Squaring Technique using Vedic Mathematics

Jasmine Bajaj* and Babita Jajodia

Department of Electronics and Communication Engineering, Indian Institute of Information Technology Guwahati, Guwahati, India

* Corresponding author

doi:https://doi.org/10.21467/proceedings.114.75

ABSTRACT

Vedic Mathematics provides an interesting approach to modern computing applications by offering an edge of time and space complexities over conventional techniques. Vedic Mathematics consists of sixteen sutras and thirteen sub-sutras, to calculate problems revolving around arithmetic, algebra, geometry, calculus and conics. These sutras are specific to the decimal number system, but this can be easily applied to binary computations. This paper presented an optimised squaring technique using Karatsuba-Ofman Algorithm, and without the use of Duplex property for reduced algorithmic complexity. This work also attempts Taylor Series approximation of basic trigonometric and inverse trigonometric series. The advantage of this proposed power series approximation technique is that it provides a lower absolute mean error difference in comparison to previously existing approximation techniques.

Keywords: COordinate Rotation DIgital Computer (CORDIC), Duplex Property, Maclaurin Series, Squaring, Trigonometric Functions, Taylor Series, Vedic Mathematics.

I. Introduction

Trigonometric functions are one of the fundamental, yet the most essential core functions in the domain of digital signal processing, digital image processing and high-performance computing applications [1], [2]. This, in turn, demands efficient and optimized hardware implementations. Efficient hardware implementation of trigonometric and inverse trigonometric functions can be employed by using COordinate Rotation DIgital Computer (CORDIC) [3], or Look-Up Table method or power-series method [4]. Among the three methods, CORDIC implementation is the simplest using shift-and-add operations only, but the major disadvantage in it is that it requires a minimum of n iterations which provides high latency and low speed [3]; whereas the look-up table method has a drawback of higher area [4]. But, the power series method provides higher speed system compared to other two methods, but this require an optimized and efficient squaring/multiplication operation and these can be achieved using Vedic Mathematics [5]. Vedic Mathematics consists of sixteen sutras and thirteen sub-sutras and these can be easily applied to binary computations although they are specific to decimal number system [5]. Several researchers have reported Vedic squarers [6]–[12] using the Dwandwa Yoga Duplex property of Urdhwa Tiryagbhyam sutra and Anurupyena sutra of Vedic mathematics [5].

The authors in [4] presented the power series approximation method for calculation of trigonometric functions using the duplex property of binary numbers for squaring operations; but at the cost of algorithm complexity and higher error relative to actual trigonometric functions. This confirms that the use of Duplex property increases algorithm complexity for calculation of trigonometric functions via power series method. In this work, the authors have put an attempt to do power series approximation of trigonometric and inverse trigonometric functions. This paper proposes squaring technique using Vedic Mathematics based on Karatsuba-Ofman Algorithm [4] and without the use of Duplex property for reducing algorithmic complexity.

Proceedings DOI: 10.21467/proceedings.114; Series: AIJR Proceedings; ISSN: 2582-3922; ISBN: 978-81-947843-8-8

The rest of the paper is organized as follows: Section II illustrates the power series approximation of trigonometric and inverse trigonometric functions along with its methodology and analysis in Section II-C and Section II-D respectively. Section III explains the proposed squaring technique using Vedic Mathematics. This is followed by conclusion in Section IV.

II. Power-Series Approximation of Trigonometric and Inverse Trigonometric Functions

A. *Trigonometric Taylor Series*

The Taylor series expansion [13] of a real-valued or complex-valued $f(x)$, an infinitely differentiable function at a real or complex point a, can be given by

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k \tag{1}$$

where, $f^{(k)}(a)$ is the k^{th} derivative of the function f evaluated at the point a (real or complex) and $k!$ is the factorial of k. A Maclaurin series can be considered as a special Taylor series expansion for a particular case when $f(x)$ is evaluated over $a = 0$ as follows:

$$\begin{aligned} f(x) &= \sum_{k=0}^{\infty} \frac{f^{(k)}(0)}{k!}(x)^k \\ &= f(0) + f'(0)x + \frac{f^{(2)}(0)}{2!}x^2 + \frac{f^{(3)}(0)}{3!}x^3 + \dots \\ &\quad + \dots + \frac{f^{(n)}(0)}{k!}x^k + \dots\dots \end{aligned} \tag{2}$$

The Maclaurin series expansion of a few important trigonometric functions ($sin(x)$, $cos(x)$, $tan(x)$) and inverse trigonometric functions ($sin^{-1}(x)$, $cos^{-1}(x)$) valid for all real and complex values of x are as follows:

$$\begin{aligned} sin(x) &= \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!}x^{2k+1} \\ &= x - \frac{x^3}{3!} + \frac{x^5}{5!} + \dots for\ all\ values\ of\ x \end{aligned} \tag{3}$$

$$\begin{aligned} cos(x) &= \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k)!}x^{2k} \\ &= 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \dots for\ all\ values\ of\ x \end{aligned} \tag{4}$$

$$\begin{aligned} tan(x) &= \sum_{k=1}^{\infty} \frac{B_{2n}(-4)^k(1-4^k)}{(2k)!}x^{2k-1} \\ &= x - \frac{x^3}{3} + \frac{2x^5}{15} - \dots\dots for \mid x \mid < \frac{\pi}{2} \end{aligned} \tag{5}$$

where B_{2n} is the Bernoulli series.

$$\begin{aligned} sin^{-1}(x) &= \sum_{k=0}^{\infty} \frac{(2k)!}{4^k(k!)^2(2k+1)}x^{2k+1} \\ &= x + \frac{x^3}{6} + \frac{3x^5}{40} + \dots\dots for \mid x \mid < 1 \end{aligned} \tag{6}$$

$$\begin{aligned} cos^{-1}(x) &= \frac{\pi}{2} - sin^{-1}(x) \\ &= \frac{\pi}{2} - \sum_{k=0}^{\infty} \frac{(2k)!}{4^k (k!)^2 (2k+1)} x^{2k+1} \\ &= \frac{\pi}{2} - x - \frac{x^3}{6} - \frac{3x^5}{40} - \ldots for \mid x \mid < 1 \end{aligned} \tag{7}$$

B. Basic Mathematical Functions for Binary Approximation

1) Ceil Function

This function returns the nearest integer value above the specified number x. Symbolically, the ceil of a number x is denoted by $\lceil x \rceil$. Suppose, the fixed-point fractional number is $x = b_i \ldots b_1 b_0 \cdot b_{-1} b_{-2} \ldots b_j$, the corresponding $\lceil x \rceil$ for binary number system can be given as

$$\begin{aligned} &\lceil b_i \ldots b_1 b_0 \cdot b_{-1} b_{-2} \ldots b_j \rceil \\ &\qquad = b_i \ldots b_1 b_0 + (b_{-1} \oplus b_{-2} \oplus \cdots \oplus b_j) \end{aligned} \tag{8}$$

where $\oplus$ is the logical OR operation.

2) Floor Function

This function returns the nearest integer value below the specified number x. Symbolically, the floor of a number x is denoted by $\lfloor x \rfloor$. Suppose, the fixed-point fractional number is given by $x = b_i \ldots b_1 b_0 \cdot b_{-1} b_{-2} \ldots b_j$, its corresponding $\lfloor x \rfloor$ for binary number system is given as

$$\lfloor b_i \ldots b_1 b_0 \cdot b_{-1} b_{-2} \ldots b_j \rfloor = b_i \ldots b_1 b_0 \tag{9}$$

3) Round Function

This function returns the nearest integer value depending on the fractional part of the number. If the fractional part of the number x is equal to or greater than 0.5, the function returns the smallest integer still above the specified number. Otherwise, the function returns the largest possible integer still smaller than the specified number. Suppose, the fixed-point fractional number is $x = b_i \ldots b_1 b_0 \cdot b_{-1} b_{-2} \ldots b_j$, the $round\ (x)$ for binary number system is stated as

$$round(b_i \ldots b_1 b_0 \cdot b_{-1} b_{-2} \ldots b_j) = b_i \ldots b_1 b_0 + b_{-1} \tag{10}$$

C. Methodology for Power Series Approximation of Trigonometric and Inverse Trigonometric Functions

This subsection elaborates the methodology that can be followed for power series approximation of a few trigonometric and inverse trigonometric series. This is explained in the following.

1) Trigonometric Series

a) Sine Series ($sin(\theta)$)

The power series approximation of $sin(\theta)$ can be given using (3) as

$$sin_{apx}(\theta_{apx}) = \theta_{apx} - \frac{\theta_{apx}^3}{3! \times (2^6)^3} + \frac{\theta_{apx}^5}{5! \times (2^6)^5} \tag{11}$$

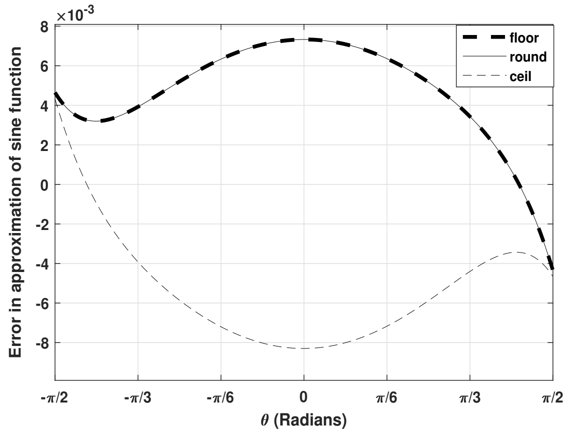
for all the values of θ. Since sine is a periodic function, θ is valid for $-\pi/2 \leq \theta \leq \pi/2$ only.

b) Cosine Series ($cos(\theta)$)

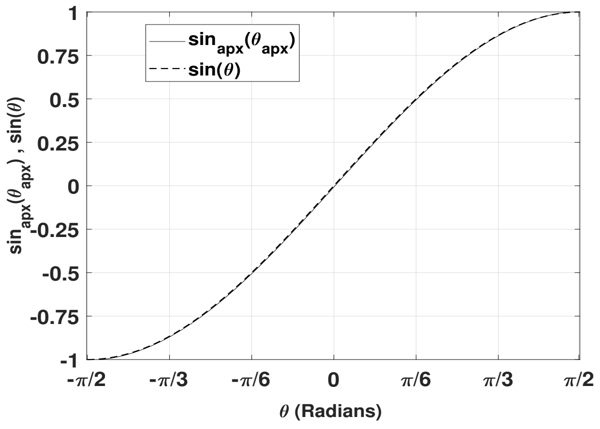
The power series approximation of $cos(\theta)$ can be given using (4) as

$$cos_{apx}(\theta_{apx}) = 1 - \frac{\theta_{apx}^2}{2! \times (2^6)^2} + \frac{\theta_{apx}^4}{4! \times (2^6)^4} - \frac{\theta_{apx}^6}{6! \times (2^6)^6} \tag{12}$$

for all the values of θ. Since cosine is a periodic function, θ is valid for $-\pi/2 \leq \theta \leq \pi/2$ only.



(a) $sin(\theta)$-$sin_{apx}(\theta_{apx})$



(b) $sin(\theta)$, $sin_{apx}(\theta_{apx})$

Fig. 1. Variation of approximated sine series ($sin_{apx}(\theta_{apx})$ w.r.t sine series ($sin(\theta)$)

c) Tan Series ($tan(\theta)$)

The power series approximation of $tan(\theta)$ can be given using (5) as

$$tan_{apx}(\theta_{apx}) = \theta_{apx} + \frac{\theta_{apx}^3}{3\times(2^6)^3} + \frac{2\theta_{apx}^5}{15\times(2^6)^5} + \frac{17\theta_{apx}^7}{315\times(2^6)^7} \tag{13}$$

for $-\pi/4 < \theta < \pi/4$.

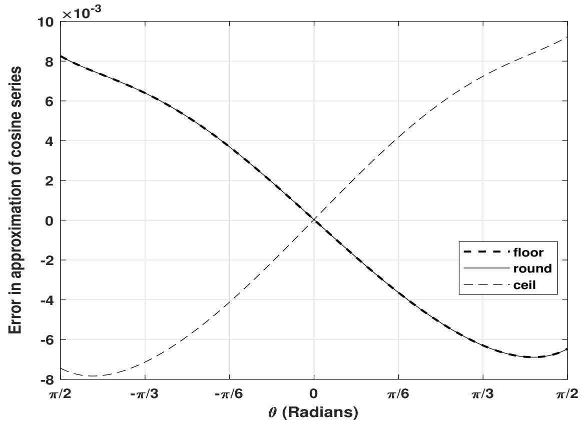
2) Inverse Trigonometric Series

a) Inverse Sine Series ($sin^{-1}(\theta)$)

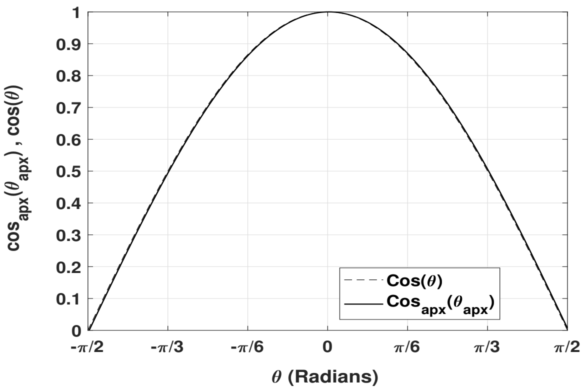
The power series approximation of $sin^{-1}(\theta)$ can be given using (6) as

$$\begin{aligned} sin_{apx}^{-1}(\theta_{apx}) &= \frac{(\theta_{apx})}{2^6} + \frac{(\theta_{apx})^3}{2\times3\times(2^6)^3} \\ &\quad + \frac{3\times(\theta_{apx})^5}{2\times\ 4\times5\times(2^6)^5} + \frac{(3\times5\times\theta_{apx})^7}{2\times4\times6\times7\times(2^6)^7} \end{aligned} \tag{14}$$

for -0.75 $\leq \theta \leq$ 0.75.



(a) $cos(\theta)$- $cos_{apx}(\theta_{apx})$



(b) $cos(\theta)$, $cos_{apx}(\theta_{apx})$

Fig. 2. Variation of approximated cosine series ($cos_{apx}(\theta_{apx})$ w.r.t cosine series ($cos(\theta)$)

b) Inverse Cosine Series ($cos^{-1}(\theta)$)

The power series approximation of $cos^{-1}(\theta)$ can be giving using (7) as

$$\begin{aligned} cos^{-1}_{apx}(\theta_{apx}) &= \frac{\pi}{2} - sin^{-1}_{apx}(\theta_{apx}) \\ &= \frac{\pi}{2} - \frac{(\theta_{apx})}{2^6} - \frac{(\theta_{apx})^3}{2\times 3\times (2^6)^3} \\ &\quad - \frac{3\times(\theta_{apx})^5}{2\times\ 4\times 5\times (2^6)^5} - \frac{(3\times 5\times\theta_{apx})^7}{2\times 4\times 6\times 7\times (2^6)^7} \end{aligned} \tag{15}$$

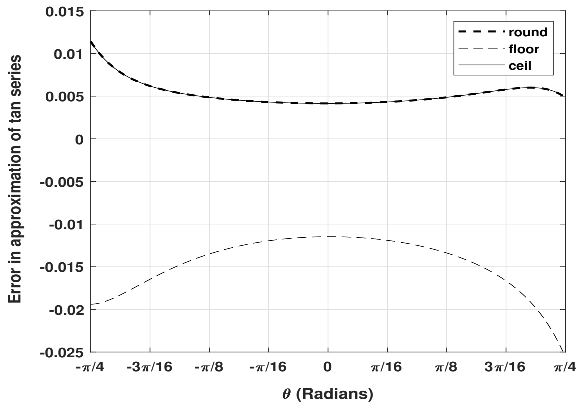
for -0.75 $\leq \theta \leq$ 0.75.

D. Analysis of Approximated Power Series of Trigonometric and Inverse Trigonometric Functions

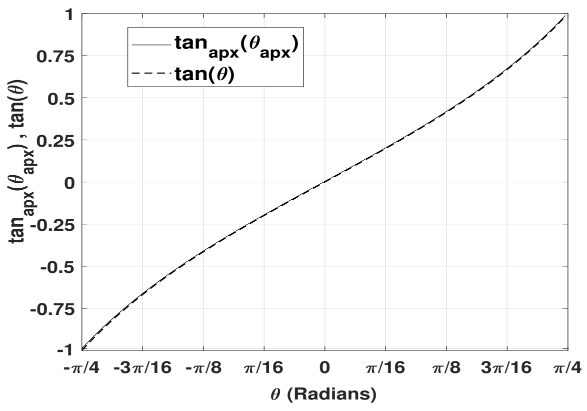
1) Trigonometric Series

a) Sine Series ($sin(\theta)$)

Fig. 1(a) shows the absolute error difference between $sin_{apx}(\theta_{apx})$ and $sin(\theta)$ for different binary approximations explained in Section II-B. The plot for sine approximations ($sin(\theta)$ and $sin_{apx}(\theta_{apx})$) with the best binary approximation for θ_{apx} is also shown in Fig. 1(b). From Fig. 1(b), it can be shown that the maximum possible error is 7.32×10^{-3} and the mean absolute error difference is 4.6×10^{-3}.



(a) $tan(\theta)$- $tan_{apx}(\theta_{apx})$



(b) $tan(\theta)$, $tan_{apx}(\theta_{apx})$

Fig. 3. Variation of approximated tan series ($tan_{apx}(\theta_{apx})$ w.r.t tan series ($tan(\theta)$)

b) Cosine Series ($cos(\theta)$)

Fig. 2(a) shows the absolute error difference between $cos_{apx}(\theta_{apx})$ and $cos(\theta)$ for different binary approximations explained in Section II-B. The plot for cosine approximations ($cos(\theta)$ and $cos_{apx}(\theta_{apx})$) with the best binary approximation for θ_{apx} is also shown in Fig. 2(b). The maximum possible error is 8.25×10^{-3} and the mean absolute error difference is 1.23×10^{-4} clearly shown in Fig. 2(b).

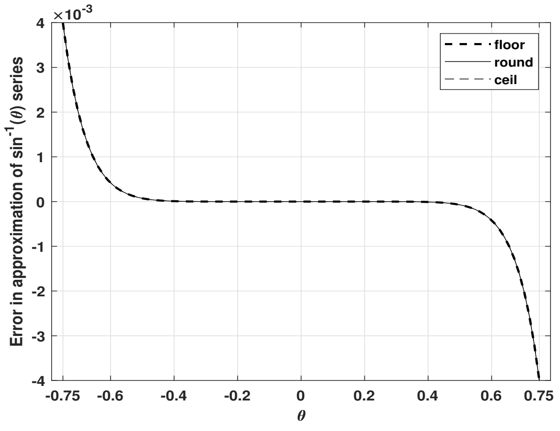
c) Tan Series ($tan(\theta)$)

Fig. 3(a) shows the absolute error difference between $tan_{apx}(\theta_{apx})$ and $tan(\theta)$ for different binary approximations explained in Section II-B. The plot for tan approximations ($tan(\theta)$ and $tan_{apx}(\theta_{apx})$) with the best binary approximation for θ_{apx} is also shown in Fig. 3(b). From Fig. 3(b), it can be shown that the maximum possible error is 11×10^{-3} and the mean absolute error difference is 9.7×10^{-3}.

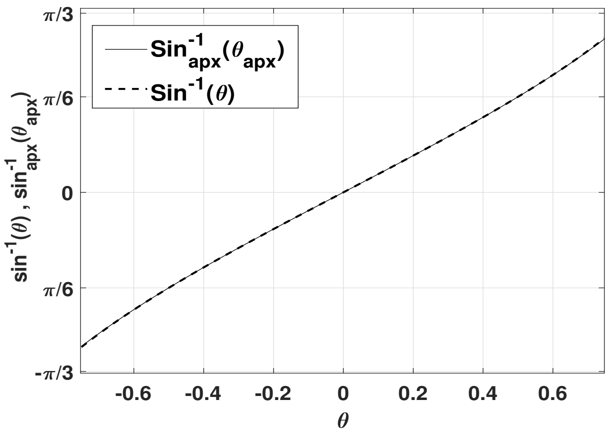
2) Inverse Trigonometric Series

a) Inverse Sine Series ($sin^{-1}(\theta)$)

Fig. 4(a) shows the absolute error difference between $sin^{-1}_{apx}(\theta_{apx})$ and $sin^{-1}(\theta)$ for different binary approximations explained in Section II-B. The plot for inverse sine approximations ($sin^{-1}(\theta)$ and $sin^{-1}_{apx}(\theta_{apx})$) with the



(a) $sin^{-1}(\theta)$- $sin^{-1}_{apx}(\theta_{apx})$



(b) $sin^{-1}(\theta)$, $sin^{-1}_{apx}(\theta_{apx})$

Fig. 4. Variation of approximated inverse sine series ($sin^{-1}_{apx}(\theta_{apx})$ w.r.t sine series ($sin^{-1}(\theta)$)

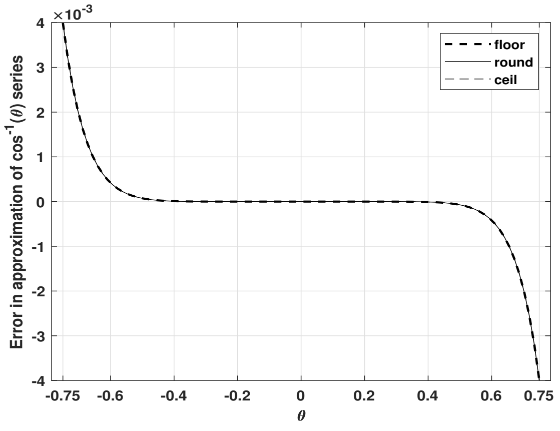
best binary approximation for θ_{apx} is also shown in Fig. 4(b). The maximum possible error is 4.2×10^{-3} and the mean of absolute error difference is 7.5×10^{-3} clearly shown in Fig. 4(b).

b) Inverse Cosine Series ($cos^{-1}(\theta)$)

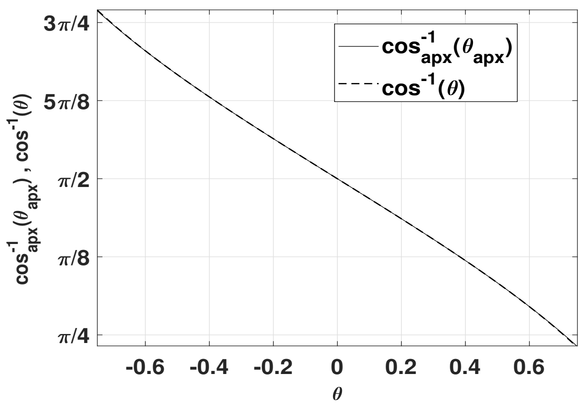
Fig. 5(a) shows the absolute error difference between $cos^{-1}_{apx}(\theta_{apx})$ and $cos^{-1}(\theta)$ for different binary approximations explained in Section II-B. The plot for Inverse cos approximations ($cos^{-1}(\theta)$ and $cos^{-1}_{apx}(\theta_{apx})$) with the best binary approximation for θ_{apx} is also shown in Fig. 5(b). From Fig. 5(b), it can be shown that the maximum possible error is 4.2×10^{-3} and the mean absolute error difference is 7.5×10^{-3}.

Table I gives the comparison of absolute mean error difference of the proposed work with existing power series approximations of trigonometric series [4]. $sin(\theta)$ and $cos(\theta)$ gives better results than [4]; although $tan(\theta)$ does not give better performance due to a high relative error near $\theta = -\frac{\pi}{4}$.

The absolute mean difference of the inverse trigonometric series is also summarized in Table II in reference to Fig. 3 and Fig. 4. From Table II it is observed that $sin^{-1}(\theta)$ and $cos^{-1}(\theta)$ produces the same absolute error difference of 7.5×10^{-3}. This is due to the fact that $cos^{-1}(\theta) = \frac{\pi}{2} - cos^{-1}(\theta)$ and the same error is reflected in both the sine and cosine inverse series.



(a) $cos^{-1}(\theta)$- $cos^{-1}_{apx}(\theta_{apx})$



(b) $cos^{-1}(\theta)$, $cos^{-1}_{apx}(\theta_{apx})$

Fig. 5. Variation of approximated inverse cosine series ($cos^{-1}_{apx}(\theta_{apx})$ w.r.t cosine series ($cos^{-1}(\theta)$)

TABLE I
COMPARISON OF THE PROPOSED WORK WITH EXISTING POWER SERIES APPROXIMATIONS OF TRIGONOMETRIC SERIES

Trigonometric Series	Absolute Error Difference	
	Tiwari [4]	Proposed
$sin(\theta)$	6.3×10^{-3}	4.6×10^{-3}
$cos(\theta)$	3.6×10^{-3}	1.23×10^{-4}
$tan(\theta)$	4.6×10^{-3}	9.7×10^{-3}

TABLE II
ABSOLUTE ERROR DIFFERENCE VALUES OF INVERSE TRIGONOMETRIC SERIES APPROXIMATIONS

Inverse Trigonometric Series	Absolute Error Difference
$sin^{-1}(\theta)$	7.5×10^{-3}
$cos^{-1}(\theta)$	7.5×10^{-3}

III. Proposed Vedic Squaring Technique

Let us illustrate the proposed squaring technique taking an example for calculating the square of a binary number A with n-bit length. A can be partitioned into two equal-sized binary numbers: A_H and A_L, using Karatsuba-Ofman algorithm (K-O) algorithm [14]. Note: The subscripts H and L represent higher-order $n/2$ bits and lower-order $n/2$ bits respectively only if n is even. If n is odd, the MSB bit of A_H need to be padded with zeros so that A_H and A_L are of equal $n/2$ bit-length [4].

The square of A can be calculated in terms of A_H and A_L and stated as follows:

$$A^2 = A \times A = 2^n(A_H^2) + 2^{\frac{n}{2}+1}(A_H \times A_L) + (A_L^2) \tag{16}$$

$A_H \times A_L$ can be easily calculated with the help of Urdhwa Tribhagyam Sutra of Vedic multiplication for binary numbers [5]. The terms A_H^2 and A_L^2 in 16 can be further split up recursively to two-bit binary numbers for squaring up using K-O algorithm. Next, the conditional squarer can be realized for squaring two-bit binary number with the following cases:

$$\begin{aligned} If\ A &= 00, A^2 = 0000 \\ If\ A &= 11, A^2 = 1001 \\ If\ A &= 10, A^2 = 0100 \\ If\ A &= 01, A^2 = 0001 \end{aligned} \tag{17}$$

For better understanding of the proposed squaring technique using 16, let us take another example $A = 1101$. Here, $n = 4$, therefore, the two equal-sized binary numbers are: $(A_H) = 11$ and $(A_L) = 01$. Using (17) of the conditional squarer, the values of A_H^2 and A_L^2 are $= 1001$ and 0001 respectively. Next, the value of $A_H \times A_L$ = 0011 can be easily calculated using Urdhwa Tribhagyam Sutra algorithm. Since $n = 4$, the left shifting of obtained A_H^2 by n bits result into 10010000; similarly, the left-shift of $A_H \times A_L$ by $(\frac{n}{2} + 1)$ bits give 0011000. Lastly, adding A_L^2 and the left-shifted values of $A_H \times A_L$, A_H^2 by $\frac{n}{2} + 1$ bits and n, respectively gives the value of A^2= 010101001. This explains the squaring operation of A using (16). The major advantage of the proposed squaring technique is square calculation of a binary number without the use of the "Duplex" property of binary numbers providing lesser algorithmic complexity [4], [5], .

Future works focuses on the hardware implementation of the power series approximation of trigonometric and inverse trigonometric functions. This aims towards high-speed architecture for integers. Preliminary methodology for conversion of floating-point numbers to approximated binary numbers are already analyzed in detailed in Section II. A close observation of subsection states that, for hardware implementation of approximated power series (for trigonometric and inverse trignometric functions) requires squarer, multiplier, divider and factorial circuitry [1], [15], [16]. Hardware implementation of these circuits is beyond the scope of this paper; but these will be considered as future works.

IV. Conclusions

This paper presents an efficient Vedic squaring technique without the use of Duplex property and it can easily be extended for any n-bit binary numbers. The main advantage of this proposed technique is that it focuses on reduced algorithmic complexity. This work also presented the analytical study of Taylor series approximation of trigonometric functions: $sin(x)$, $cos(x)$, $tan(x)$) and inverse trigonometric functions: $sin^{-1}(x)$, $cos^{-1}(x)$. Future works will focus on hardware implementations of the proposed power series approximations on digital VLSI architectures using FPGAs and ASICs. Experimental results might reflect great potential in digital arithmetic for prospective exploration of high-performance and embedded computing systems.

References

[1] J. Pihl and E. J. Aas, "A multiplier and squarer generator for high performance DSP applications," in *Proceedings of the 39th Midwest Symp. on Circuits and Systems*, vol. 1, 1996, pp. 109–112.

[2] A. Kant and S. Sharma, "Applications of Vedic multiplier designs - A review," in *2015 4th Int. Conf. on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)*, 2015, pp. 1–6.

[3] K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*. Wiley India Pvt. Limited, 2007.

[4] H. D. Tiwari, "Vedic processor based trigonometric calculations using power series," in *2019 5th Int. Conf. on Advanced Computing and Communication Systems (ICACCS)*, 2019, pp. 118–123.

[5] Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja, *Vedic Mathematics: Sixteen Simple Mathematical Formulae from the Vedas*. Motilal Banarsidass Publishers, 01 2005.

[6] L. Sriraman, K. Kumar, and T. N. Prabakar, "Design and FPGA implementation of binary squarer using Vedic mathematics," in *2013 4th Int. Conf. on Computing, Communications and Networking Technologies (ICCCNT)*. Los Alamitos, CA, USA: IEEE Computer Society, July 2013, pp. 1–5.

[7] S. Barve, S. Raveendran, C. Korde, T. Panigrahi, Y. B. Nithin Kumar, and M. H. Vasantha, "FPGA Implementation of Square and Cube Architecture Using Vedic Mathematics," in *2018 IEEE Int. Symp. on Smart Electronic Systems (iSES) (Formerly iNiS)*, 2018, pp. 6–10.

[8] H. Thapliyal, S. Kotiyal, and M. B. Srinivas, "Design and Analysis of a novel parallel square and cube architecture based on ancient Indian Vedic mathematics," in *48th Midwest Symp. on Circuits and Systems, 2005.*, vol. 2, 2005, pp. 1462–1465.

[9] S. Jalaja and A. M. V. Prakash, "High Speed VLSI Architecture for Squaring Algorithm Using Retiming Approach," in *3rd Int. Conf. on Advances in Computing and Communications*, 2013, pp. 233–238.

[10] S. Akhter and S. Chaturvedi, "Modified Binary Multiplier Circuit Based on Vedic Mathematics," in *6th Int. Conf. on Signal Processing and Integrated Networks (SPIN)*, 2019, pp. 234–237.

[11] A. Kumar, D. Kumar, and Siddhi, "Hardware Implementation of 16 $\times$ 16 bit Multiplier and Square using Vedic Mathematics," 2012.

[12] S. Patil, D. V. Manjunatha, and D. Kiran, "Design of speed and power efficient multipliers using Vedic mathematics with VLSI implementation," in *Int. Conf. on Advances in Electronics Computers and Communications*, 2014, pp. 1–6.

[13] A. T. Claudio Canuto, *Mathematical Analysis I*, 2nd ed. Springer International Publishing, 2015.

[14] H. Thapliyal and M. Srinivas, "High Speed Efficient N $\times$ N Bit Parallel Hierarchical Overlay Multiplier Architecture Based On Ancient Indian Vedic Mathematics," 12 2004.

[15] P. Saha, A. Banerjee, A. Dandapat, and P. Bhattacharyya, "ASIC design of a high speed low power circuit for factorial calculation using Ancient Vedic mathematics," *Microelectron. J.*, vol. 42, pp. 1343–1352, 2011.

[16] P. Saha, A. Banerjee, P. Bhattacharyya, and A. Dandapat, "Vedic Divider: Novel Architecture (ASIC) for High Speed VLSI Applications," in *2011 Int. Symp. on Electronic System Design*, 2011, pp. 67–71.