an FPAX-CORDIC based 2-D DCT architecture to generate an inexact DCT $\hat{X}(i,j)$, then the original $\hat{x}(i,j)$ is recovered using an exact IDCT transformation. The peak signal-to-noise ratio (PSNR) of $\hat{x}(i,j)$ against $x(i,j)$ is measured and the results are plotted in Fig. 12(a). The PSNR shows that higher are the values of p and the Y-path bits, the higher the PSNR is; the PSNR reaches a constant value when the number of bits in the X/Y path reaches 20. When p>8, the PSNR increases rapidly and reaches the largest value.

Fig. 11 Evaluation of the error of FPAX-CORDIC based inexact DCT

Hardware implementation of DCT: For the DCT application presented previously, FPAX-CORDIC at a specific p value has been described in a previous section. For the non-equal precision path introduced solely for the DCT application, the bit-width of the data path is controlled by a Dynamic Bit-width Control (DBC) circuit [20]. In this scheme, the hardware overhead consists of the pull-up and pull-down turn-off gate transistors for each X/Y-path bit in the positional logic. The power dissipation of a single DCT is measured and the results are plotted in Fig. 12(b). By considering the PSNR (Fig. 12(a)), a better accuracy is achieved at higher power consumption; a larger error has to be tolerated for a lower power dissipation. To achieve both low power and high accuracy in DCT, the scheme with p=8 and an X/Y-path width of 24 offers the best combined performance.



(a) (b)

Fig. 12 (a) PSNR (b) Power for inexact DCT application

VI. CONCLUSION

This paper has proposed an approach for an approximate CORDIC design. A fully parallel approximate CORDIC (FPAX-CORDIC) scheme has been proposed by modifying the Para-CORDIC architecture. The proposed approximate CORDIC does not utilize the memory register of Para-CORDIC and makes the generation of the rotation direction fully parallel. Power dissipation can be reduced when properly selecting the parameters in the proposed scheme. A comprehensive evaluation of the error control parameter *p* and different circuit metrics including complexity and power dissipation has been presented. An error analysis that combines traditional figures of merit (such as MED) with CORDIC specific parameters has been analytically pursued. Therefore, the following conclusive evidence is applicable: (a) The error parameter *p* plays a critical role when assessing the trade-off between power consumption and accuracy for the proposed FPAX-CORDIC. As *p* increases (i.e., when the number of bits to be processed increases) more accuracy is obtained, *i.e.*, the MED decreases to reach a nearly constant value. (b) The power dissipation of FPAX-CORDIC increases as a function of *p*, *i.e.*, for a 16-bit design, the proposed FPAX-CORDIC reaches the best trade-off between power dissipation and accuracy at *p*=8. (c) For image compression and decompression using DCT, additional power saving can be gained by tolerating more errors for the high frequency components as well as selecting *p*. Additional power saving can be realized by using the DBC circuit on the Y-path of each FPAX-CORDIC module.

REFERENCES

[1] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *Proc. 18th IEEE European Test Symposium (ETS)*, 2013, pp. 1-6.

[2] H. Jiang, J. Han, and F. Lombardi, "A Comparative Review and Evaluation of Approximate Adders," in *Proc. 25th Great Lakes Symposium on VLSI*, Pittsburgh, Pennsylvania, USA, 2015, pp. 343-348.

[3] J. Liang, J. Han, and F. Lombardi, "New Metrics for the Reliability of Approximate and Probabilistic Adders," *IEEE Trans. on Computers*, vol. 62, pp. 1760-1771, 2013.

[4] M. J. Schulte and E. E. Swartzlander, "Truncated multiplication with correction constant," *VLSI Signal Processing VI*, pp. 388-396, 1993.

[5] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-Power Digital Signal Processing Using Approximate Adders," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, pp. 124-137, 2013.

[6] A. Sampson, L. Ceze, and D. Grossman, "Good-enough computing," *IEEE Spectrum*, vol. 50, pp. 54-59, 2013.

[7] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. on Electronic Computers*, vol. EC-8, pp. 330-334, 1959.

[8] J. S. Walther, "A unified algorithm for elementary functions," in *Proc. Spring Joint Computer Conference*, Atlantic City, New Jersey, 1971, pp. 379-385.

[9] D. Timmermann, H. Hahn, and B. J. Hosticka, "Low latency time CORDIC algorithms," *IEEE Trans. on Computers*, vol. 41, pp. 1010-1015, 1992.

[10] T. Srikanthan and B. Gisuthan, "A novel technique for eliminating iterative based computation of polarity of micro-rotations in CORDIC based sine–cosine generators," *Microprocessors and Microsystems*, vol. 26, pp. 243-252, 2002.

[11] B. Gisuthan and T. Srikanthan, "Pipelining flat CORDIC based trigonometric function generators," *Microelectronics Journal*, vol. 33, pp. 77-89, 2002.

[12] H. S. Kebbati, J. P. Blonde, and F. Braun, "A new semi-flat architecture for high speed and reduced area CORDIC chip," *Microelectronics Journal*, vol. 37, pp. 181-187, 2006.

[13] B. Lakshmi and A. S. Dhar, "CORDIC Architectures: A Survey," *VLSI Design*, 2010.

[14] T.-B. Juang, S.-F. Hsiao, and M.-Y. Tsai, "Para-CORDIC: parallel CORDIC rotation algorithm," *IEEE Trans. on Circuits and Systems I: Regular Papers*, vol. 51, pp. 1515-1524, 2004.

[15] S. Wang, V. Piuri, and E. E. Swartzlander, Jr., "Hybrid CORDIC algorithms," *IEEE Trans. on Computers*, vol. 46, pp. 1202-1207, 1997.

[16] A. Avizienis, "Signed-Digit Number Representations for Fast Parallel Arithmetic," *IRE Transactions on Electronic Computers*, vol. EC-10, pp. 389-400, 1961.

[17] S. Sathyanarayana, R. K. Satzoda, and S. Thambipillai, "Unified Cordic Based Processor for Image Processing," in *Proc. 15th International Conference on Digital Signal Processing*, 2007, pp. 343-346.

[18] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete Cosine Transform," *IEEE Trans. on Computers*, vol. C-23, pp. 90-93, 1974.

[19] M.-W. Lee, J.-H. Yoon, and J. Park, "Reconfigurable CORDIC-Based Low-Power DCT Architecture Based on Data Priority," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 22, pp. 1060-1068, 2014.

[20] J. Park, J. Choi, and K. Roy, "Dynamic Bit-Width Adaptation in DCT: An Approach to Trade Off Image Quality and Computation Energy," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 18, pp. 787-793, 2010.