International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

DESIGN OF LOW POWER AND AREA EFFICIENT APPROXIMATE MULTIPLIER WITH REDUCED COMPLEXITY USING ENCODED PARTIAL PRODUCTS

BHUTHAPURI SAI KIRAN[1], N.SIVAKUMAR REDDY[2]
[1]PG SCHOLAR.DEPT OF ECE, SIR C.V. RAMAN INSTITUTE OF TECHNOLOGY & SCIENCE, AP, INDIA
[2] ASST. PROFESSOR, DEPT OF ECE, SIR C.V. RAMAN INSTITUTE OF TECHNOLOGY & SCIENCE, AP, INDIA

ABSTRACT:
Arithmetic and logic unit has been the most significant unit in any electronic devices. In the recent advancement, for an arithmetic and logic unit to be significant it needs to have an efficient algorithmic operation such as Multiplications and addition. Multiplication is the most essential operation in digital signal processing, artificial intelligence, neural networks, and machine learning. In the VLSI domain, the performance of electronic devices depends on area, power, and delay. Need of approximate computing has increased to the pinnacle for error-tolerant applications. This leads to further optimization in terms of power and area with remarkable enhancement in computational speed within an acceptable error range. So, the performance and the error characteristic are considered with equal importance. This work presents a power and area optimized architecture for error-tolerant multiplier. A trade-off is made with the error characteristic by using approximation at the lower stage of partial product. Results for area, power and error factor are shown for range of multipliers. A significant improvement is achieved by using the proposed technique

1.INTRODUCTION

In the process of evolution of technology, many Information Technology (IT) applications are requiring low-power integrated circuits (ICs), since handling huge amount of data to process. Due to integrated applications the size also increases. The energy efficiency is measured as the product of power consumption and computational time. To provide an energy efficient solution, the designer has to reduce the power consumption and the computing time as well. By reducing logic blocks, the static power can be reduced. However, reduction in power consumption has been achieved by different methods that demonstrates a trade-off relationship with circuit performance [1]. Low power consumption has become the decisive design goal in wide range of electronic systems and sub-systems. Inbuilt-powered sensing modules stand first in this row for the designers [2].

Processing units are the most power-hungry part of any computing system and are the intrinsic part of all the computing hardware designs [9]. The strict power challenges and high performance can be achieved by adopting new design techniques to meet computing e_ciency [10]. Approximate or inexact computing serves a better and promising methodology for a series of applications such as Machine-Learning (ML), Computer Vision (CV), Image and/or Video Signal Processing [11]. Reduced computation time saves the energy significantly [1]. Many applications where errors are introduced intentionally and that makes the approximate computation more suitable to apply because of difficulties to recognize small errors. Especially, we will find such cases in image compression and decompression by separating the intensity. For aforesaid applications approximate computing is the ultimate and efficient solution for low power logic implementation.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

However, the fact remains that the area and speed are two conflicting performance constraints. Hence, innovating increased speed always results in larger area. The proposed architecture enhances the speed performance of the widely acknowledged Wallace tree multiplier when implemented on a FPGA. The structural optimization is performed on the conventional Wallace multiplier, in such a way that the latency of the total circuit reduces considerably. A truncated multiplier with constant correction has the maximum error if the partial products in the n-k least significant columns are all ones or all zeros. A variable correction truncated multiplier has been proposed. This method changes the correction term based on column n-k-1. If all partial products in column n-k-1 are one, then the correction term is increased. Similarly, if all partial products in this column are zero, the correction term is decreased. In a simplified 22 multiplier block is proposed for building larger multiplier arrays. In the design of a fast multiplier, compressors have been widely used to speed up the partial product reduction tree and decrease power dissipation. Kelly et al. and Ma et al. have also considered compression for approximate multiplication. An approximate signed multiplier has been proposed for use in arithmetic data value speculation (AVDS); multiplication is performed using the Baugh Wooley algorithm. However no new design is proposed for the compressors for the inexact computation.

2.LITERATURE SURVEY

High-performance and energy-efficient approximate multiplier for error-tolerant applications by S. Kim and Y. Kim,

Approximate computing in the arithmetic logic has recently emerged as a promising solution for energy-efficient designs in error-tolerant applications. So not only the performance but also the error characteristic become important criteria. In this study, we propose a high-performance and energy-efficient approximate multiplier with suitable error characteristics by using low energy approximate adders for lower bits. Simulation results of 16×16 multipliers show that area, delay, and power saving are significantly improved by the proposed multiplier compared to the conventional binary multiplier. In addition, error metrics of the proposed multiplier have shown better results than other approximate multipliers.

Approximate Multipliers Based on Inexact Adders for Energy Efficient Data Processing by M. Osta, A. Ibrahim, H. Chible and M. Valle

Approximate computing circuits are considered as a promising solution to reduce the power consumption in embedded data processing. This paper proposes an FPGA implementation for an approximate multiplier based on inexact adder circuits. The performance of the proposed multiplier is evaluated by comparing the power consumption, the accuracy of computation, and the time delay with those of an approximate multiplier based on exact adder presented in literature. Results reports a power saving up to 17.39% with an improvement in time delay by 13.49%, at cost of less than 5% of accuracy loss.

Power and Delay Efficient Exact Adder for Approximate Multiplier by V. V. Kavipranesh, J. Janarthanan, T. N. Amruth, T. M.Harisuriya and E. Prabhu,

Approximate results are required in many embedded data processors as they reduce time delay and power. As error tolerance adder (ETA) has decreased power drastically trading with accuracy. This work focuses on reducing delay on existing adders when replaced with a fast adder. When compared to the past works on ETA, the proposed work has high power

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

utilization and more accuracy of speed. The proposed design is compared and synthesized for the power and delay. When observed the existing ETA designs, the proposed work achieves significant improvement in power dissipation about 17.13%, 4.6%, 15.4%, 5.35% decrement for 4, 8, 16, 32 bits respectively, and significant improvement in delay about 28.90%, 23.59%, 20.08%, 24.44% decrement for 4, 8, 16, 32 bits respectively.

3 EXISTING SYSTEM

The algorithm of WALLACE multiplier is based on the below matrix form. The partial product matrix is formed in the first stage by AND stages

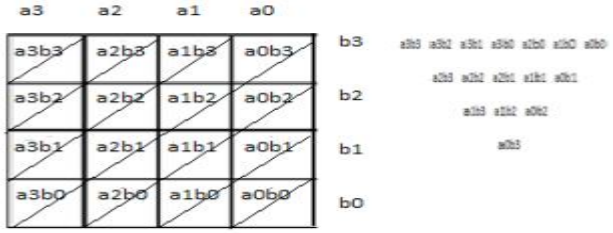

Fig.1 4x4 WALLACE Algorithm

Steps involved in WALLACE TREE multipliers Algorithm

Multiply (that is - AND) each bit of one of the arguments, by each bit of the other, yielding N results. Depending on position of the multiplied bits, the wires carry different weights.

Reduce the number of partial products to two layers of full adders.

Group the wires in two numbers, and add them with a conventional adder.. Product terms generated by a collection of AND gates.

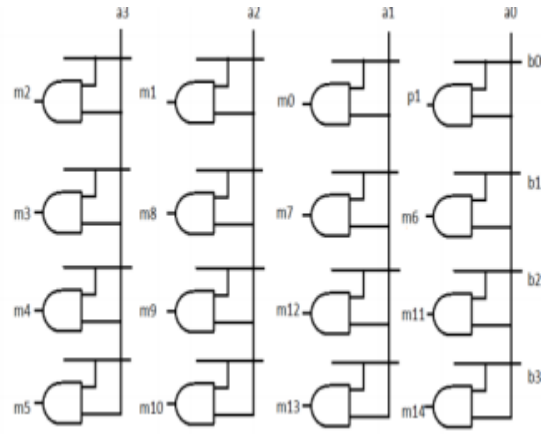

Fig2 Product terms generated by a collection of AND gates.

Wallace Tree Multiplier Using Ripple Carry Adder

Ripple Carry Adder is the method used to add more number of additions to be performed with the carry in sand carry outs that is to be chained. Thus multiple adders are used in ripple carry adder. It is possible to create a logical circuit using several full adders to add multiple-bit numbers. Each full adder inputs a Cin, which is the Cout of the previous adder. This kind of adder is a ripple carry adder, since each carry bit "ripples" to the next full adder. The proposed architecture of WALLACE multiplier algorithm using RCA is shown in Figs.9 to 11 Take any 3 values with the same weights and gives them as input into a full adder. The result will be an output wire of the same weight. Partial product obtained after multiplication is taken at the first stage. The data's are taken with 3 wires and added using adders and the carry of each stage is added with next two data‟s in the same stage. Partial products reduced to two layers of full adders with same procedure. At the final stage, same method of ripple carry adder method is performed and thus product terms p1 to p8 is obtained.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

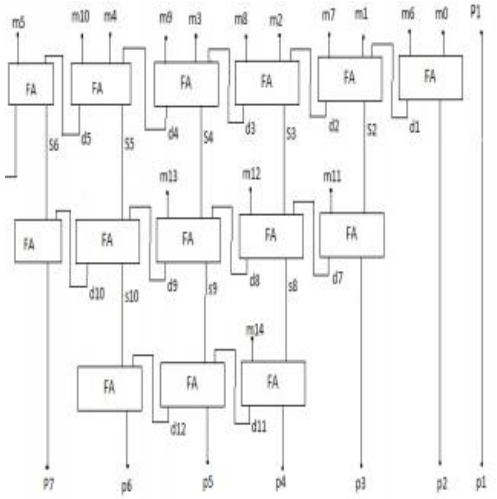

Fig 3 .4x4 Wallace Multiplier Implementation.

4.PROPOSED SYSTEM

Wallace Tree Multiplier (WTM) is yet to be considered as a hardware efficient arithmetic circuit [20]. After a generation of partial products using N2 AND-Gates (for N-bit multiplier), compressors are used to add. An accurate 2-bit multiplier will generate maximum "1001" when A [1:0] = "11" and B [1:0] ="11". Other input combinations will generate the result less than or equal to "110" which can be represented by 3 binary bits.

Eliminating the MSB of the 4-bit result of an accurate multiplier, the multiplier will result "001" instead of "1001" with an error magnitude of 8 for A[1:0] = "11" and B[1:0] = "11" combination. The functional table is given in TABLE I to represent the reduced functionalities by masking the carry bit in 2-bit multiplier. Using the 2-bit multiplier, the higher bit multipliers can be designed with an increased Error Factor (EF).

By using K-Maps, the Boolean Functions are expressed as shown in Eq. 1 to Eq. 3. The circuit implementation can be realized as shown in Fig. 1.

$$P[0]=A[0].B[0] \quad (1)$$

$$P[1]=A[1].A[0].B[1]+A[0].B[1].\overline{B[0]}+A[1].\overline{A[0]}.B[0]+A[1].\overline{B[1]}.B[0] \quad (2)$$

$$P[2]=A[1].B[1].B[0]+A[1].A[0].B[1] \quad (3)$$

A trade-off is made in terms of accuracy to optimize the power and area. The Boolean equations are given for a 2-bit multiplier, where A[1:0] and B[1:0] are the input vectors and P[2:0] is the output vector. The multiplier is truncated for a trade-off in accuracy and logic gates. The manipulated functionality is given in the TABLE 1 and its logic in Fig. 1.Now replacing the output by "111" in TABLE I for input A[1:0] = "11" and B[1:0] = "11", the error factor can be reduced to 2. Secondly, the Boolean expressions are simplified further and shown in Eq. 4 to Eq. 6.

$$P[0] = A[0] . B[0] \quad (4)$$

$$P[1] = A[0] . B[1] + A[1] . B[0] \quad (5)$$

$$P[2]=A[1] . B[1] \quad (6)$$

TABLE I. FUNCTIONALITY OF 2-BIT APPROXIMATE MULTIPLIER

Inputs		Outputs
A [1:0]	*B [1:0]*	*P [2:0]*
00	00	000
00	01	000
00	10	000
00	11	000
01	00	000
01	01	001
01	10	010
01	11	011
10	00	000
10	01	010
10	10	100
10	11	110
11	00	000
11	01	011
11	10	110
11	11	001

Using 4-variable K-Map the simplified Boolean expressions are derived and given in Eq. 4 to Eq. 6.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

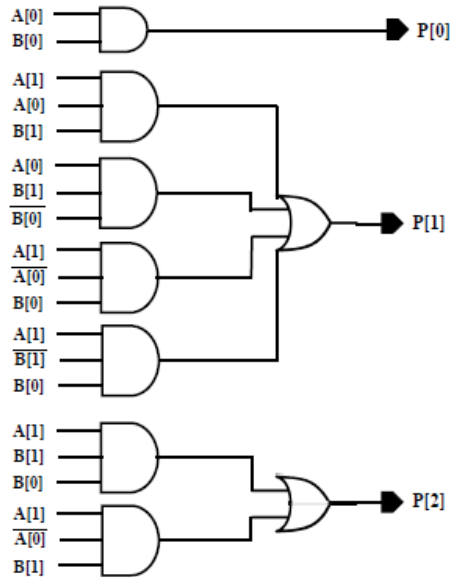

Fig.4. Simplified Reduced Partial Product Architecture

In first stage, partial products A[0]B[0], A[0]B[1], A[1]B[0] and A[1]B[1] are generated. The partial product results in a 3-bit output and adder stages are eliminated as shown in Fig. 2. Six 3-input AND-Gates are replaced by 3 two-input AND Gates. This gives an area efficient solution with improved accuracy.

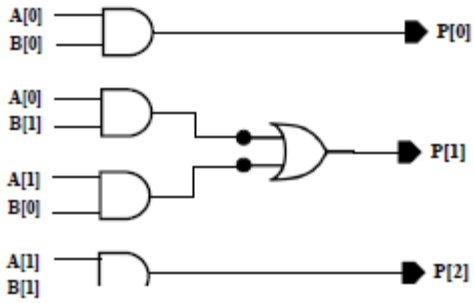

Fig 5. Reduced Partial Product Multiplier with Less EF and Area

The operation of 4-bit multiplier can be explained using Fig. 3 as shown in two stages. In first stage operands are grouped and multiplication is carried out using 2-bit multipliers. The outputs of the first stage are added up to derive the result. Using similar techniques, 8-bit multiplier is designed. The Architecture of a 4- bit error-tolerant multiplier is shown in Fig. 4 and an 8-bit multiplier in Fig-5.

The operation can be explained by dividing the 4-bit operands into a group of 2-bit. The partial products are represented by solid circles. Lower part adders are used to generate the results represented solid squares.

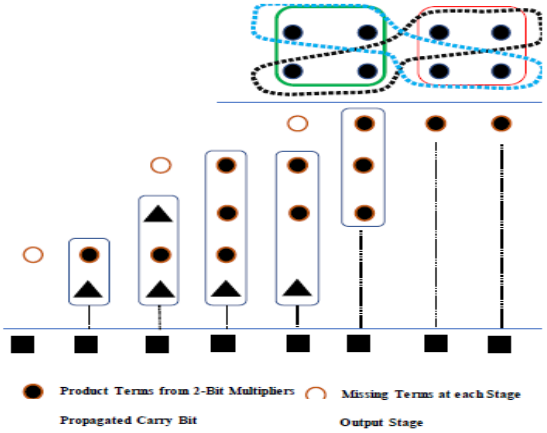

Fig. 6. Operation of 4-bit Approximate Multiplier

The proposed architecture is shown in Fig. 3. The partial products are generated using 2-bit multipliers and the basic adders are used to derive the output. Because of reduced partial product the adder size is reduced during 4-bit and 8-bit multiplier design. The Reduced Partial Product Multiplier (RPPM) gives low-power and area efficient solution for approximate computing and error tolerant applications. The functional blocks are shown in Fig. 5 after RTL implementation.

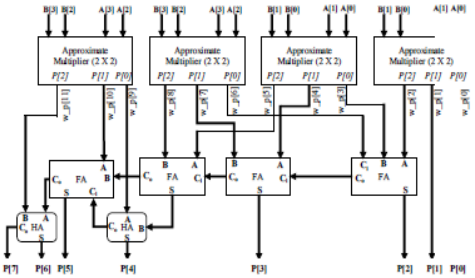

Fig.7. 4-Bit Approximate Multiplier using RPPM

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

The 8x8 bit Vedic multiplier module as shown in the block diagram in Fig. 6 can be easily implemented by using four 4x4 bit Vedic multiplier modules as discussed in the previous section. Let's analyze 8x8 multiplications, say A= A7 A6 A5 A4 A3 A2 A1 A0 and B= B7 B6 B5B4 B3 B2 B1B0. The output line for the multiplication result will be of 16 bits as – S15 S14 S13 S12 S11 S10 S9 S8 S7 S6S5S4 S3 S2 S1 S0. Let's divide A and B into two parts, say the 8 bit multiplicand A can be decomposed into pair of 4 bits AH-AL. Similarly multiplicand B can be decomposed into BH-BL. The 16 bit product can be written as:

Using the fundamental of Vedic multiplication, taking four bits at a time and using 4 bit multiplier block as discussed we can perform the multiplication. The outputs of 4x4 bit multipliers are added accordingly to obtain the final product. Here total three 8 bit Ripple-Carry Adders are required as shown in Fig. 6.

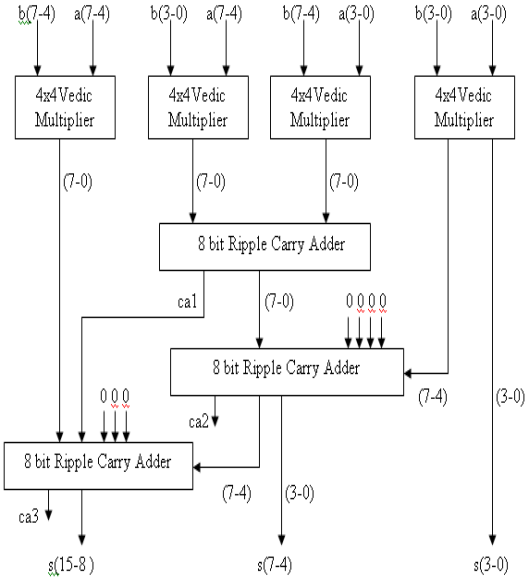

Fig 8. Block Diagram of 8x8 bit Vedic Multiplier

Multiplier is an integral part of the processor. Vedic multiplier is based on the Vedic mathematics. Vedic multiplier's architecture is based on the Sutra called "Urdhva Tiryakbhyam". Urdhva Tiryakbhyam (Vertically and Crosswise), deals with the multiplication of numbers. This Sutra has been traditionally used for the multiplication of two numbers in the decimal number system. By applying the algorithm to the binary number system, architecture for multiplier is designed. Other sutra used for Vedic multiplier is Nikhilam Sutra which literally means "all from 9 and last from 10". Although it is applicable to all cases of multiplication, it is more efficient when the numbers involved are large. Compared with the other multipliers, Vedic multiplier has less delay time. Vedic multiplier is also used for computing Fast Fourier transform.

5.EXTENSION

Implementation of FIR Filter using proposed multiplier

A filter is a device or process that removes some unwanted component or feature from a signal. Filtering is a class of signal processing, the defining feature of filter being the complete or partial suppression of some aspect of the signal. There are two main kinds of filter, analog and digital. Filters can be classified in several different groups, depending on what criteria are used for classification. The two major types of digital filters are finite impulse response digital filters (FIR filters) and infinite impulse response digital filters (IIR).

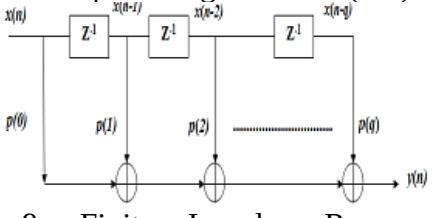

Fig 9. Finite Impulse Response Filter Realization

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

FIR filters are one of the primary types of filters used in Digital Signal Processing. FIR filters are said to be finite because they do not have any feedback. Therefore, if we send an impulse through the system (a single spike) then the output will invariably become zero as soon as the impulse runs through the filter. A non-recursive filter has no feedback. The Finite Impulse Response Filter Realization is as shown in figure 10.

Finite Impulse response (FIR) digital filter is widely used in several digital signal processing applications, such as speech processing, loud speaker equalization, echo cancellation, adaptive noise cancellation, and various communication applications, including software-defined radio (SDR) and so on. Many of these applications require FIR filters of large order to meet the stringent frequency specifications. Very often these filters need to support high sampling rate for high-speed digital communication. The number of multiplications and additions required for each filter output, however, increases linearly with the filter order. Since there is no redundant computation available in the FIR filter algorithm, real-time implementation of a large order FIR filter in a resource constrained environment is a challenging task. Filter coefficients very often remain constant and known apiarian signal processing applications. Finite Impulse Response (FIR) filters are widely used in digital signal processing. An N-tap FIR filter is defined by the following input-output equation 1

$$out(n) = \sum_{i=0}^{N-1} x(n-i)\, h(i)$$

-→1

Where {h (i): i = 0… N-1} are the filter coefficients.

An FIR filter implements a convolution operation [1], which is often built on the assumption of infinite length signals. Finite length signals (e.g. images) on the other hand, have discontinuities at the boundaries. Thus emerges the problem of which values to use at these regions. A usually recommended solution is to extend each row by reflection at the signal edges. The number of extra samples introduced at the signal boundaries is equal to N-1. They can be partitioned unequally between the left and the right side signal. We will refer by α and µto the number of samples introduced respectively at the left side and the right side input signal (α+µ=N-1).

6.SIMULATION RESULTS

8.1 Results

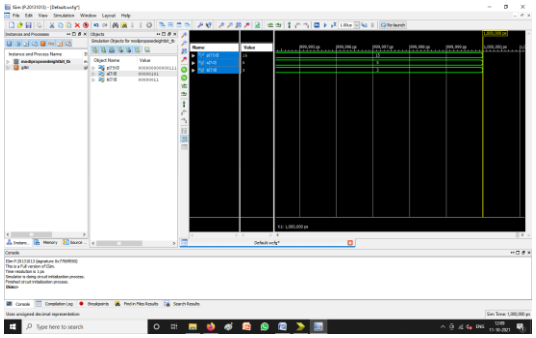

Fig 10 Simulation Result of Multiplier

Here we can give the inputs as A=3, B=5, then the final output is 15.

Table 2 Comparison between existing and proposed in Area, power and delay

SYSTEM	AREA	POWER(mw)	DELAY(ns)
Existing(Wallace tree multiplier)	143	0.299	32.681
Proposed method	121	0.162	22.008

7.CONCLUSION AND FUTURE SCOPE

The approximate 8-bit multiplier and 8-bit accurate Wallace tree multiplier are implemented using Xilinx software. This project presents a highly efficient method of multiplication – “Urdhva Tiryakbhyam Sutra”

based on Vedic mathematics. It gives us method for hierarchical multiplier design and clearly indicates the computational advantages offered by Vedic methods. Hence our motivation to reduce delay is finely fulfilled. Therefore, we observed that the Vedic multiplier is much more efficient than Array and Booth multiplier in terms of execution time (speed). An awareness of Vedic mathematics can be effectively increased if it is included in engineering education. In future, all the major universities may set up appropriate research centers to promote research works in Vedic mathematics. The power saving is achieved up to 29.29% as compared to WTM in static mode and dynamic power up to 33.35%. The poposed approximate architecture is also found to be even better in the chip area. The silicon area is saved up to 25.7% because of the reduced product method.

This multipliers plays a very important role in our day to day life. In future the multipliers are going to play a major role. The speed of the multipliers are increased by using carry save adders, carry look ahead adder, and so on. Rounding patterns will be optimized based on required accuracy and different compression techniques. The area and delay can be reduced in future by using advanced technology.

BLIOGRAPHY

[1] S. Kim and Y. Kim, "High-performance and energy-efficient approximate multiplier for error-tolerant applications," International SoC Design Conference (ISOCC), Seoul, pp. 278- 279, 2017.

[2] M. Osta, A. Ibrahim, H. Chible and M. Valle, "Approximate Multipliers Based on Inexact Adders for Energy Efficient Data Processing," New Generation of CAS (NGCAS), Genova, pp. 125-128, 2017.

[3] V. V. Kavipranesh, J. Janarthanan, T. N. Amruth, T. M. Harisuriya and E. Prabhu, "Power and Delay Efficient Exact Adder for Approximate Multiplier," International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, pp. 1896-1899, 2018.

[4] W. Liu, T. Zhang, E. McLarnon, M. O. Neill, P. Montuschi and F. Lombardi, "Design and Analysis of Majority Logic Based Approximate Adders and Multipliers," IEEE Transactions on Emerging Topics in Computing, 2019.

[5] K. R. Varma and S. Agrawal, "High speed, Low power Approximate Multipliers," International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, pp. 785-790, 2018.

[6] P. Yadav, etal.: "Low Power Approximate Multipliers with Truncated Carry Propagation for LSBs," IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS), Windsor, ON, Canada, pp. 500-503, 2018.

[7] S. Venkatachalam, H. J. Lee and S. Ko, "Power Efficient Approximate Booth Multiplier," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, pp. 1-4, 2018.

[8] S. Ravi, G. S. Nair, R. Narayan and H. M. Kittur, “Low Power and Efficient Dadda Multiplier,” Research Journal of Applied Sciences, Engineering and Technology, pp. 53-57, 2015.

[9] A. Sampson, J. Nelson, K. Strauss and Luis Ceze, “Approximate storage in solid-state memories,” ACM Trans. Comput. Syst. vol. 32, No. 3, Article 9, pp. 1-23, 2014.

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

[11] V. K. Chippa, etal.: “Analysis and characterization of inherent application resilience for approximate computing,” DAC, 2013.
[12] Z. Yang, A. Jain, J. Liang, J. Han and F. Lombardi, “Approximate xor/xnor-based adders for inexact computing,” in Proc. Nanotechnology (IEEE-NANO), pp. 690-693, 2013.
[13] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie and C. Lucas, "Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 4, pp. 850-862, April 2010.
[14] T. Yang, T. Sato and T. Ukezono, "An Approximate Multiply- Accumulate Unit with Low Power and Reduced Area," IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Miami, FL, USA, pp. 385-390, 2019.
[15] S. Hashemi, R. I. Bahar and S. Reda, "DRUM: A Dynamic Range Unbiased Multiplier for approximate applications," IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 418-425, 2015.
[16] T. Yang, T. Ukezono and T. Sato, "A Low-Power Yet High- Speed Configurable Adder for Approximate Computing," IEEE International Symposium on Circuits and Systems (ISCAS), Florence, pp. 1-5, 2018.
[17] R. Ye, T. Wang, F. Yuan, R. Kumar and Q. Xu, "On reconfiguration-oriented approximate adder design and its application," IEEE/ACM International Conference on Computer- Aided Design (ICCAD), pp. 48-54, 2013.
[18] K. Bhardwaj, P. S. Mane and J. Henkel, "Power- and areaefficient Approximate Wallace Tree Multiplier for error-resilient systems," Fifteenth International Symposium on Quality Electronic Design, pp. 263-269, 2014.
[19] N. Prajwal, S. K. Amaresha and S. S. Yellampalli, "Low power ASIC implementation of signed and unsigned wallace-tree with vedic multiplier using compressors," International Conference on Smart Technologies for Smart Nation (SmartTechCon), pp. 750- 753, 2017