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

Che-Wei Tung and Shih-Hsu Huang
Department of Electronic Engineering
Chung Yuan Christian University
Taoyuan, Taiwan
e-mail:{g10776037,shhuang}@cycu.edu.tw

***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

I. INTRODUCTION

Multiplication is a crucial fundamental arithmetic operation in digital signal processing. To reduce the power consumption of an embedded system, the design of approximate multiplier appears as a promising solution for many error-resilient applications. In [1,2], different approximate multiplier designs (based on approximate 4:2 compressors) have been proposed to save the power consumption. However, their NMED (normalized mean error distance) values are relatively large.

Liu et al. [3] limited the carry propagation to the nearest neighbors for fast partial product accumulation. Zervakis et al. [4] used partial product perforation to reduce the power consumption. Yang et al. [5] used carry-maskable adders to reduce the length of carry chain. These approaches [3-5] rely on extra logic for post-processing (i.e., error recovery) to reduce the error distance.

In this paper, we propose a low-power high-accuracy approximate 8 x 8 multiplier design. The main contribution of our work is that we successfully demonstrate that high-order approximate compressors (e.g., 8:2 compressor) can be used in the approximate 8 x 8 multiplier design for achieving lower power dissipation while still maintaining high accuracy. Note that the proposed design does not require any extra logic for post-processing (i.e., error recovery).

The architecture of the proposed approximate 8 x 8 multiplier design has the following two main features.

(1) *Significance Driven Logic Compression*. According to the significance, different weights use different compressors (i.e., counters) to accumulate their product terms. The higher significance weights use accurate 4:2 compressors, the middle significance weights use near-accurate compressors, and the lower significance weights use inaccurate compressors. As a result, the power consumption can be reduced with a small error.

(2) *High Order Approximate Compression*. For the middle significance weights, we use high-order approximate compressors (e.g., 8:2 compressor) to reduce the logic of carry chains. As a result, both the delay and the power can be greatly saved. To the best of our knowledge, the proposed design is the first work that utilizes high-order approximate compressors in the approximate multiplier design.

Note that this architecture allows the designers 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 dissipation and the computational accuracy. Compared with an exact multiplier (Dadda tree multiplier), experimental results show that the proposed approximate multiplier can achieve 14.62% ~ 25.92% reduction in power consumption with only 0.07% ~ 0.89% NMED. Therefore, the proposed approximate multiplier does achieve both low power and high accuracy.

II. 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 [1,2]. Fig. 1 (a) gives the block diagram of an accurate (i.e., exact) 4:2 compressor. The four input bits are denoted as X_0, X_1, X_2 and X_3. 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

978-1-7281-1439-2/19/$31.00 ©2019 IEEE

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).

Previous works [1,2] did not consider high-order compression (i.e., did not consider n ≥ 5). In fact, high-order compression can further reduce the delay and power. In this section, we introduce our high-order compressor design (i.e., n ≥ 5). Section II-A presents the approximation of *Carry*. Section II-B presents the approximation of *Sum*. Section II-C presents the implementation results.

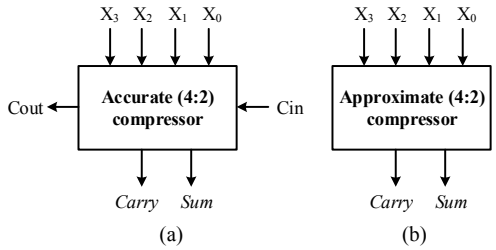

Figure 1. (a) Accurate 4:2 compressor (b) Approximate 4:2 compressor.

A. The Approximation of Carry

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

$$C_h (X_0,X_1) = X_0 \cdot X_1 \quad (1)$$

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

$$C_f (X_0,X_1,X_2) = X_0 \cdot X_1 + X_1 \cdot X_2 + X_0 \cdot X_2 \quad (2)$$

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 X_0, X_1, and X_2, and the other group includes X_3 and X_4. Then, the *Carry* output of our approximate 5:2 compressor is as below: $C_f (X_0,X_1,X_2) + C_h (X_3,X_4) + C_h (X_0+X_1+X_2,X_3+X_4)$. 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 X_0, X_1, and X_2, one group includes X_3, X_4, and X_5, and one group includes X_6 and X_7. Then, the *Carry* output of our approximate 8:2 compressor is as below: $C_f (X_0,X_1,X_2) + C_f (X_3,X_4,X_5) + C_h(X_6,X_7) + C_f(X_0+X_1+X_2, X_3+X_4+X_5, X_6+X_7)$.

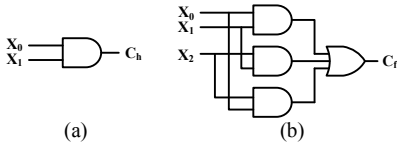

Figure 2. (a) Modified half adder (b) Modified full adder.

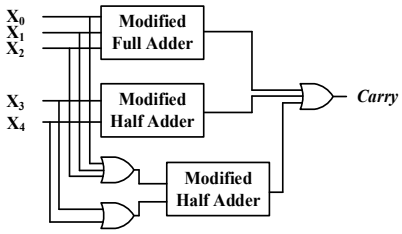

Figure 3. The logic of *Carry* output of our approximate 5:2 compressor.

B. 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.

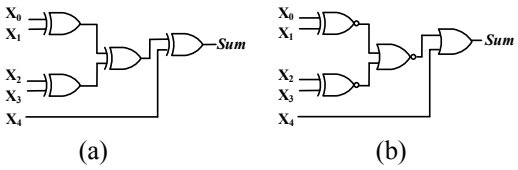

Figure 4. *Sum* of 5:2 compressor. (a) Accurate (b) Our approximate

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.

TABLE I. COMPARISONS ON DIFFERENT LOGIC GATES

Logic gate	Power (μW)	Area (μm^2)	Delay (ns)
OR	0.1693	2.28	0.05
NOR	0.2040	2.79	0.07
XNOR	0.3885	4.57	0.11

XOR	0.4232	4.57	0.12

TABLE II. COMPARISONS ON DIFFERENT HIGH-ORDER COMPRESSORS

n	Accurate n:2 Compressor			Accurate *Sum* & Approximate *Carry*			Approximate *Sum* & Approximate *Carry*		
	Power (μW)	Area (μm^2)	Delay (ns)	Power (μW)	Area (μm^2)	Delay (ns)	Power (μW)	Area (μm^2)	Delay (ns)
5.00	61.67	123.34	0.94	37.72	82.65	0.58	36.41	76.47	0.43
6.00	78.26	177.04	0.94	43.02	93.43	0.58	41.46	87.25	0.57
7.00	98.58	218.63	0.95	49.43	113.29	0.58	46.49	98.09	0.50
8.00	119.27	291.41	1.07	55.22	127.58	0.58	51.50	107.86	0.50

C. *Implementation Results*

We have used SAED 32nm cell library to realize three different implementations of n:2 compressor for comparisons. For each n:2 compressor, one implementation is an accurate (i.e., exact) compressor, one implementation is an approximate compressor based on our approximation on *Carry* (i.e., accurate *Sum* and approximate *Carry*), and one implementation is an approximate compressor based on our approximation on both *Sum* and *Carry* (i.e., approximate *Sum* and approximate *Carry*).

Table II tabulates the comparisons on these three implementations of different n:2 compressors, where n = 5, 6, 7, and 8. Note that the clock rate is assumed to be 1 GHz. With an analysis to Table II, we have the following observations.

(1) Among these three implementations, the implementation with both approximate *Sum* and approximate *Carry* has the smallest design overheads (including the power, the area, and the delay).

(2) Compared with the accurate implementation, the approximation of *Carry* can reduce 38.84% ~ 53.70% power consumption.

(3) Compared with the accurate implementation, the approximation of both *Sum* and *Carry* can reduce 40.96% ~ 56.82% power consumption.

III. 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 (i.e., the approximate n:2 compressors proposed in Section II); 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.

A. *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 (as described in Section II) 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.

B. 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.

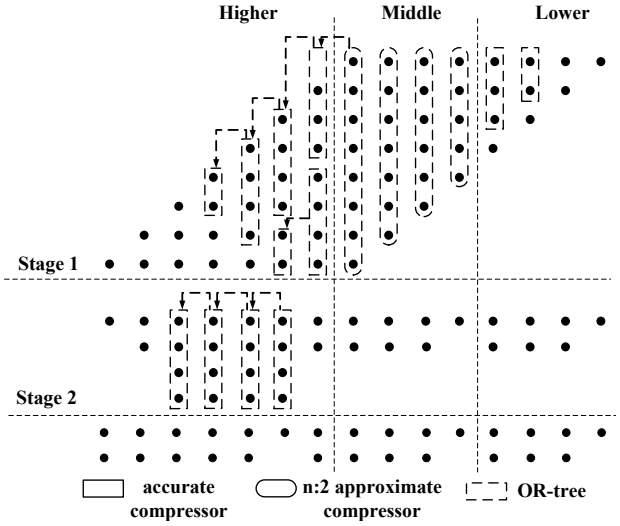

Figure 5. The PPM reduction in the proposed approximate multiplier.

IV. Experimental Results

In this section, we present our experimental results. We have used SAED 32nm cell library to implement the proposed approximate 8 x 8 multiplier. Moreover, we also implemented convention Dadda tree 8 x 8 multiplier and two approximate 8 x 8 multipliers (called AM_1 and AM_2) proposed in [2] for comparisons. In our experiments, the clock rate is assumed to be 1 GHz.

Note that the proposed approximate 8 x 8 multiplier allows the designers 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. Moreover, the proposed high-order compressor (used in the compression of middle significance weights) also has two possible implementations: one implementation is with accurate *Sum* and approximate *Carry* and the other implementation is with approximate *Sum* and approximate *Carry*.

We have realized many different implementations (configurations) of the proposed approximate 8 x 8 multiplier. Here, we use the term *X*_AM*Y*,*Z* for the naming of each implementation. We explain the naming as below.

X: the implementation of high-order compressors. $X = 1$ means accurate *Sum* and approximate *Carry*; $X = 2$ means approximate *Sum* and approximate *Carry*.

Y: the rightmost column (i.e., position) of middle significance weights.

Z: the number of middle significance weights.

Using Fig. 5 as an example, we have $Y = 4$ (i.e., the rightmost column of middle significance weights is the 4th column) and $Z = 4$ (i.e., the number of middle significance weights is 4). Note that the rightmost column of the PPM is defined as the 0th column.

In the experiments, we have realized 10 different implementations of the proposed approximate 8 x 8 multiplier for comparisons. For example, the implementation 1_AM5,3 means accurate *Sum* and approximate *Carry* (for high-order compressors), the rightmost column of middle significance weights is 5, and the number of middle significance weights is 3.

A. Circuit Analysis

Table III tabulates the comparisons of power consumption and circuit area among different multiplier designs. From Table III, we find that, compared with the Dadda tree multiplier (i.e., an exact multiplier), the proposed approximate multiplier can greatly reduce both the power consumption and circuit area. With further analysis, we find that, compared with the Dadda tree multiplier, the proposed approximate multiplier can achieve 14.62% ~ 25.92% reduction in power dissipation.

Table IV tabulates the comparisons of critical path delays among different multiplier designs. From Table IV, we find that, compared with the Dadda tree multiplier and the approximate multiplier AM_2 proposed in [2], the proposed approximate multiplier can reduce the critical path delay. On the other hand, compared with the approximate multiplier AM_1 proposed in [2], the critical path delay of each implementation of the proposed approximate multiplier is slightly larger.

B. Accuracy Analysis

Here we adopt the accuracy analysis metrics proposed in [7]. The ED (error distance) is defined as the difference between the accurate product (P) and the approximate product (P'), i.e., ED=|P'-P|. For each multiplier design, we perform an exhaustive simulation. Then, the ED of each test pattern can be calculated. The MED (mean error distance) is the average ED for a set of outputs. The normalized MED (NMED) is defined as MED/P_{max}, the P_{max} is the maximum magnitude of the output of the multiplier. The RED (relative error distance) is defined as RED = ED/P, and the MRED (mean relative error distance) can be obtained by averaging RED.

Table V tabulates the comparisons on NMED and MRED among different approximate multiplier designs. As shown in Table V, the proposed approximate multiplier achieves 0.07% ~ 0.89% NMED and 0.41% ~ 11.29% MRED. In other words, each implementation of the proposed approximate multiplier has low NMED and low MRED. The main reason is that we use accurate 4:2 compressors for higher significance weights. On the other hand, the two approximate multipliers (i.e., AM_1 and AM_2) proposed in [2] have very high NMED and very high MRED. Thus, compared with these two approximate multipliers (i.e., AM_1 and AM_2) proposed in [2], each implementation of

the proposed approximate multiplier has a very significant improvement in the computation accuracy.

V. CONCLUSIONS

This paper 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. Compared with the Dadda tree multiplier, experimental results show that the proposed approximate multiplier design can save 14.62% ~ 25.92% power consumption with only 0.07% ~ 0.89% NMED. 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.

TABLE III. COMPARISONS ON POWER AND AREA

Design	Power (μW)	Normalized Power (%)	Area (μm^2)	Normalized Area (%)
Dadda tree	338.95	100.00	1445.49	100.00
AM_1 [2]	251.13	74.09	1116.94	77.27
AM_2 [2]	251.4	74.17	1073.88	74.29
1_AM5,3	285.57	84.25	1241.89	85.91
1_AM5,4	276.54	81.59	1145.74	79.26
1_AM4,4	289.4	85.38	1238.11	85.65
1_AM4,5	281.91	83.17	1187.2	82.13
1_AM4,6	272.42	80.37	1141.72	78.98
2_AM5,3	271.39	80.07	1207.92	83.56
2_AM5,4	261.53	77.16	1127.08	77.97
2_AM4,4	275.13	81.17	1234.13	85.38
2_AM4,5	262.94	77.57	1151.18	79.64
2_AM4,6	251.09	74.08	1113.39	77.03

TABLE IV. COMPARISONS ON CRITICAL PATH DELAYS

Design	Delay (ns)	Normalized Delay (%)
Dadda tree	3.67	100.00
AM_1 [2]	3.20	87.19
AM_2 [2]	3.67	100.00
1_AM5,3	3.45	94.01
1_AM5,4	3.47	94.55
1_AM4,4	3.66	99.73
1_AM4,5	3.29	89.65
1_AM4,6	3.29	89.65
2_AM5,3	3.45	94.01
2_AM5,4	3.47	94.55
2_AM4,4	3.64	99.18
2_AM4,5	3.28	89.37
2_AM4,6	3.29	89.65

TABLE V. COMPARISONS ON NMED AND MRED

Design	NMED (%)	Normalized NMED (%)	MRED (%)	Normalized MRED (%)
AM_1 [2]	6.07	100.00	452.16	100.00
AM_2 [2]	5.41	89.13	424.48	93.88
1_AM5,3	0.07	1.15	0.41	0.09
1_AM5,4	0.38	6.26	2.74	0.61
1_AM4,4	0.21	3.46	1.98	0.44
1_AM4,5	0.34	5.60	1.89	0.42
1_AM4,6	0.45	7.41	1.80	0.40
2_AM5,3	0.18	2.97	0.28	0.06
2_AM5,4	0.55	9.06	8.47	1.87
2_AM4,4	0.28	4.61	5.24	1.16
2_AM4,5	0.52	8.57	7.91	1.75
2_AM4,6	0.89	14.66	11.29	2.50

ACKNOWLEDGMENTS

This work was supported in part by Ministry of Science and Technology, Taiwan, under grant number 107-2218-E-033-007. We also thank the Chung Yuan Christian University for financial support.

REFERENCES

[1] Z. Yang, J. Han, and F. Lombardi, "Approximate Compressor for Error-Resilient Multiplier Design", Proc. 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", Proc. 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 Trans. on Very Large Scale Integration (VLSI) 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", Proc. of IEEE Asia and South Pacific Design Automation Conference (ASP-DAC), 2018.

[6] A. Cilardo, et al., "High-Speed Speculative Multipliers Based on Speculative Carry-Save Tree", IEEE Trans. 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.