ISSN: 2278-4632
Vol-10 Issue-7 No. 13 July 2020

Low-Power High-Accuracy Approximate Multiplier Using Approximate High-Order Compressors

Modalavalasa Rudrani[1], Mohankrishna Potnuru[2], Gadu Rambabu[3]

[1] PG Scholar, Dept of ECE, Sri Venkateswara College of Engineering and Technology, Srikakulam
[2-3] Assistant Professor, Dept of ECE, Sri Venkateswara College of Engineering and Technology, Srikakulam

[1]modalavalasarudrani@gmail.com
[2]mohan.424@gmail.com
[3] rambabu.g040@gmail.com

Abstract – To reduce the power consumption, the design of approximate multiplier appears as a promising solution for many error-resilient applications. In this paper, we propose a low-power high-accuracy approximate 8 x 8 multiplier design. The proposed design has two main features. First, according to the significance, different weights utilize different compressors (in different levels of accuracy) to accumulate their product terms. As a result, the power consumption can be saved with a small error. Second, for the middle significance weights, we use high-order approximate compressors (e.g., 8:2 compressor) to reduce the logic of carry chains. To our knowledge, the proposed design is the first work that successfully uses high order approximate compressors in the approximate multiplier design. Compared with an exact multiplier (DADDA tree multiplier), experimental results show that the proposed approximate multiplier can achieve both low power and high accuracy.

Keywords: Approximate Computing, Arithmetic Circuits, Logic Design, Low-Power Design, Partial Product Reduction

1. INTRODUCTION

Multipliers play an important role in today's digital signal processing and various other applications. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following design targets – high speed, low power consumption, regularity of layout and hence less area or even combination of them in one multiplier thus making them suitable for various high speed, low power and compact VLSI implementation.

The common multiplication method is "add and shift" algorithm. In parallel multipliers number of partial products to be added is the main parameter that determines the performance of the multiplier. In applications like multimedia signal processing and data mining which can tolerate error, exact computing units are not always necessary. They can be replaced with their approximate counterparts. Research on approximate computing for error tolerant applications is on the rise. Adders and multipliers form the key components in these applications. In, approximate full adders are proposed at transistor level and they are utilized in digital signal processing applications. Their proposed full adders are used in accumulation of partial products in multipliers. To reduce hardware complexity of multipliers, truncation is widely employed in fixed-width multiplier designs. Then a constant or variable correction term is added to compensate for the quantization error introduced by the truncated part. Approximation techniques in multipliers focus on accumulation of partial products, which is crucial in terms of power consumption. Broken array multiplier is implemented in, where the least significant bits of inputs are truncated, while forming partial products to reduce hardware complexity. The proposed multiplier in saves few adder circuits in partial product accumulation. In, two designs of approximate 4-2 compressors are presented and used in partial product reduction tree of four variants of 8×8 DADDA multiplier.

The major drawback of the proposed compressors in is that they give nonzero output for zero valued inputs, which largely affects the mean relative error (MRE) as discussed later. The approximate design proposed in this brief overcomes the existing drawback. This leads to better precision. In static segment multiplier (SSM) proposed in, m-bit segments are derived from n-bit operands based on leading 1 bit of the operands. Then, $m \times m$ multiplication is performed instead of $n \times n$ multiplication, where $m<n$. Partial product perforation (PPP) multiplier in omits k successive partial products starting from j^{th} position, where $j \in [0, n-1]$ and $k \in [1, \min(n-j, n-1)]$ of a n-bit multiplier. In 2×2 approximate multiplier based on modifying an entry in the Karnaugh map is proposed and used as a building block to construct 4×4 and 8×8 multipliers. In inaccurate counter design has been proposed for use in power efficient Wallace tree multiplier.

A new approximate adder is presented in which is utilized for partial product accumulation of the multiplier. For 16-bit approximate multiplier in, 26% of reduction in power is accomplished compared to exact multiplier. Approximation of 8-bit Wallace tree multiplier due to voltage over-scaling (VOS) is discussed in. Lowering supply voltage creates paths failing to meet delay constraints leading to error. Previous works on logic complexity reduction focus on straight forward application of approximate adders and compressors to the partial products. In this brief, the partial products are altered to introduce terms with different probabilities. Probability statistics of the altered partial products are analyzed, which is followed by systematic approximation. Simplified arithmetic units (half-adder, full-adder, and 4-2 compressor) are proposed for approximation. The arithmetic units are not only reduced in complexity, but care is also taken that error value is maintained low. While systemic approximation helps in achieving better accuracy, reduced logic complexity of approximate arithmetic units consumes less power and area.

2. LITERATURE SURVEY

Gao et al., presents an optimized design approach of two's complement large-size multipliers using embedded multipliers in FPGAs. The realization is based on Baugh-Wooley's algorithm. To achieve efficient implementation, a set of optimized schemes for the realization of the addition of partial products is proposed. The implementations of the multipliers is carried out for operands with sizes from 20 to 128 bits. The results indicate that the proposed approach outperforms the traditional methods by 50% in terms of LUT-delay product. In this chapter, a research overview of different technique used forthe power reduction in Modified Booth's Multiplier for the past one decade is presented. Design techniques have clearly focused on power reduction in recent years, although the larger goal is to achieve power efficiency and performance without compromising the system functionality, which is much more difficult. In subsequent chapters, describes analysis and design of approximate computing for low power, using multiplier designs are down to reinforce the conclusions.

3. PROPOSED HIGH ORDER COMPRESSORS

The critical path of a multiplier is often related to the maximum height of PPM (partial product matrix). Thus, there is a need to compress the PPM. A n:2 compressor is a slice of a multiplier that reduces n numbers (i.e., product terms) to two numbers when properly replicated. In slice *i* of the multiplier, the n:2 compressor receives n bits in position i and one or more carry bits from the lower positions (such as *i-1*), and produces two output bits in positions *i* and *i+1* and one or more carry bits into the higher positions.

Conventionally, 4:2 compressors are used in the multiplier design. Fig. 1 (a) gives the block diagram of an accurate (i.e., exact) 4:2 compressor. The four input bits are denoted as X0, X1, X2 and X3. The two output bits in positions i and i+1 are denoted to as Sum and Carry respectively. The carry bit from the lower position is denoted as Cin while the carry bit into the higher position is denoted as Cout. Fig. 1 (b) gives the block diagram of an approximate 4:2 compressor. To save the logic of carry chains, the carry bits Cin and Cout are omitted. Moreover, in [1,2], to reduce the error rate, the logics of Sum and Carry are re-designed (i.e., different from the logics of Sum and Carry in an accurate 4:2 compressor).

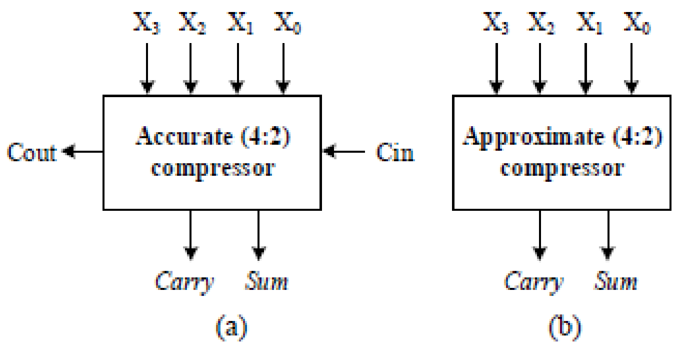

Figure 1. (a) Accurate 4:2 compressor (b) Approximate 4:2 compressor

3.1. The Approximation of Carry

Here, we study the approximation of the logic of the Carry output. In a conventional half adder, the carry bit Ch is defined as below:

Ch (X0,X1) = X0 . X1 (1)

In a conventional full adder, the carry bit Cf is defined as below:

Cf (X0,X1,X2) = X0. X1 + X1. X2 + X0. X2 (2)

As described in [6], we can implement the equation (1) as a modified half adder, and implement the equation (2) as a modified full adder. Fig. 2 (a) and Fig. 2 (b) give the logic of modified half adder and the logic of modified full adder, respectively. Then, based on the modified half adder and the modified full adder, we can construct the approximation logic for the Carry output of a high-order approximate compressor. In the following, we use the Carry output of our approximate 5:2 compressor examples. When the number of input bits is 5 (i.e., n = 5), we can split the 5 input bits into 2 groups: one group includes X0, X1, and X2, and the other group includes X3 and X4. Then, the Carry output of our approximate 5:2 compressor is as below:

Cf (X0,X1,X2) +Ch (X3,X4) + Ch (X0+X1+X2,X3+X4). Fig. 3 displays the logic of the Carry output of our approximate 5:2 compressor. When the number of input bits is 8 (i.e., n = 8), we can split the 8 input bits into 3 groups: one group includes X0, X1, and X2, one group includes X3, X4, and X5, and one group includes X6 and X7. Then, the Carry output of our approximate 8:2 compressor is as below: Cf (X0,X1,X2) + Cf (X3,X4,X5) + Ch(X6,X7) + Cf(X0+X1+X2, X3+X4+X5, X6+X7).

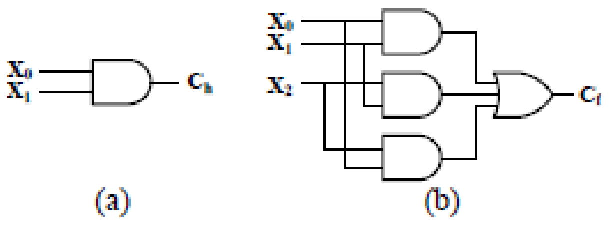

Figure 2. (a) Modified half adder (b) Modified full adder Modified

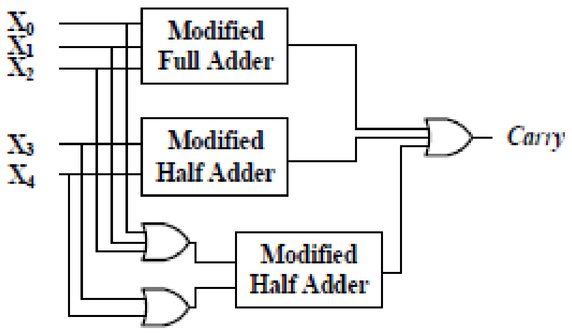

Figure 3. The logic of Carry output of our approximate 5:2 compressor.

3.2. The Approximation of Sum

Here, we study the approximation of the logic of *Sum* output. Conventionally, the tree of XOR gates are used to produce the output Sum. However, compared with other logic gates, XOR gate often has larger design overheads. We use the logic gates in SAED 32nm cell library as an example. Table I tabulates the comparisons among OR gate, NOR gate, XNOR gate, and XOR gate. From Table I, we find that XOR gate has the largest power, the largest area, and the largest delay. Thus, if we can replace XOR gates with other logic gates, all the design overheads (including the power, the area, and the delay) can be reduced.

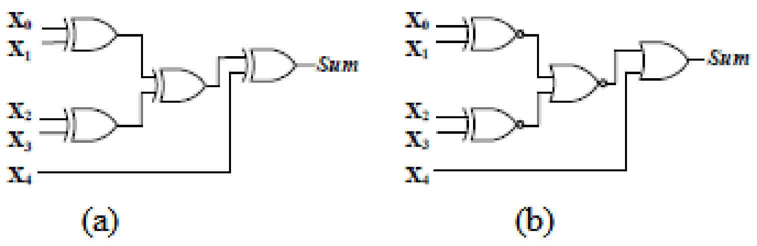

Figure 4. Sum of 5:2 compressor (a) Accurate (b) Our approximate

We construct the approximation logic (a tree of logic gates) for the Sum output of a high-order approximate compressor as below. At the first level, we use XNOR gate instead of XOR gate. Note that the output of XNOR gate is the inverse of the output of XOR gate. To compensate for the error rate, we use NOR gate at the second level and OR gate at the third level. Since all XOR gates are replaced by other logic gates, the design overheads are greatly saved.

In the following, we use the Sum output of 5:2 compressor (Fig. 4) as examples. Fig 4 (a) gives the logic of the Sum output of an accurate 5:2 compressor. Fig 4 (b) gives the logic of the *Sum* output of our approximate 5:2 compressor.

4. PROPOSED APPROXIMATE MULTIPLIER DESIGN

Typically, a multiplier consists of three parts. In the first part, AND gates are utilized to generate partial products. In the second part, the maximum height of PPM (partial product matrix) is reduced by using a carry save adder tree. In the third part, a carry propagation adder is used to produce the final result. The design complexity of a multiplier is primarily related to the PPM reduction circuitry (i.e., the multiplier is primarily related to the PPM reduction circuitry (i.e., the second part). Thus, the study of multiplier design [1-6] focuses on the optimization of the PPM reduction circuitry. In this section, we propose an approximate 8 x 8 multiplier design. Fig. 5 gives the overall structure of our PPM reduction circuitry. According to the significance, the weights are classified into three categories: the higher significance weights, the middle significance weights, and the lower significance weights. Note that the designers are allowed to configure the number of higher significance weights, the number of middle significance weights and the number of lower significance weights for the trade-off between the power consumption and the computational accuracy.

To reduce the power consumption with a small error, our PPM reduction circuitry applies the significance driven logic compression technique as below: the higher significance weights use accurate (i.e., exact) 4:2 compressors; the middle significance weights use our approximate high-order compressors; the lower significance weights use inaccurate compressors (OR-tree based approximation). Our PPM reduction circuitry has two stages. The first stage is for all the weights. The second stage is only for the higher significance weights. After the second stage is completed, each weight has at most two product terms. Thus, a carry propagation adder can be used to produce the final result. In the following, we elaborate the details of these two stages.

4.1. The First Stage

For each lower significance weight, we use a simple OR tree based approximation for power saving. Suppose that the number of inputs is n. If $n \leq 2$, no action is performed. On the other hand, if $n > 2$, we use an OR tree for n-1 inputs to approximate the accumulation result of these n-1 inputs. Thus, after the first stage is done, each lower significance weight has at most two product terms. For each middle significance weight, we use our approximate n:2 compressor for power saving, where n is the number of product terms in this weight. As described in Section II, the designers can choose one of the following two implementations: one implementation is with accurate Sum and approximate Carry and the other implementation is with approximate Sum and approximate Carry. After the first stage is done, each middle significance weight has at most two product terms. To achieve high accuracy, for each higher significance weight, we use accurate (i.e., exact) 4:2 compressors. For each accurate 4:2 compressor, if the number of product terms is less than 4, the values of other inputs to this compressor are set to be 0. In the rightmost higher significance weight, the carry bit Cin of one accurate 4:2 compressor is from the Carry output of the leftmost middle significance weight, and the carry bit Cin of the other one accurate 4:2 compressor is set to be 0.

4.2. The Second Stage

Note that the second stage is only for the higher significance weights. In order to achieve high accuracy, we use accurate (i.e., exact) 4:2 compressors to reduce the maximum height of the PPM. The carry bit Cin of the rightmost accurate 4:2 compressor is set to be 0. As shown in Fig. 5, after the second stage is completed, each higher significance weight has two product terms.

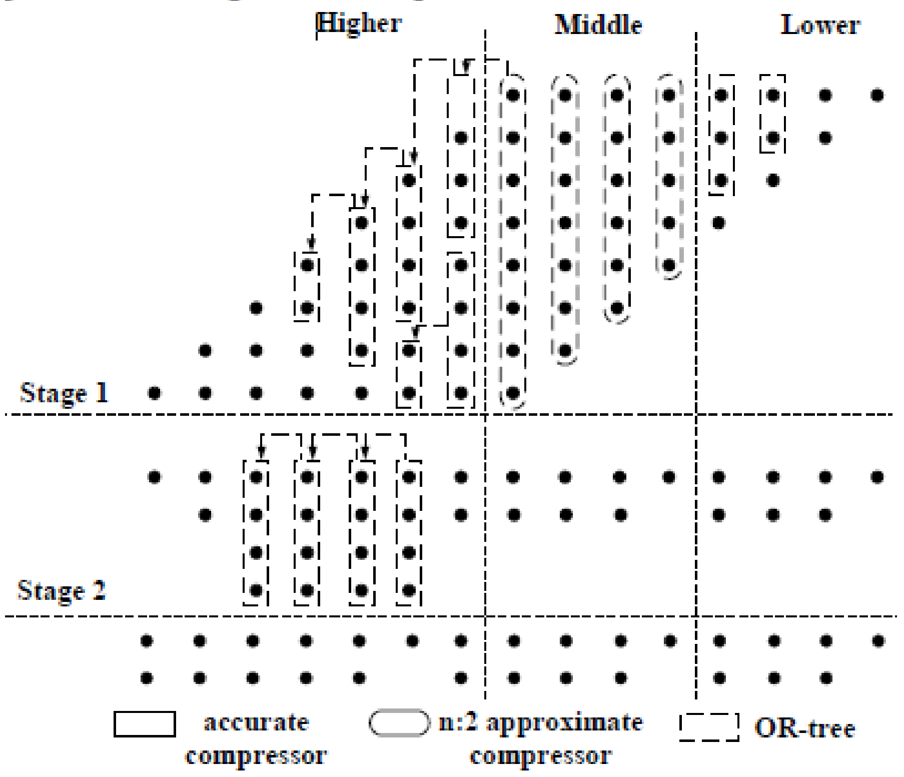

Figure 5: The PPM reduction in the proposed approximate multiplier

5. RESULTS

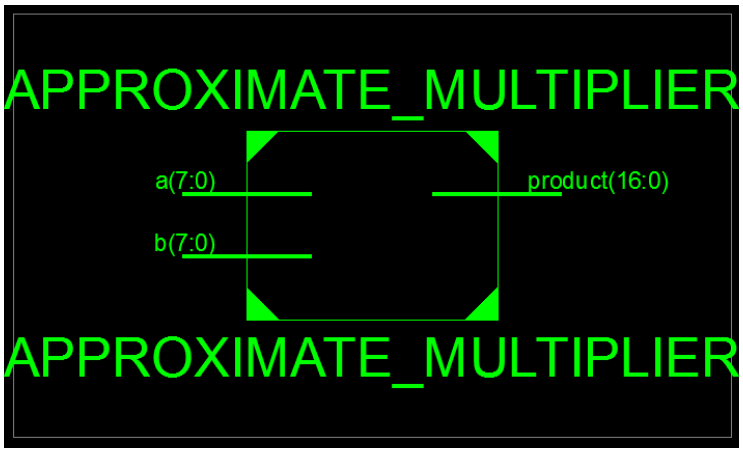

Figure 6: RTL Schematic of approximate multiplier

ISSN: 2278-4632
Vol-10 Issue-7 No. 13 July 2020

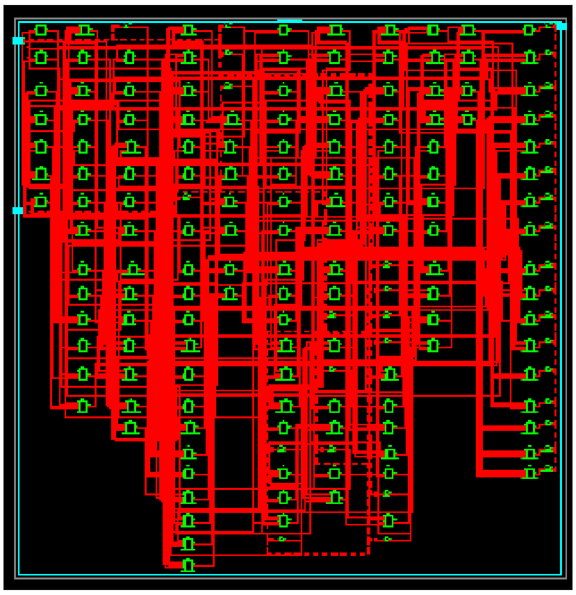

Figure 7: View Technology Schematic of approximate multiplier

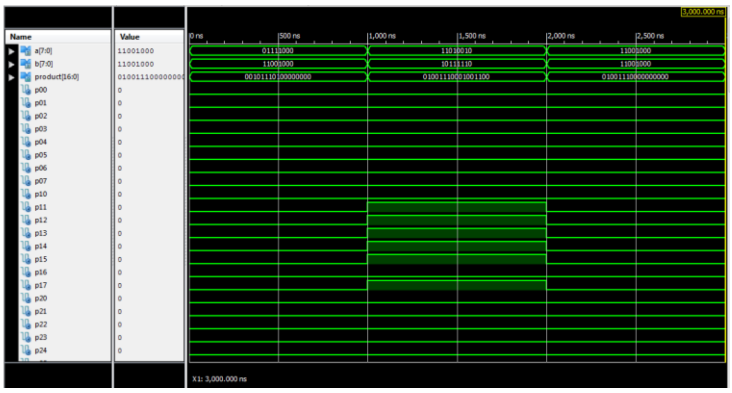

Figure 8: Simulated Waveforms of approximate multiplier

Parameter	DADDA multiplier	Approximate multiplier
No of LUTs	146	142
Delay (ns)	24.257	22.186
Power (m Watt)	1.191	1.158

Table 1: Parameter comparison table

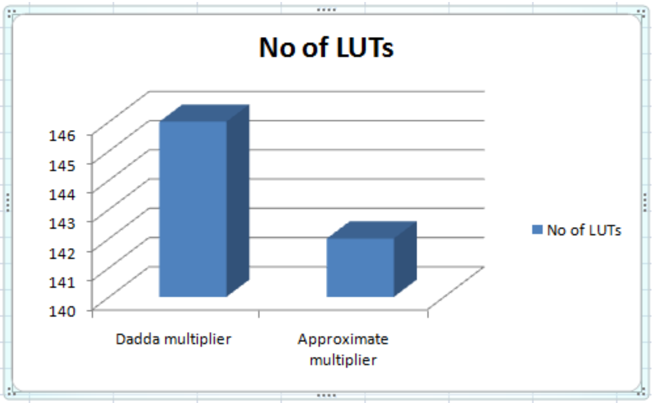

Figure 9: LUT comparison bar graph

ISSN: 2278-4632
Vol-10 Issue-7 No. 13 July 2020

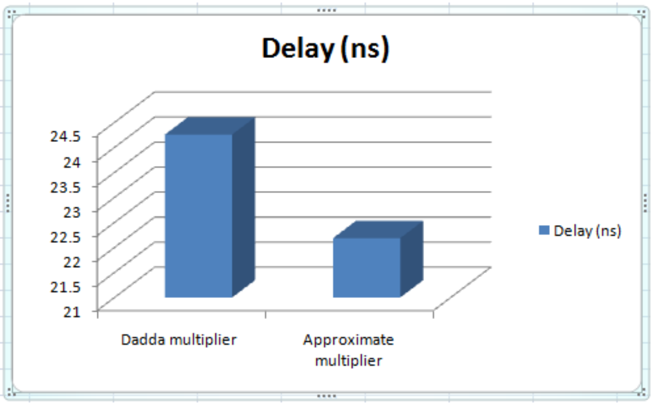

Figure 10: Delay comparison bar graph

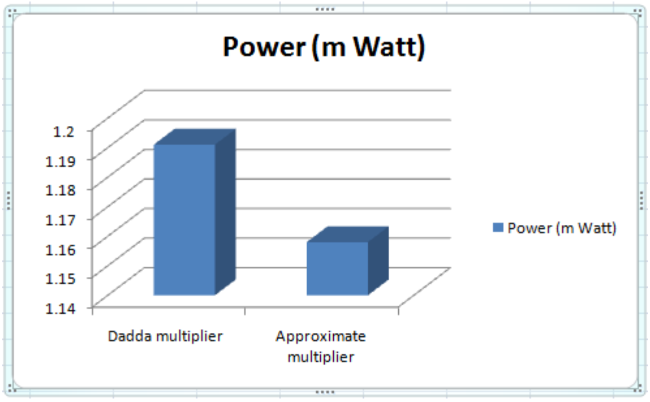

Figure 11: Power comparison bar graph

6. CONCLUSION

This project presents a low-power high-accuracy approximate 8 x 8 multiplier design. To achieve high accuracy, we use accurate (i.e., exact) 4:2 compressors in the higher significance weights. To reduce power consumption, we use high-order approximate compressors in the middle significance weights. The experimental results show that the proposed approximate multiplier design can save area power consumption and high speed compared to normal DADDA multiplier. To our knowledge, the proposed design is the first work that successfully utilizes high-order approximate compressors in the approximate multiplier design for achieving low power dissipation while still maintaining high accuracy.

7. REFERENCES

1. Z. Yang, J. Han and F. Lombardi, “Approximate Compressor for Error-Resilient Multiplier Design”, Proceedings of IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, 2015.
2. A. Momeni, J. Han, P. Montuschi and F. Lombardi, “Design and Analysis of Approximate Compressors for Multiplication”, IEEE Trans. on Computers, vol. 64, no. 4, pp. 984-994, 2015.

3. C. Liu, J. Han and F. Lombardi, "A Low-Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery", Proceedings of IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014.

4. G. Zervakis, et al., "Design-Efficient Approximate Multiplication Circuits Through Partial Product Perforation", IEEE Transactions on Very Large Scale Integration Systems, vol.24, no.10, pp. 3105-3117, 2016.

5. T. Yang, T. Ukezono and T. Sato, "A Low-Power High-Speed Accuracy-Controllable Approximate Multiplier Design", Proceedings of IEEE Asia and South Pacific Design Automation Conference (ASPDAC), 2018.

6. A. Cilardo, et al., "High-Speed Speculative Multipliers Based on Speculative Carry-Save Tree", IEEE Transaction on Circuits and Systems - I, vol. 61, no. 12, pp. 3426–3435, 2014.

7. J. Liang, et al., "New Metrics for The Reliability of Approximate and Probabilistic Adders", IEEE Trans. on Computers, vol. 62, no. 9, pp. 1760-1771, 2013.

8. P. Kulkarni, P. Gupta and M. D. Ercegovac, "Trading accuracy for power in a multiplier architecture", J. Low Power Electron., vol. 7, no. 4, pp. 490–501, 2011.

9. C.H. Lin and C. Lin, "High accuracy approximate multiplier with error correction," in Proceedings IEEE International Conference on Computational Design, pp. 33–38, Sep. 2013.

10. C. Liu, J. Han and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in Proc. Conf. Exhibit, pp. 1–4, 2014.