IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

DESIGN OF OPTIMIZED-DYNAMIC RANGE UNBIASED MULTIPLIER(DRUM) FOR APPROXIMATE MULTIPLICATION

[1]Mr. Akash K,[2]Ms.K Parvadha
[1,2]Teaching assistant,,M.E, Microelectronics,
Department Of Electronics Engineering
Birla Institute Of Technology and Science, Hyderabad, India

Abstract: The digital world we live in demands a high application of digital signal processing, neural networking, computer automation. Multiplier is one of the essential operators for these applications. The demand for power efficiency and higher speed introduces errors into these applications. These demands give birth to a new field known as Approximate computing. In the paper approximate multiplier is designed with a huge range of dynamic selection. One of the main advantages of the design is the error introduced during a computation, which cancels with previous computation instead of summing up. An approximate DRUM multiplier is designed using Verilog. Register Transfer Level (RTL) is retrieved. The computation of error metrics is done in MATLAB. This design is scalable and so it can be used according to the required application. The propagation delay of the multiplier is reduced in this design which makes it faster. Error analysis is done based on various parameters and it is observed the error introduced is least among all the approximate multipliers. Since multiplier consumes more power in the processor, this multiplier is designed with power efficiency as one of the major constrain. As designed the multiplier when used in a processor saves power to 58 %. The design outperforms various other approximate multipliers which was analyzed during the literature survey. After analyzing the DRUM multiplier a modification is done in the design to improve the accuracy. This optimized design is termed as Optimized DRUM multiplier and it is analyzed and the comparative study between both the multipliers is made in terms of accuracy.

Index Terms -**Approximate computing, error metrics, Dynamic Range, RTL schematic.**

I. INTRODUCTION:

Some applications like wireless communication, data mining can tolerate errors to some extent. The error tolerance level varies based on various parameters like noisy input, data redundancy. The main demand for many designs turns out to be a higher speed. Another important parameter that plays a vital role in the design is power consumption. Approximate computing provides the option for the users to trade-off among all these parameters. A simple approach for analyzing approximate computing is to design the approximate circuits and the error percentage is controlled based on the design of the modules in the circuit. These approximate blocks are then introduced in the main computational unit of the system. Approximate arithmetic tells us that any basic building block in a system can be made approximate which in turn reduces the power by a huge amount and the propagation delay.

In this paper the design of the approximate multiplier is done. Since the multiplier is the basic unit of many processors and it occupies more silica area and consumes more power. Thus, it turns out to be an impressive idea to approximate the multiplier. A Dynamic Range Unbiased Multiplier (DRUM) is designed to meet the approximate applications. The unbiased design of the multiplier gives the advantage of error cancelation instead of error accumulation in the output. Another important advantage is that trade-off can be made between accuracy and power loss by the user. An analytical formula is used to calculate the average error and maximum error. The formula is based on the parameters which determine the configuration of the multiplier. The multiplier inaccuracy, power consumption, timing analysis is done as a function of the number of bits truncated from the inputs of the multiplier. After all the analysis, even more, modification is done in the design of the existing DRUM multiplier to improve the accuracy which is termed ad Optimised DRUM multiplier. The error matrix calculation, power analysis, timing analysis is performed for this design as well. The results are compared with the DRUM multiplier.

The works done in the paper are as follows. A substantial amount of papers has been referred on the topics of Approximate multiplier, approximate computing, types of multiplier, and discussed in the related work section. In the section of proposed work, the proposed multiplier is described elaborately. To increase the accuracy further few changes have been made to the multiplier and the new design is termed as **Optimized DRUM multiplier**. The brief working of this multiplier is discussed in the section of the optimized DRUM multiplier. MATLAB codes are written for both the multipliers and their error metrics are calculated by performing numerous iterations and by varying the parameters which affect the accuracy. The analysis of both the multiplier and the comparison study between them is made. This is discussed in the Results and Analysis section. Further, a conclusion is drawn from various hypothesis that was taken at the initial stage of projects and is presented

II. RELATED WORKS:

In this section some researches in the field of approximate multiplier and approximate computing is done.

Gupta et al. [1] introduced various approximate adders, multipliers by removing some of the logics in the blocks which introduces some error but the area and power consumption are reduced drastically. Cong Liu et al. [2] discussed various types of multipliers and their performances. Array multiplier is explained followed by carry save multiplier which has lesser propagation delay. To decrease the area Wallace tree multiplier and its approximation is discussed. For multiplication of negative number booth multiplier and its approximation is discussed. Finally, for multiplication of decimal numbers floating point multiplier and logarithmic multiplier is explained. Mahadiani et al. [3] suggested an approximate adder by introducing OR gates for computation of sum for the lower bits of the operands, which results in a significant reduction in area. New methodologies for approximations have been discussed in [4],[5], [6].

Multiplier with error tolerance has been discussed in [7] where the multiplier has two modules. One module computes the exact multipion and the other the approximate value. To increase the accuracy an iterative approach is followed in [8]. Mitchel multiplier et al. suggest a multiplier that calculates the partial product with the help of fast adders. Kalkurni et.al [9] suggest the multiplier which computes the exact multiplication value for the MSB and the approximate results for the LSB. Narayanamoorthy et.al [10] introduced the multiplier based on truncation the upper middle and lower parts are truncated and the results are calculated separately which is then added together to form the output. One of the disadvantages in [11], [12] is the number of bits to be truncated is not mentioned which will be corrected in the design of the DRUM multiplier. A recent approach for approximation is on the generation of approximate logics which is discussed in [13],[14]. In this way approximate circuits are synthesized automatically and they are developed as hardware.

III. PROPOSED WORK:

The main idea of the design is that not all the bits of the operands are equally important. So, the number of bits to be multiplied exactly is decided by the user which plays a vital role in the accuracy. This idea helps in a large area reduction as the full multiplier is replaced by a small core multiplier. In the design the number of bits is selected using a dynamic bit selection scheme. The design uses Leading One Detector (LOD) which detects the most significant bit with the value of '1' in both the operands. After the deduction of leading one, the number of bits of the operands which are to be multiplied exactly is chosen. This is elaborated with the following example:

Let us assume an 'n' bit number is passed through the LOD block Then after the deduction the number of least significant bits to be truncated is decided which tells the size of an accurate multiplier. This is illustrated in the Fig1.

0	0	1	0	1	0	x	x	x	x

(a)

1	0	1	0	x	x	x	x

(b)

1	0	1	1

(c)

***Fig. 1**. (a)Original given number (b) Number after the detection of Leading one (c) Number after truncation of the bits.*

The reason for the DRUM multiplier to be successful in comparison with other approximate multipliers in terms of error percentage which is decided by the truncation part design. In many approximate multipliers after truncation, the last bit of the biased operand is retained as its original value but in the design of the DRUM multiplier the last bit is changed to the value of '1'.This technique reduces the error significantly from 1.86 % to 0.26%. The truncation technique is explained in Fig 1. After truncation of both the operands are multiplied using a small exact core multiplier and the result is obtained. This exact result is shifted and the shift is decided by the LOD block and the no of bits truncated in both the operands. This is illustrated in Fig 2.

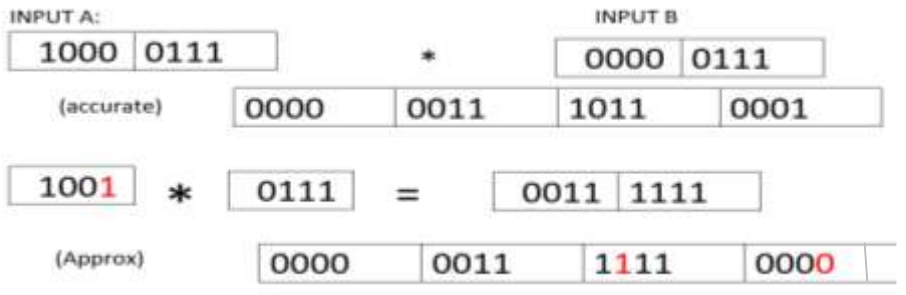


***Fig. 2** Numerical example of DRUM multiplier*

The multiplier proposed has two major logics namely steering logic and computational logic. The steering logic is responsible for the selection of the operands and feed those in the accurate multiplier. The computational logic is responsible for the accurate multiplication. Therefore, steering logic includes multiplexers, LOD, encoders, and barrel shifters. The computational unit consists of an array multiplier which computes the exact multiplication of the values after truncation. The inclusion of the steering part reduces the power and area significantly. One of the most desirable advantages of the design is the steering part is user-defined, thus user can decide the number of bits to be truncated. The hardware used and the schematic diagram are shown in Fig 3.

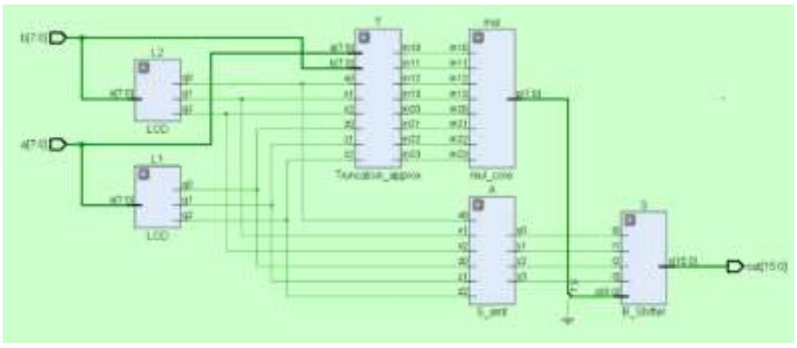


Fig. 3 *Schematic diagram of the DRUM multiplier*

The leading one detector says the bit position where the leading '1' is present. This value also decides the number of bits to be shifted after accurate multiplication. The number after unbiasing is given as input to the encoder. The encoder decides the number of bits to be truncated and the truncated number undergoes exact multiplication. The exact multiplication is done by an array multiplier. The results from the array multiplier is passed through the barrel shifter. The barrel shifter computes the number of bits to be shifted with the help of LOD and encoder blocks. The output of the DRUM multiplier is obtained from the barrel shifter. The result obtained is compared with the accurate multiplier and the error percentage is calculated. In comparison with the accurate multiplier, the DRUM multiplier saves a huge amount of power consumption, area reduction, and speed improvement.

IV. OPTIMIZED DRUM MULTIPLIER:

After analyzing DRUM multiplier working and error analysis completely it is clear that a still more improvement in the DRUM multiplier can be done to increase the accuracy. The improvement is done in the truncation part of the design. Instead of simply making the last bit of the biased operand '1' a sensible approach is implemented. In the hardware perspective the encoder of the DRUM multiplier is altered other modules remaining the same. An additional shifter and comparator are required to design the new logic of the encoder. Instead of simply making the last bit of biased operands as '1'. The proposed encoder design makes sure the last bit is set as '1' if the truncated bits are greater than half of their maximum value else it is set to 'zero'. This is illustrated in Fig 4.

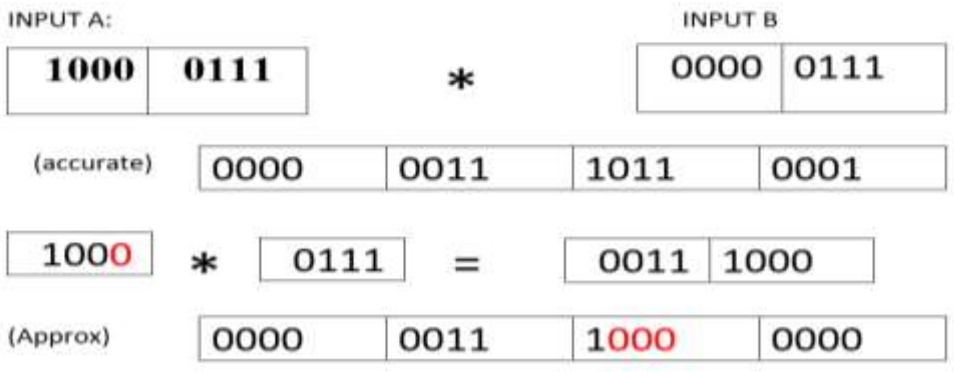


Fig 4*. Numerical example of Optimized Drum multiplier.*

V. RESULTS AND ANALYSIS:

The 8 bit approximate Dynamic Range Unbiased Multiplier and the suggested optimization is implemented in MATLAB version-**9.6.0.1072779 (R2019a) and its error metrics are studied.** The following are the error metrics that are used for the analysis: Average Error, NMED, MRED and Error rate
Let O be the approximate result and M be the accurate result,
Error distance (ED): The Error distance (ED) is calculated as

$$ED = |O^1 - M|$$

Relative error distance (RED):
The Relative error distance (RED) is calculated as shown,

$$RED = \frac{ED}{M}$$

Mean error distance (MED):
The mean error distance (MED) is the mean of all possible EDs.

Normalized MED (NMED):
The normalized MED (NMED), is the normalization of MED by the maximum output of the accurate design.

Mean relative error distance (MRED):
The mean relative error distance (MRED) is the average value of all possible REDs which are used to assess the error characteristics of the approximate designs.

Average error: The average error the mean of all possible errors (O − M) is used to evaluate the bias of an approximate arithmetic design. The average error, error rate MRED and NMED are calculated for the proposed approximate multiplier (with and without optimization) and the result is as follows

Table 1. Error Metrics of Drum multiplier

K	NMED	MRED	ERROR RATE	AVG ERROR
2	0.0716	0.3435	0.9990	-2.2459e+03
3	0.0535	0.3036	0.9971	-2.0155e+03
4	0.0660	0.3533	0.9966	-2.3163e+03
5	0.0845	0.4232	0.9926	-2.9900e+03

Table 2. Error Metrics of Optimized DRUM multiplier

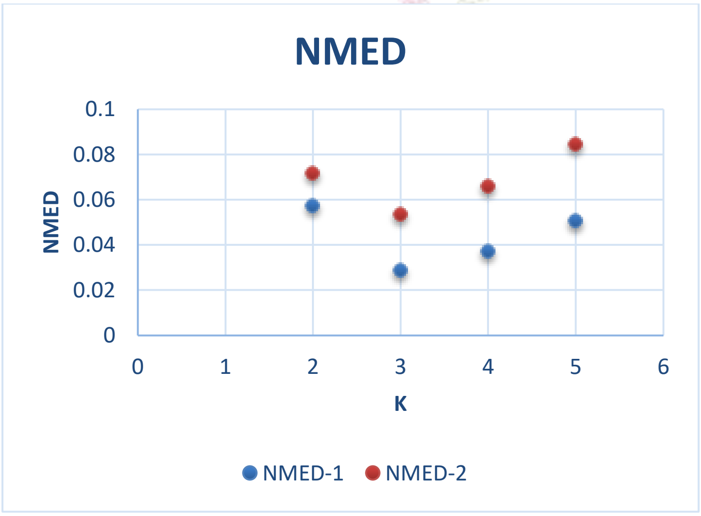
K	NMED	MRED	ERROR RATE	AVG ERROR
2	0.0572	0.2569	0.9990	127.7461
3	0.0287	0.1267	0.9970	127.7461
4	0.0371	0.1561	0.9954	127.7461
5	0.0505	0.2107	0.9915	127.7461

COMPARISON OF ERROR METRICS OF DRUM MULTIPLIER AND OPTIMIZED DRUM MULTIPLIER:

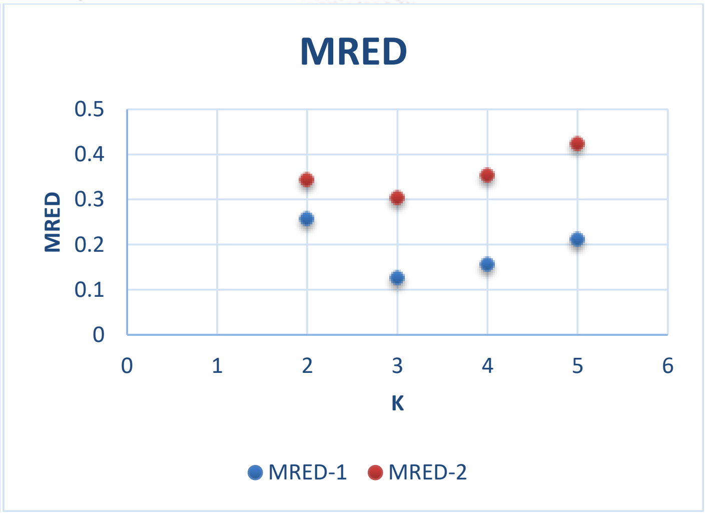
The following are the variations of NMED and MRED for various values of k for both optimized and non-optimized circuits, here the NMED-1 and MRED-1 respectively correspond to the optimized multiplier and NMED-2 and MRED-2 to non-optimized. We can as well see a dip in the curve at k=3, thus at k-3 the NMED and MRED both are minimum, thus it is the optimum k value

In the following graph (1) represents the DRUM multiplier

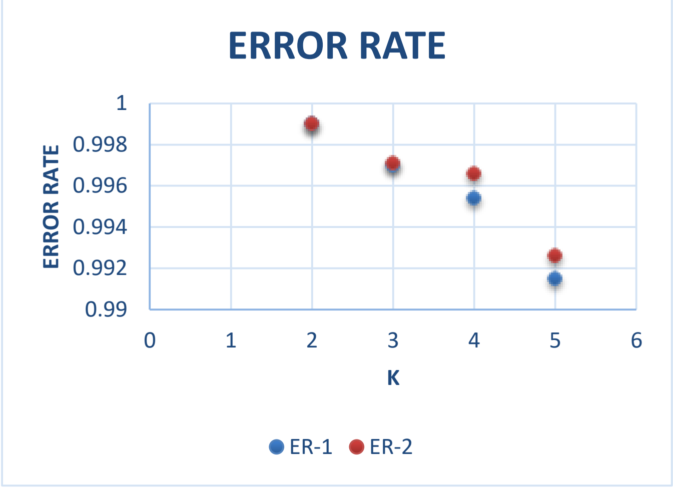
(2) represents the optimized DRUM multiplier.



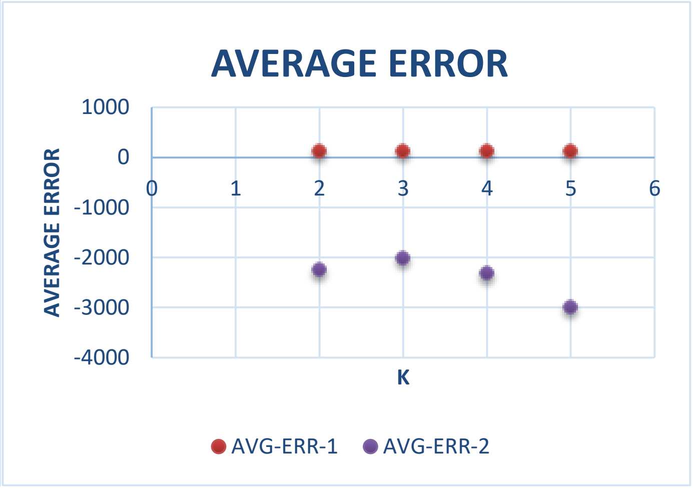
(a)



(b)



(c)



(d)

***Fig. 5** Comparison of conventional DRUM and Optimised DRUM Multiplier (a)NMED (b)MRED (c)Error Rate (d)Average error*

VI. CONCLUSION:

Upon analyzing various approximate arithmetic circuits DRUM was selected analyzed and an optimization of the same is proposed and implemented that can be used in applications like JPEG compression, perceptron classifier, etc.,

- We observe that upon varying the k values k=3 seems to be having less error compared to other circuits hence we consider that to be optimum for an 8-bit DRUM multiplier.
- We also see that the error of the optimized multiplier is lesser than the drum1 Multiplier.
- We also noticed that both the DRUM multipliers save area compared to the original multiplier, yet the optimized multiplier uses more area as a trade-off for accuracy of the result.

It is seen that the proposed multiplier gives an average error of around 127.74 and NMED of 0.0287 and MRED of 0.1267 which is comparable with the existing approximate multipliers.

VII. REFERENCES

[1] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, “Low-power digital signal processing using approximate adders,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 1, pp. 124–137, 2013.

[2] Cong Liu, H.jiyang,” A Review, Classification, and Comparative Evaluation of Approximate Arithmetic Circuits”. ACM Journal on Emerging Technologies in Computing Systems, Vol. 13, No. 4, Article 60, Publication date: August 2017.

[3] H. Mahdiani, A. Ahmadi, S. Fakhraie, and C. Lucas, “Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications,” IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 4, pp. 850–862, 2010.

[4] J. Liang, J. Han, and F. Lombardi, “New metrics for the reliability of approximate and probabilistic adders,” IEEE Transactions on Computers, vol. 62, no. 9, pp. 1760–1771, 2013.

[5] J. Miao, K. He, A. Gerstlauer, and M. Orshansky, “Modeling and synthesis of quality-energy optimal approximate adders,” in IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2012, pp. 728–735.

[6] J. Huang, J. Lach, and G. Robins, “A methodology for energy-quality tradeoff using imprecise hardware,” in ACM J. N. Mitchell, “Computer multiplication and division using binary logarithms,” IRE Transactions on Electronic Computers, vol. EC-11, no. 4, pp. 512–517, 1962.

[7] K. Y. Kyaw, W. L. Goh, and K. S. Yeo, “Low-power high-speed multiplier for error-tolerant application,” in IEEE International Conference of Electron Devices and Solid-State Circuits (EDSSC), 2010, pp. 1–4. Proceedings of the 49th Annual Design Automation Conference (DAC), 2012, pp. 504–509.

[8]C. Liu, J. Han, and F. Lombardi, “A low-power, high-performance approximate multiplier with configurable partial error recovery,” in Proceedings of the Conference on Design, Automation & Test in Europe(DATE), 2014, pp. 95:1–95:4.

[9]P. Kulkarni, P. Gupta, and M. Ercegovac, “Trading accuracy for power with an underdesigned multiplier architecture,” in 24th International Conference on VLSI Design, 2011, pp. 346–351.

[10] S. Narayanamoorthy, H. Moghaddam, Z. Liu, T. Park, and N. S.Kim, “Energy-efficient approximate multiplication for digital signal processing and classification applications,” IEEE Transactions on Very Large Scale Integration Systems, vol. 23, no. 6, pp. 1180–1184, 2015.

[11] M. S. Lau, K.-V. Ling, and Y.-C. Chu, “Energy-aware probabilistic multiplier: Design and analysis,” in ACM Proceedings of the International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, 2009, pp. 281–290.

[12] K. Palem, “Energy-aware computing through probabilistic switching: a study of limits,” IEEE Transactions on Computers, vol. 54, no. 9, pp. 1123–1137, 2005.

[13] V. Chippa, D. Mohapatra, A. Raghunathan, K. Roy, and S. Chakradhar, “Scalable effort hardware design: Exploiting algorithmic resilience for energy efficiency,” in 47th ACM/IEEE Design Automation Conference(DAC), 2010, pp. 555–560.

[14] S. Venkataramani, A. Sabne, V. Kozhikkottu, K. Roy, and A. Raghunathan, “Salsa: Systematic logic synthesis of approximate circuits,” in 49th ACM/EDAC/IEEE Design Automation Conference (DAC), 2012, pp. 796–801.