Energy Efficient And High-Speed Approximate Multiplier Using Rounding Technique

V.Chandran
Assistant Professor, Department of ECE
Bannari Amman Institute of Technology
Sathyamangalam, Tamilnadu, India
chandran@bitsathy.ac.in

B.Elakkiya
M.E Student, Department of ECE
Government College of Technology
Coimbatore, Tamilnadu, India
elakz05@gmail.com

Abstract

Consumption of Energy is the major factor, in the various processing application like DSP, ASIC, and FPGA. The motive of this work is to approximate the multiplication process. The multiplier operands are rounding off to the two power N format which is nearest to the input values. With a small penalty of error, the speed and energy considerably increased. Literature survey reveals that earlier works are based on modifying the structure or complexity reduction of a specific accurate multiplier. This multiplier leads to better error rate when compared with other multipliers. So the rounding based inexact multiplication provides high speed and energy efficient for various processors. The hardware architecture is constructed for the approximate multiplication process for all possible multiplications using Quartus II 10.0 tools. The area, speed, and timing analysis are performed for this approach and for some existing accurate and approximate multipliers. The proposed 8-bit RoBA multiplier multiplication offers better efficiency in energy consumption when compared with other existing accurate and approximate multipliers. Furthermore, the area is compacted well besides it provides the reduction in Power Delay Area (PDA). In future, the capability of approximate RoBA multiplier was processed in the various processing in images.

Keywords*—Round-off, Approximate, Energy consumption Rounding Based Approximate Multiplier (RoBA), Power Delay Product (PDA).*

INTRODUCTION

Energy minimization is major requirements in almost any electronic systems, especially the portable ones such as smart phones, tablets, and different gadgets. It is extremely desired to attain this minimization with minimal performance (speed) penalty [1]. Digital signal processing (DSP) blocks are most wanted in transportable components for realizing various multimedia applications. The computational core of these blocks is the ALU where the multiplications and additions are the major part [6]. The multiplications plays foremost operation in the processing elements which can leads to high consumption of energy and power. Many of the DSP cores implement image and video processing algorithms where final outputs are either images or videos prepared for human consumptions. It facilitates to go for approximations for improving the speed and energy in the arithmetic circuits. This originates from the limited perceptual abilities in observing an image or a video for human beings. In addition to the image and video processing applications, there are other areas where the exactness of the arithmetic operations is not critical to the functionality of the system (see [2],[3]).

Approximate computing provides an accuracy, speed and power/energy consumption. The advantage of approximate multiplier reduces the error rate and gain high speed. For correcting the division error compare operation and a memory look up is required for the each operand is required which increases the time delay for entire multiplication process [4]. At various level of abstraction including circuit, logic and architecture levels the approximation is processed [5]. In the category for approximation methods in function, a number of approximating arithmetic building blocks, such as adders and multipliers, at different design levels have been suggested in various structures [6],[7]. Broken array multiplier was designed for efficient VLSI implementation[8]. The error of mean and variance of the imprecise model increase by only 0.63% and 0.86% with reverence to the precise WPA and the maximum error increases by 4%. Low-Power DSP uses approximate adders which are employed in different algorithms and design for signal processing. In contrast with standard multiplier, the dissipated power for the ETM dropped from 75% to 90%. While maintaining the lower average error from the conventional method, the proposed ETM achieves an impressive savings of more than 50% for a 12 x 12 fixed-width multiplication.

ROBA MULTIPLIER DESCRIPTION

The motive behind this approximate multiplier is to make use of the ease of operation of power n (2n) .To elaborate on the process of the approximate multiplier, first, let us denote of the input of A and B rounded value by Ar and Br, respectively. The multiplication of A by B can be write as

$$A \times B = (Ar - A) \times (Br - B) + Ar \times B + Br \times A - Ar \times Br \quad \text{----1}$$

Key observation is to facilitate the multiplications of Ar * Br, Ar * B, and Br X A may be implemented just by the operation of shifting which is publicized in the eqn (1). The hardware implementation (Ar – A) X (Br – B), however, is rather complex. The weight of this term in the concluding result, depends on differences of the exact numbers from their rounded ones, is typically small. Hence, it is proposed to omit this part from (Ar -A) x (Br – B), helping simplify the multiplication operation shown in the eqn (2). Hence, to perform the multiplication process, the following expression is used

$$A \times B = Ar \times B + Br \times A - Ar \times Br \quad \text{----2}$$

While both values lead to same effect on the accuracy of the multiplier, selecting the larger one (expect for the value p=2) leads to a smaller hardware implementation for determining the nearest rounded value. It originates from the detail that the number in the composition of 3 × 2 p−2 considered as do not care in the both rounding process up and down manner, and smaller logic expressions may be achieved. With the help of accurate and approximate equation the proposed architecture can be designed. Fig 1 provides the detail block diagram for the RoBA multiplier which is applicable for the two processing such as unsigned multiplication, signed multiplication

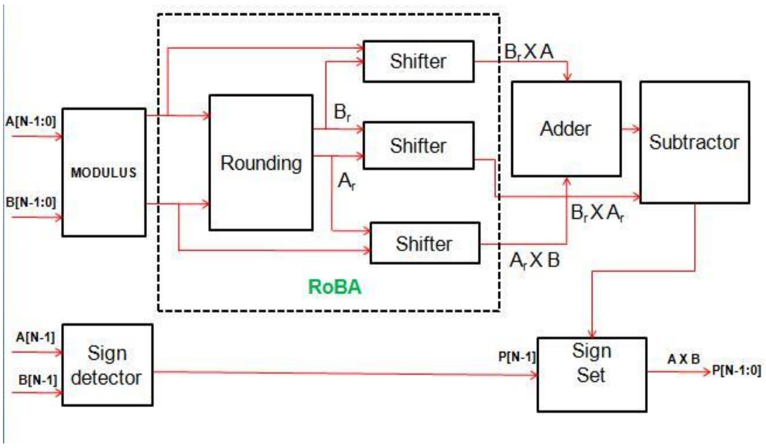


Fig.1 *Block Diagram for RoBA Multiplier*

If the operation is for unsigned multiplication the sign detector and sign set is disabled which can speed up the multiplication process. The two inputs are provided to the detector block which detects MSB of the input and it is provided to the sign set block to denoted signed or unsigned multiplication. Rounding and shifter are worn to reduce the operands value to the nearest power of 2 and it can be shifted with the help of barrel shifter. There are 3 levels of shifter for the following terms obtained in the approximate equation. The kongee stone adder is used to add the two functions from the shifter. The sign can be set with the help detector block [9].

If the output is negative the error value is calculated by inverting the output equation and it is added with binary value of 1. It supposed to be noted that contrary to the previous work where the approximate result is lesser than the exact result, the final result calculated by the RoBA multiplier may be either larger or lesser than the exact result depending on the magnitudes of Ar and Br compared with those of A and B , respectively. Note that if one of the operands (say A) is lesser than its equivalent rounded value while the other operand (say B) is larger than its equivalent rounded value, then the approximate result will be larger than the exact result. Because the term (A r − A) × (B r − B) will be neglected. Since the differentiation between (1) and (2) is precisely this product, the approximate result becomes higher than the exact one. Similarly, if both A and B are larger or both are lesser than Ar and Br, then the approximate result is lesser than the exact result. Hence, before the multiplication operation starts, the values of both input are absolutes and the output sign of the result are based on the inputs signs be determined and then the operation be performed for unsigned numbers and, at the last stage, the proper sign be applied to the unsigned result.

STRUCTURE LEVEL DESIGN OF ROBA MULTIPLIER

From the equation 1 and 2 the structure level implementation of the multiplier were designed. The inputs are represented in the format of two's complement. First, the signs of the inputs are determined, and for each negative value, the unconditional value is generated. Next, the rounding block extracts the nearest value for each unconditional value in the form of 2n. The bit width of the output of this block is n (the most significant bit of the absolute value of an n-bit number is zero for two's complement format). To determine the nearest value of input A, the operands are rounding off to the power of 2 with the help of rounding criteria.

There are four cases for selecting final rounded of value from the original input values there are discussed below

1. Ar is high and Br is low.
2. Ar is low and Br is high.
3. Ar is high and Br is high.
4. Ar is low and Br is low.

By selecting the case one, the approximate result is larger when observed with exact result. From the case two and three, the approximate result is somewhat larger than the accurate result in contrast with case one. For case four, the approximate result is lower than the exact result. The program should be slightly modified for each one of the cases. The rate or error is extremely low down for case one and four in contrast with other two cases.

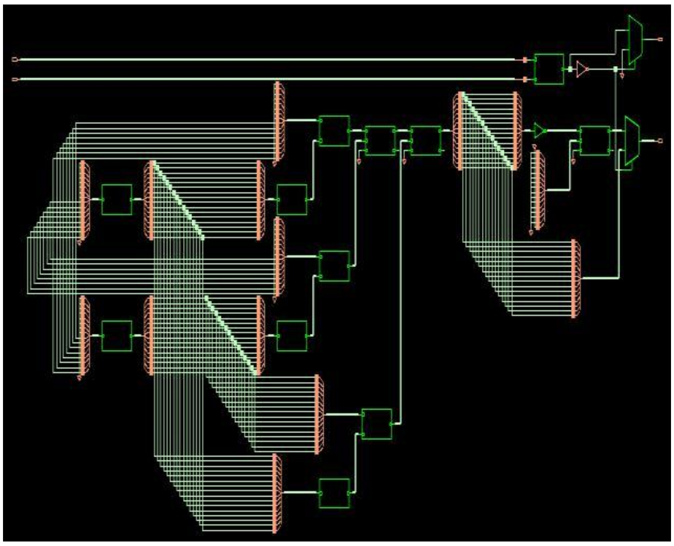


***Fig.2** RTL architecture for ROBA multiplier*

The error rate is the important factors that should be considered while designing the approximate multiplier. The distance between exact and inexact results for the approximate multiplier is calculated before calculating the error rate of the rounding based approximate multiplier. The hardware architectures of the sign detector, rounding, barrel shifter, kongee stone, subtractor and the sign set modules. The RTL architecture for RoBA multiplier is shown in Fig 2 taken by cadence encounter tool 180-nm technology. ~~T~~The sign set block is used to negate the output if the final output is negative valued. To negate values, which have the representation of two's complement, the corresponding circuit based on X+ 1 should be used. To speed up negation operation, one may skip the incrementation process in the negating phase by accepting its associated error. As will be seen later, the impact on the error decreases when an input width increases.

If the negation is performed exactly (approximately), the implementation is called signed RoBA (S-RoBA) multiplier [approximate S-RoBA (AS-RoBA) multiplier]. If the inputs are always positive, to speed up and decrease the power consumption, the sign detector and sign set blocks are omitted from the architecture, providing us with the architecture called unsigned RoBA (U-RoBA) multiplier.

RESULTS AND DISCUSSIONS

In this section, inaccuracies of the three architectures discussed above are considered. The Verilog code was implemented in Xilinx 14.2 software [10]. The inaccuracies of the U-RoBA multiplier and S-RoBA multiplier, which originate from omitting the term (A r − A) × (B r − B) from the accurate multiplication of A × B, are the same. Assuming A_r and B_r are equal to 2^n and 2^m, respectively, the maximum error occurs when A and B are equal to 3 x 2^n and 3 x 2^m, respectively. Hence the error rate for signed approximate RoBA is specified in the eqn (3).

$$\text{Error}(A, B) = \frac{(Ar - A)(Br - B)}{AB} \quad \text{------------} \quad (3)$$

In this case, both Ar and Br have the maximum arithmetic difference from their corresponding inputs rounding which is equal to and 2^m, respectively and their maximum error rate is given in the eqn (4).

$$\text{Max}\{\text{error}(A, B)\} = \frac{(2^n - 3 * 2^{n-2})(2^m - 3 * 2^{m-2})}{(3 * 2^{n-2}) * (3 * 2^{m-2})} \quad \text{------------} \quad (4)$$

For the AS-RoBA multiplier, the error includes the supplementary term due to approximate negation. In the worst case (where both inputs are negative), one may obtain the maximum error from the eqn (5).

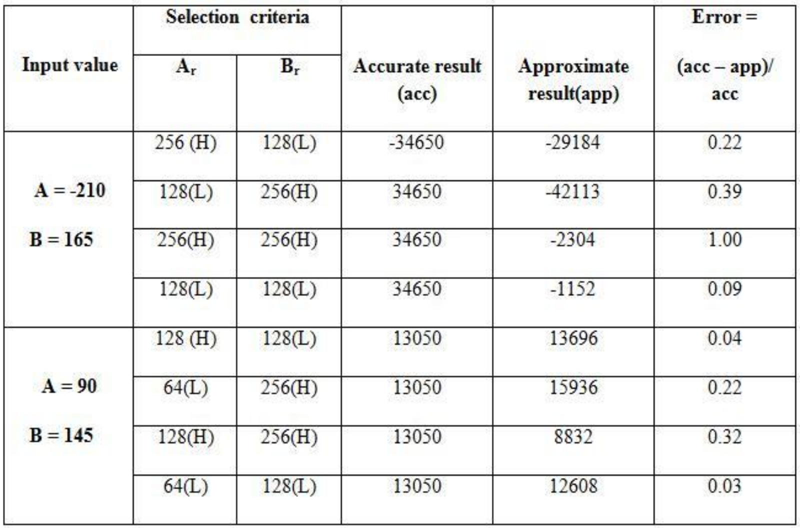
$$\text{Error}(A, B) = (Ar' - A')(Br' - B) + (A' + B' + 1) / AB \quad \text{------------} \quad (5)$$

Compared with above equation the 2nd term comes from the negation approximation taken from the following relation:

$$A \times B = (\bar{A}+1)(\bar{B}+1) = \bar{A}+\bar{B}+1+\bar{A} \times \bar{B} \approx \bar{A} \times \bar{B} \quad \text{------------} \quad (6)$$

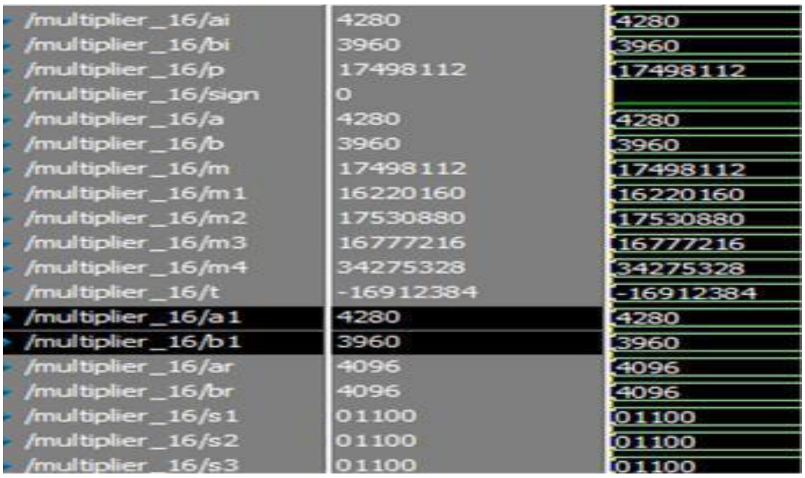
Therefore the eqn (6) shows that the error was A+B+1. If one of the inputs is negative, the AS-RoBA multiplier error is higher than that of the two other RoBA multiplier types. Therefore the maximum error for the U-RoBA and S-RoBA architectures is %11.1, which is same as that of [1]. Also, when both inputs are negative, the final result will be positive; one still needs to negate the negative inputs. Based on this formulation, when one of the inputs is −1, the maximum error, which is 100%, occurs. Finally, for the AS-RoBA multiplier, as mentioned before, the maximum error happens when anyone of the inputs is -1.

Table-I *8-Bit signed multiplication*



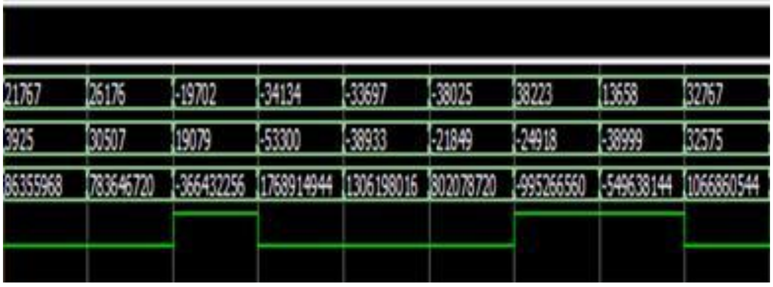
Input value	Selection criteria		Accurate result (acc)	Approximate result(app)	Error = (acc – app)/ acc
	A_r	B_r			
A = -210 B = 165	256 (H)	128(L)	-34650	-29184	0.22
	128(L)	256(H)	34650	-42113	0.39
	256(H)	256(H)	34650	-2304	1.00
	128(L)	128(L)	34650	-1152	0.09
A = 90 B = 145	128 (H)	128(L)	13050	13696	0.04
	64(L)	256(H)	13050	15936	0.22
	128(H)	256(H)	13050	8832	0.32
	64(L)	128(L)	13050	12608	0.03

The simulation process is performed with the help of Quartus tool and the performance characteristics such as area, power and delay are reported through the Cadence tool. The simulation result of the RoBA is shown in following figures 3 and 4. From the simulation diagram the given input vectors A and B are forced to value required in the simulation window.



```
/multiplier_16/ai      4280       4280
/multiplier_16/bi      3960       3960
/multiplier_16/p       17498112   17498112
/multiplier_16/sign    0
/multiplier_16/a       4280       4280
/multiplier_16/b       3960       3960
/multiplier_16/m       17498112   17498112
/multiplier_16/m1      16220160   16220160
/multiplier_16/m2      17530880   17530880
/multiplier_16/m3      16777216   16777216
/multiplier_16/m4      34275328   34275328
/multiplier_16/t       -16912384  -16912384
/multiplier_16/a1      4280       4280
/multiplier_16/b1      3960       3960
/multiplier_16/ar      4096       4096
/multiplier_16/br      4096       4096
/multiplier_16/s1      01100      01100
/multiplier_16/s2      01100      01100
/multiplier_16/s3      01100      01100
```

Fig.3 *simulation result for 8-bit multiplier*

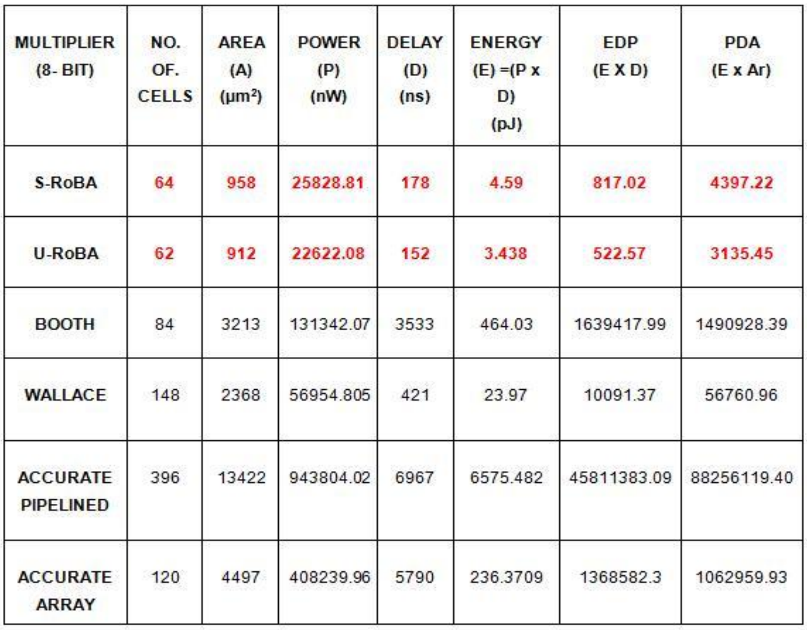


```
21767     26176     -19702     -34134      -33697      -38025     38223      13658      32767
3925      30507     19079      -53300      -38933      -21849     -24918     -38999     32575
86355968  783646720 -366432256 1768914944  1306198016  802078720  -995266560 -549638144 1066860544
```

Fig.4 *simulation result for 16-bit RoBA operation*

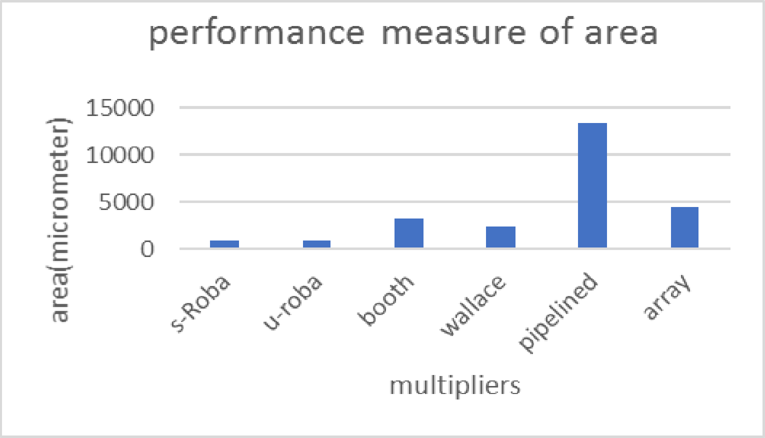
The area, power and delay are calculated for different approximate and accurate multiplier in cadence encounter tools.

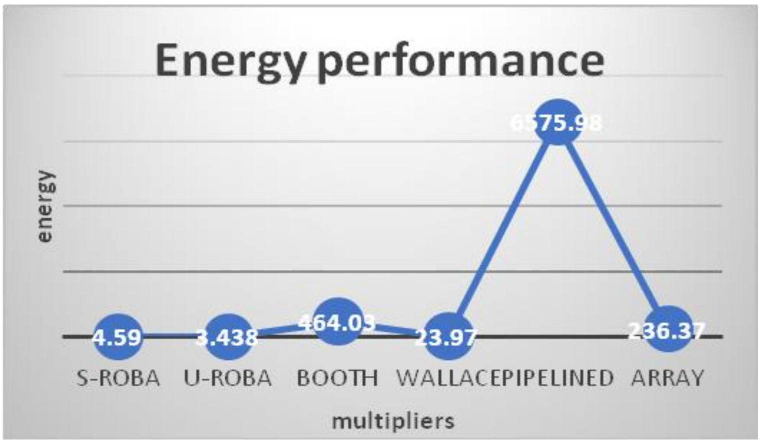
Table – II *Performance Analysis For Various 8-Bit Approximate And Accurate Multipliers*

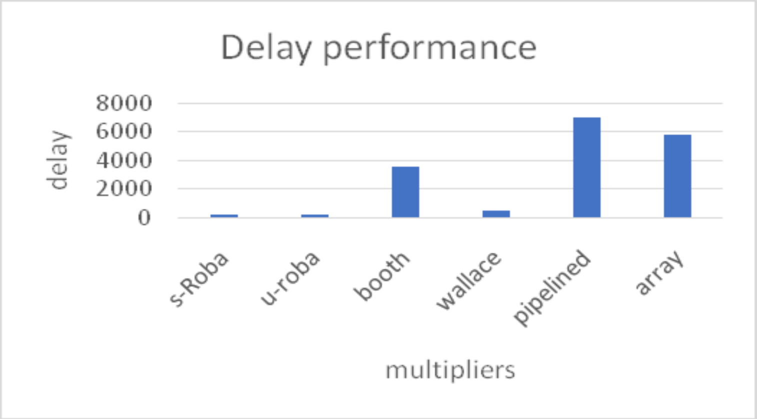


MULTIPLIER (8- BIT)	NO. OF. CELLS	AREA (A) (μm^2)	POWER (P) (nW)	DELAY (D) (ns)	ENERGY (E) =(P x D) (pJ)	EDP (E X D)	PDA (E x Ar)
S-RoBA	64	958	25828.81	178	4.59	817.02	4397.22
U-RoBA	62	912	22622.08	152	3.438	522.57	3135.45
BOOTH	84	3213	131342.07	3533	464.03	1639417.99	1490928.39
WALLACE	148	2368	56954.805	421	23.97	10091.37	56760.96
ACCURATE PIPELINED	396	13422	943804.02	6967	6575.482	45811383.09	88256119.40
ACCURATE ARRAY	120	4497	408239.96	5790	236.3709	1368582.3	1062959.93

The U-RoBA, S-RoBA multiplier are compared with the some of the exact and in-exact multipliers like Booth, Wallace, accurate pipelined, accurate array multiplier in stipulations of logic cells, area, power, energy, delay, power delay area and energy-delay product. The performance analysis for various multipliers mentioned above is performed in cadence encounter tool in 180nm technology. The unsigned and signed multiplier using rounding based approximation provided the efficient energy when compared to all the other multiplier techniques. The overall energy for the S-RoBA and U-RoBA multiplier achieved 4.59 pJ and 3.43 pJ respectively which are dyed by the red mark in table II. Therefore the 8-bit approximate S-RoBA and U-RoBA multipliers are provided high efficiency by comparing the all other 8-bit accurate and approximate multiplier in the cadence 180-nm technology.







CONCLUSION AND FUTURE SCOPE

High-speed and energy efficient approximate multiplier were proposed. The RoBA multiplier had a high accuracy depend upon the 2n input form. The high exhaustive computation part is neglected to provide high performance. So hardware structural design is designed for S-RoBA, RoBA and AS-RoBA multiplier. The efficiencies of the RoBA multiplier were compared with some existing accurate and approximate multipliers with different parameters. With the help of comparison table, RoBA multiplier provides the better area, power, and energy efficient when compared with some already proposed accurate and approximate multiplier.

The negation of output bit in the signed multiplier causes the maximum error rate. Therefore in future, the maximum error rates can be compact by minimizing the error rate equation which is identical to the error rate equation of unsigned RoBA multiplier. In future, the proposed multiplier will be applied in the various images processing applications.

REFERENCES

1. Alioto, M., *Ultra-Low Power VLSI Circuit Design Demystified and Explained: A Tutorial.* IEEE Transactions on Circuits and Systems I: Regular Papers, 2012. **59**(1): p. 3-29.
2. Mahdiani, H.R., et al., *Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications.* IEEE Transactions on Circuits and Systems I: Regular Papers, 2010. **57**(4): p. 850-862.
3. R. Venkatesan, A.A., K. Roy, and A. Raghunathan, *MACACO: Modeling and analysis of circuits for approximate computing.* Int. Conf. Comput.-Aided Design, 2011: p. 667–673.
4. Mitchell, J.N., *Computer multiplication and division using binary logarithms*, in *IRE Trans. Electron. Comput*. 1962. p. 512–517.
5. Gupta, V., et al., *Low-Power Digital Signal Processing Using Approximate Adders.* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2013. **32**(1): p. 124-137.
6. Kulkarni, P., P. Gupta, and M. Ercegovac, *Trading Accuracy for Power with an Underdesigned Multiplier Architecture*, in *2011 24th Internatioal Conference on VLSI Design*. 2011. p. 346-351.
7. K.Y. Kyaw, W.L.G., and K. S. Yeo, *Low-power high-speed multiplier for error tolerant application.* IEEE Int. Conf. Electron Devices Solid-State Circuits (EDSSC), 2010: p. 1–4.
8. F. Farshchi, M.S.A., and S. M. Fakhraie, *New approximate multiplier for low power digital signal processing.* 17th Int. Symp. Comput. Archit. Digit. Syst. (CADS), 2013: p. 25–30.
9. R.Dubey , J.J., *An efficient Processing by Using Kongee Stone High Speed technique.* International Journal of Computer Applications (IJCA), 2015: p. 21-23.
10. Palnitkar, S., *Verilog HDL: A Guide to Digital Design and Synthesis*, ed. S. edition. 2003: Prentice-hall PTR.