*International Journal on Applications in Electrical and Electronics Engineering*
*Volume 7 : Issue 2 : August 2021, pp 19 – 23 www.aetsjournal.com* ISSN (Online) : 2395-3527

# Measurement of Power Area and Delay Efficient By Using Approximate Multiplier

Mr.R.Sivakumar AP / EEE, Nehru Institute of Engineering and Technology
Mr.D.Saravanakumar AP / EEE, Nehru Institute of Engineering and Technology

***Abstract*** — For the energy-efficient multiply-accumulate (MAC) processing, we introduce a unique approximate computing approach in this study. In order to generate mistakes in the opposite direction while minimizing computing costs, we first develop approximate 4-2 compressors. Positive and negative multipliers are then meticulously built based on the probabilistic analysis to produce a comparable error distance. The proposed MAC processing provides the energy-efficient computing scenario by expanding the range of approximative portions, according to simulation results on various real-world applications. The low-energy, MAC-intensive algorithms are created by this work, which brand-new introduces the advanced interleaving technique for the balanced error accumulation. We created two different types of approximate multipliers with opposing error directions based on the previous compressor-based approximation. To be more specific, when designing compressors, we simply take into account the direction of the faults, and each approximative multiplier is built to have the lowest hardware cost possible. Verilog HDL is used to implement this design, and Model Sim 6.4 c is used to simulate it. The Synthesis Process tool from Xilinx measures performance.

***Keywords***— Approximation, Low power, Low error, Multiplier.

## I. INTRODUCTION

Data mining and multimedia signal processing are some of the examples which are error tolerant. For this type of applications exactness of the output is not necessary. Their computational circuits can be replaced by their approximate counter parts in order to improve performance. This gives the rise to designing of approximate adders and multipliers which are very essential for digital circuits. In [1], approximate multiplier is designed with a mechanism of removing partial product generation and accumulation for lower order input operands. Depending upon the input operands the non-multiplication part will produce the output. For higher order input operands normal multiplication is performed. Truncating output [2] by one bit in the case of a multiplier is another approximation strategy because the likelihood of using that output is quite low, but approximation will result in cheap power and high-speed operation. A roughly 4:2 compressor and Wallace tree multiplier are introduced in [3], which takes a somewhat similar technique.

For sum and carry generation in [4], full adders are approximated at the transistor level. Any multiplier can be used with this circuit during the stage of partial products accumulation. The approximation and error correction combining structure is detailed in [5], where approximation is accomplished by interchangeably using the preprocessed input data bits. For the purpose of applying error-resistant systems, truncation and approximation are used in the multiplier design [6] for lower order input bits. The disadvantage of [7]'s multiplier is that it is approximated by a 4:2 compressor and has non-zero output even when all inputs are zero. A static segment multiplier that multiplies a chosen segment of the input operands rather than the entire set is proposed in [8]. In [9], power is lowered with time violations and voltage over scaling effect with circuit delay is employed as an approximation. Partial product perforation is used in [10] to omit some partial products during the generation stage using an approximative multiplier design.

Error metrics like error distance (ED), which is the distinction between the normal and approximate output, are discussed in [11]. Paraphrase. Section III Array multiplier is a hardware multiplier design

Mrs.Jaisakthi.D PG Student, M.E - VLSI, Tagore Institute of Engineering and technology, Deviyakurichi, TK, Attur, Tamil Nadu. ( Mail id: Jaisakthidevaraj@gmail.com )
Mrs. R.Gowthami Assistant Professor, Department of ECE, Tagore Institute of Engineering and technology, Deviyakurichi, TK, Attur, Tamil Nadu. ( Mail id: gowthami.ece@tagoreiet.ac.in )

calculation. Section IV provides result analysis of the approach.

## II. PROPOSED ARCHITECTURE

***Three processes make up the approximation multiplier design: partial product production, reduction of partial products, and vector combine expansion. The final product is produced from the last rows produced by the reduction tree. The second stage is mostly responsible for power consumption, whereas the third stage is primarily responsible for delay. To lower the power, a reduction tree was approximated. The performance parameters of the system are further enhanced by a quicker adder technique and multi-Vt based implementation.***

The proposed methods are applied to an 8-bit unsigned DADDA multiplier. The partial products are often the result of an AND operation between the input operands, zm, n = xmyn, where m and n are the bit locations, when two input operands, x and y, are taken into consideration. The likelihood of the partial product zm, n being 1 is 1/4. The columnar format of the partial products in the DADDA architecture is based on their binary weights. In each column with more than three products, propagate and generate signals were used in place of the product words as given in Eq. 1,

$$p_{m,n} = z_{m,n} + z_{n,m}$$

The probability of being 1 for $p_{m,n}$ is 7/16 whereas for $g_{m,n}$ is 1/16. This difference is resulted into different approach for their reduction approach.

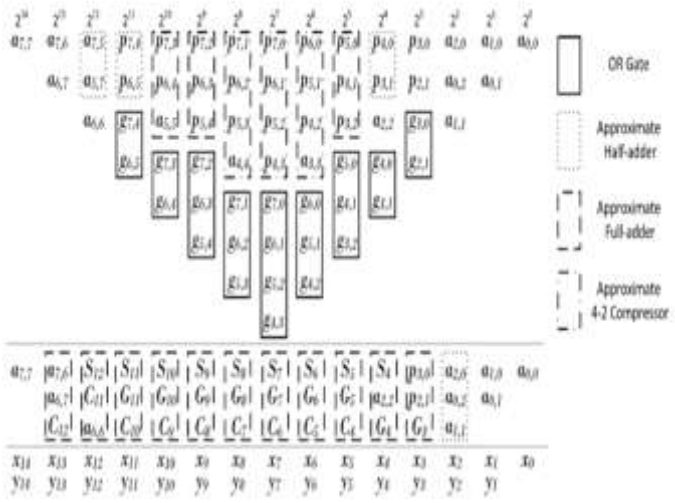


**Figure 1. Array Multiplier Using Proposed Designs**

## III. ARRAY MULTIPLIER

Array multiplier is a hardware multiplier design invented by computer scientist Luigi Array in 1965. It is similar to the Wallace multiplier, but it is slightly faster (for all operand sizes) and requires fewer gates (for all but the smallest operand sizes).

In fact, Array and Wallace multipliers have the same 3 steps:

1. Multiply (logical AND) each bit of one of the arguments, by each bit of the other, yielding results. Depending on position of the multiplied bits, the wires carry different weights, for example wire of bit carrying result of is 32.
2. Reduce the number of partial products to two by layers of full and half adders.
3. Group the wires in two numbers, and add them with a conventional adder.

However, unlike Wallace multipliers that reduce as much as possible on each layer, Array Multiplier do as few reductions as possible. Because of this, Array Multiplier have a less expensive reduction phase, but the numbers may be a few bits longer, thus requiring slightly bigger adders.

To achieve this, the structure of the second step is governed by slightly more complex rules than in the Wallace tree. As in the Wallace tree, a new layer is added if any weight is carried by three or more wires. The reduction rules for the Array tree, however, are as follows:

- Take any three wires with the same weights and input them into a full adder. The result will be an

output wire of the same weight and an output wire with a higher weight for each three input wires.

- If there are two wires of the same weight left, and the current number of output wires with that weight is equal to 2 (modulo 3), input them into a half adder. Otherwise, pass them through to the next layer.
- If there is just one wire left, connect it to the next layer.

### *1) MODULE EXPLANATIONS:*

#### A. PARTIAL PRODUCT GENERATION BLOCK

A 8-bit unsigned1 multiplier is used for illustration to describe the proposed method in approximation of multipliers. Consider two 8-bit unsigned input operands

$$a = \sum_{m=0}^{7} \alpha_m 2^m \text{ and } \beta = \sum_{n=0}^{7} \beta_n 2^n.$$

The partial product am,n = αm · βn in Fig. 1 is the result of AND operation between the bits of αm and βn.

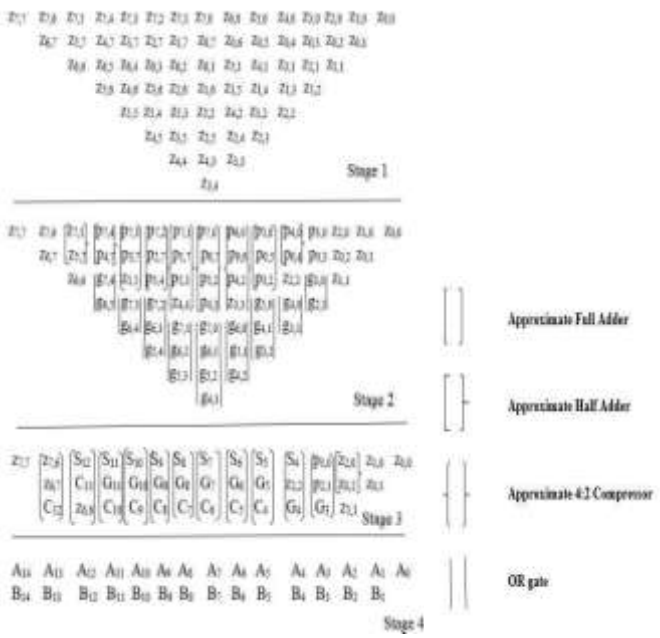


**Figure 2 Partial Product Processing**

#### B. Carry select adder (CSA) using binary to excess 1 converter (BEC)

Carry select adder has a higher speed of operation because of its parallel operation uses two ripple carry adder (RCA). Each RCA uses different carry values as there is only two possibilities i.e. 1 and 0. But it has higher area and power overhead. BEC is nothing but adding 1 to binary bit. So the requirement of RCA with carry input as 1 in CSA architecture is replaced by BEC. BEC structure used in CSA should have 1 bit higher structure than the RCA used. A 4-bit BEC expressions is shown in Eq. 5.

$$B0 = \sim I0$$

$$B1 = I0\ I1$$

$$B2 = I2\ (I1.\ I0)$$

$$B3 = I3\ \ (I2.\ I1.\ I0) \quad (5)$$

Figure 2 is showing the used 16-bit CSA-BEC architecture for 16-bit addition required for implementing the 8x8 multiplier.

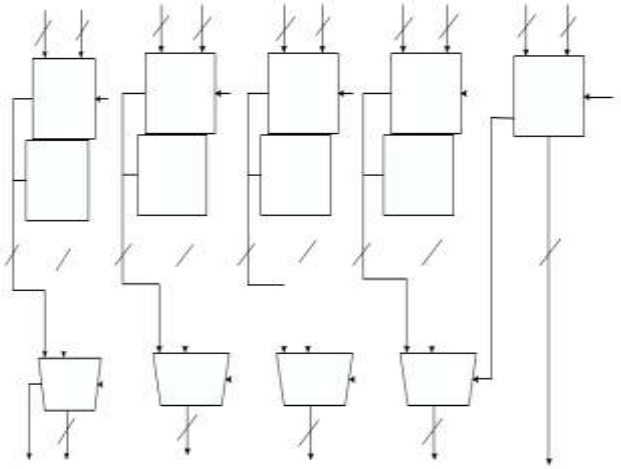

**Figure 3 16-bit CSA-BEC Architecture**

## IV. RESULTS AND DISCUSSION

8 bit multiplier is implemented using Verilog HDL and synthesized using Synopsys design compiler and SAED 32 nm standard cell library at typical process corner with supply voltage of 1.05 and temperature 25oc. For comparison purpose four multiplier structures are taken which are exact multiplier, approximate multiplier with RCA, CSA, and modified CSA with BEC. The simulated waveform from Verilog compiler will show the functionality of the multiplier as shown in Fig. 3. Both exact and approximate outputs are shown for the comparison of the outputs to figure out the deviations due to approximation.

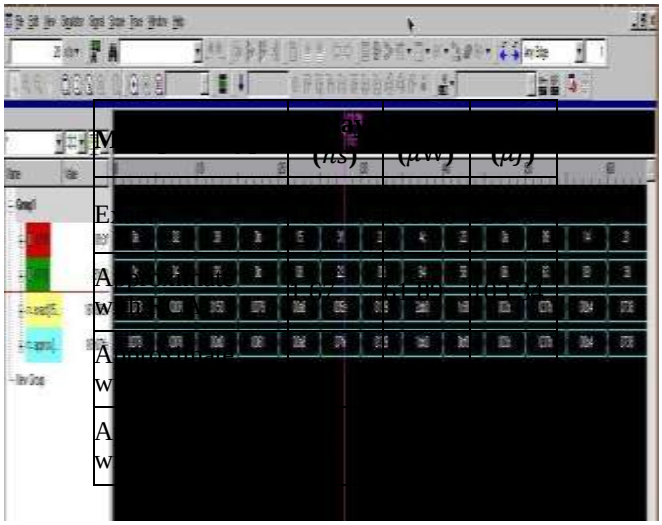

**Figure 4 Simulated waveform of exact and approximate multiplier**

| Multiplier Type | Area ($\mu m^2$) | Delay ($ns$) | Power ($\mu$W) |
|---|---|---|---|
| Exact | 780.75 | 2.19 | 167.09 |
| Approximate with RCA | 540.24 | 2.08 | 81.28 |
| Approximate with CSA | 703 | 1.37 | 114.55 |
| Approximate with CSA-BEC | 545.66 | 1.21 | 91.40 |

**Table 1 Synthesis Result**

Approximate multiplier with RCA, CSA and CSA-BEC shown 51.35%, 31.44% and 45.3% power reduction compared to exact conventional structure. Also these three circuits shown 5%, 37.44%, 44.3% speed improvement whereas 30.8%, 9.96%, 30.11% area savings respectively. High area and power consumed in CSA architecture is reduced by modified CSA with BEC.

The multiplier design is divided into two modules, one contains partial product alteration and accumulation and other contains final stage adder. From timing and power analysis we see that the module contains final stage adder having high delay contribution and the other module with high leakage contribution. So we designed the final stage adder with low Vt cells whereas other module with high Vt cells. The improved result with this approach is shown in Table 2.

**Table 2 Multi Vt Based Synthesis Result**

## V. CONCLUSION

In this paper, the proposed approximate multiplier uses propagate and generate signals which are modified from the partial products. Simple OR gates are used for generate partial product where approximate half adder, full adder and 4:2 compressor are used for other partial products accumulation. Three architecture of adder is evaluated where CSA-BEC structure stands best. The use of CSA-BEC, our design results into 70% and 67.8% savings in terms of PDP and APP in comparison with exact design. Further multi Vt design approach will results into 39.7% reduction of PDP. Error metrics with low values will conform to better precision result of our design. This design can be implemented for different applications where significant performance improvement can be achieved with minimum loss of output quality.

## VI. REFERENCES

1. Khaing Yin Kyaw, Wanh Ling Goh and Kiat Seng Yeo, 2010. Low power high speed multiplier for error tolerant systems. IEEE Intl. Conf. Electron Devices and Solid-State Circuits (EDSSC): 1-4.
2. P. Kulkarni, P. Gupta, and M. D. Ercegovac, 2011. Trading Accuracy for Power with an Under designed Multiplier Architecture. IEEE 24th Intl. Conf. on VLSI Design:346-351.
3. Chia-Hao Lin and Ing-Chao Lin, 2013. High accuracy approximate multiplier with error correction. 31st Proc. IEEE Int. Conf. Computer Design (ICCD): 33– 38.

4. V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, 2013. Low-Power Digital Signal Processing Using Approximate Adders. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 32(1): 124-137. C. Liu, J. Han, and F. Lombardi, [5] 2014. A Low- Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery. Design, Automation and Test in Europe Conference and Exhibition (DATE): 1-4.
5. Z. Yang, J. Han, and F. Lombard, 2015. Approximate Compressors for Error-Resilient Multiplier Design. IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS): 183-186.
6. A. Momeni, J. Han, P. Montuschi, and F. Lombardi, 2015. Design and Analysis of Approximate Compressors for Multiplication. IEEE Transactions on Computers, 64(4): 984-994.
7. S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim , 2015. Energy-Efficient Approximate Multiplication for Digital Signal Processing and Classification Applications. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 23(6): 1180-1184.
8. David May and Walter Stechele, 2016. Voltage Over-Scaling in Sequential Circuits for Approximate Computing. 11th Design and Technology of Integrated Systems in Nanoscale Era (DTIS): 1-6.
9. G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, 2016. Design-Efficient Approximate Multiplication Circuits Through Partial Product Perforation. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 24(10): 3105-3117.
10. J. Liang, J. Han, and F. Lombard, 2013. New Metrics for the Reliability of Approximate and Probabilistic Adders. IEEE Transactions on Computers, 62(9): 1760-1771.
11. M. Srivastav, S.S.S.P. Rao and H. Bhatnagar, 2005. Power reduction technique using multi-Vt libraries. Fifth International Workshop on System-on-Chip for Real-Time Applications (IWSOC): 363-367.
12. Deepak Kumar Patel, Raksha Chouksey and Minal Saxena, 2015. An Efficient VLSI Architecture for Carry Select Adder Without Multiplexer. International Journal of Computer Applications, 127(9): 37-40.
13. Leila kabbai, Anissa Sghaier, Ali Douik and Mohsen Machhout, 2016. FPGA implementation of filtered image using 2D Gaussian filter. International Journal of Advanced Computer Science and Applications (IJACSA), 7(7): 514-520.