High-Accuracy Fixed-Width Modified Booth Multipliers for Lossy Applications

Jiun-Ping Wang, Shiann-Rong Kuang, *Member, IEEE*, and Shish-Chang Liang

***Abstract*—The fixed-width multiplier is attractive to many multimedia and digital signal processing systems which are desirable to maintain a fixed format and allow a little accuracy loss to output data. This paper presents the design of high-accuracy fixed-width modified Booth multipliers. To reduce the truncation error, we first slightly modify the partial product matrix of Booth multiplication and then derive an effective error compensation function that makes the error distribution be more symmetric to and centralized in the error equal to zero, leading the fixed-width modified Booth multiplier to very small mean and mean-square errors. In addition, a simple compensation circuit mainly composed of the simplified sorting network is also proposed. Compared to the previous circuits, the proposed error compensation circuit can achieve a tiny mean error and a significant reduction in mean-square error (e.g., at least 12.3% reduction for the 16-bit fixed-width multiplier) while maintaining the approximate hardware overhead. Furthermore, experimental results on two real-life applications also demonstrate that the proposed fixed-width multipliers can improve the average peak signal-to-noise ratio of output images by at least 2.0 dB and 1.1 dB, respectively.**

***Index Terms*—Error compensation circuit, fixed-width multiplier, mean error, mean-square error, modified Booth multiplier.**

I. Introduction

High processing performance and low power dissipation are the most important objectives in many multimedia and digital signal processing (DSP) systems, where multipliers are always the fundamental arithmetic unit and significantly influence the system's performance and power dissipation. To achieve high performance, the modified Booth encoding [1]–[4] which reduces the number of partial products by a factor of two through performing the multiplier recoding has been widely adopted in parallel multipliers. Moreover, $n \times n$ fixed-width multipliers that generate only the n most significant product bits are frequently utilized to maintain a fixed word size in these lossy systems which allow a little accuracy loss to output data. Significant hardware complexity reduction and power saving can be achieved by directly removing the adder cells of standard multiplier for the computation of the n least significant bits of $2n$-bit output product. However, a huge truncation error will be introduced to this kind of direct-truncated fixed-width multiplier (DTFM).

To effectively reduce the truncation error, various error compensation methods, which add estimated compensation value to the carry inputs of the reserved adder cells, have been proposed. The error compensation value can be produced by the constant scheme or the adaptive scheme. The constant scheme [5]–[7] pre-computes the constant error compensation value and then feeds them to the carry inputs of the retained adder cells when performing multiplication operations regardless of the influence of the current input data value. With the advantage of simplification, the truncation error of the constant scheme is relatively large. On the contrary, the adaptive scheme [8]–[13] was developed to achieve higher accuracy than the constant scheme through adaptively adjusting the compensation value according to the input data at the expense of a little higher hardware complexity. However, most of the adaptive error compensation approaches are developed only for fixed-width array multipliers and cannot be applied to significantly reduce the truncation error of fixed-width modified Booth multipliers directly.

To overcome this problem, several error compensation approaches [14]–[16] have been proposed to effectively reduce the truncation error of fixed-width modified Booth multipliers. In [14], the compensation value was generated by using statistical analysis and linear regression analysis. This approach can significantly decrease the mean error of fixed-width modified Booth multipliers, but the maximum absolute error and the mean-square error are still large. Cho *et al.* [15] divided the truncated part of the bit product matrix of Booth multiplication into a major group and a minor group depending on their effects on the truncation error. To obtain better error performance with a simple error compensation circuit, Booth encoded outputs are utilized to generate the error compensation value. In [16], a systematic design methodology for the low-error fixed-width modified Booth multiplier via exploring the influence of various indices in a binary threshold was developed to decrease the product error. The fixed-width modified Booth multipliers in [15] and [16] achieve better error performance in terms of the maximum absolute error and the mean-square error when compared with the previous published multiplier in [14]. However, their mean errors are much larger than that of [14].

The smaller mean error and mean-square error represent that the error distribution is more symmetric to and centralized in the error equal to zero (denoted as *zero error*). For many multimedia and DSP applications, the final output data are produced from accumulating a series of products rather than from a single multiplication operation directly. In such case, the truncation errors

Manuscript received April 03, 2009; revised July 20, 2009. First published November 10, 2009; current version published December 27, 2010. This work was supported in part by the National Science Council, Taiwan, under Grant NSC 97-2220-E-110-006.

The authors are with the Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan (e-mail: d933040009@student.nsysu.edu.tw; srkuang@cse.nsysu.edu.tw; m943040017@student.nsysu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2009.2032289

are possibly accumulated to produce a huge output error. The additional compensation might be performed to decrease the accumulated output error, but different applications need different compensation value and thus additional computations for calculating the compensation value and performing the compensation are required. To avoid these additional computations, the fixed-width multiplier with very small mean and mean-square errors is largely expected to obtain more accurate output data.

In this paper, we propose a high-accuracy error compensation circuit for the fixed-width modified Booth multiplier. The circuit makes the error distribution not only be symmetric to but also centralize in zero error as much as possible. Therefore, the mean and mean-square errors can be significantly reduced simultaneously so that the resultant fixed-width multiplier is suitable for different applications whose output data may be produced from a single multiplication or multiply-accumulation operations. To accomplish this goal, we first slightly modify the partial product matrix of Booth multiplication to reduce the partial product bits in the truncated portion of DTFM. Then, the correlation between Booth encoded outputs and the truncated product error of DTFM is analyzed and explored to derive an effective and simple error compensation function, which can produce an approximation to the carry value generated by truncated portion of DTFM, to reduce the truncation error and make the error distribution as symmetric and centralized as possible. Finally, a simple compensation circuit composed of a simplified sorting network and some adder cells is developed according to the proposed error compensation function. Simulation and implementation results show that the proposed fixed-width modified Booth multiplier actually achieves much higher accuracy than existing fixed-width modified Booth multipliers [15], [16] in terms of the mean error and the mean-square error while maintaining the approximate hardware overhead.

This paper is organized as follows. In Section II, the modified Booth multiplier is briefly reviewed. Section III describes the detailed derivation of the proposed error compensation function and the corresponding compensation circuit. This section also compares the error performance of different fixed-width modified Booth multipliers. The implementation results and output quality of real-life applications for different fixed-width modified Booth multipliers are presented in Section IV. Finally, Section V concludes this paper.

II. Fundamental of Modified Booth Multiplier

Let us consider the multiplication operation of two n-bit signed numbers $A = a_{n-1}a_{n-2}\cdots a_0$ (multiplicand) and $B = b_{n-1}b_{n-2}\cdots b_0$ (multiplier). The two's complement representations of A and B can be expressed as follows:

$$A = -a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i, \quad B = -b_{n-1}2^{n-1} + \sum_{i=0}^{n-2} b_i 2^i. \tag{1}$$

TABLE I
Modified Booth Encoding Table

b_{2i+1}	b_{2i}	b_{2i-1}	Operation	neg_i	two_i	one_i	$zero_i$	cor_i
0	0	0	+0	0	0	0	1	0
0	0	1	+A	0	0	1	0	0
0	1	0	+A	0	0	1	0	0
0	1	1	+2A	0	1	0	0	0
1	0	0	−2A	1	1	0	0	1
1	0	1	−A	1	0	1	0	1
1	1	0	−A	1	0	1	0	1
1	1	1	−0	1	0	0	1	0

By modified Booth encoding which groups the bits of the multiplier into triplets, B can be expressed as

$$B = \sum_{i=0}^{n/2-1} M_i 2^{2i} = \sum_{i=0}^{n/2-1} (-2b_{2i+1} + b_{2i} + b_{2i-1})2^{2i} \tag{2}$$

where $b_{-1} = 0$ and $M_i \in \{-2, -1, 0, 1, 2\}$. Based on the encoded result shown in Table I, the Booth encoder and partial product generation circuit proposed in [4] (depicted in Fig. 1(a) and 1(b), respectively) are adopted to choose one of multiple multiplicands $-2A$, $-A$, 0, A, and $2A$ for generating each partial product row PP_i, where $0 \le i \le n/2 - 1$ and $\overline{a_j}$ is the complement of a_j $(0 \le j \le n-1)$. The $2A$ in Table I is realized by left shifting A one bit. As for the negation operation, each bit of A is inverted and an extra binary value "1" is added to the least significant bit of next partial product row. Adding "1" can be implemented as a correction bit cor_i which indicates that the partial product row PP_i is positive $(cor_i = 0)$ or negative $(cor_i = 1)$. Furthermore, the sign bit for each partial product row PP_i must be properly extended up to the $(2n-1)$th bit position because each partial product row is represented in two's complementation. These additional sign bits greatly affect the performance and power consumption of the parallel multiplier when n increases gradually. Consequently, many approaches [17], [18] have been proposed to solve this problem. Fig. 2 illustrates the partial product matrix of an 8×8 modified Booth multiplier with sign extension elimination technique, where $p_{i,j}$ and s_i denote the jth product bit and the sign bit of partial product row PP_i, respectively. The partial product matrix can be segmented into MP and LP, where LP is further divided into LP_{major} and LP_{minor}. The partial product bits in LP can be directly omitted to form a DTFM, but the truncation error is very large and unacceptable for many applications. In the proposed high-accuracy fixed-width modified Booth multiplier, only the partial product bits in LP_{minor} are removed and the carry value propagated from LP_{minor} to LP_{major} must be estimated by a simple circuit to compensate for the truncation error.

III. Proposed Fixed-Width Modified Booth Multiplier

In this section, because the values of partial product bits are heavily dependent on the outputs of Booth encoders, we first explore the relation between the outputs of Booth encoders and the

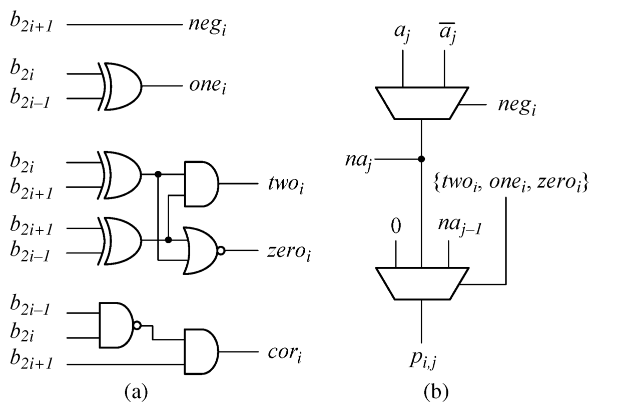
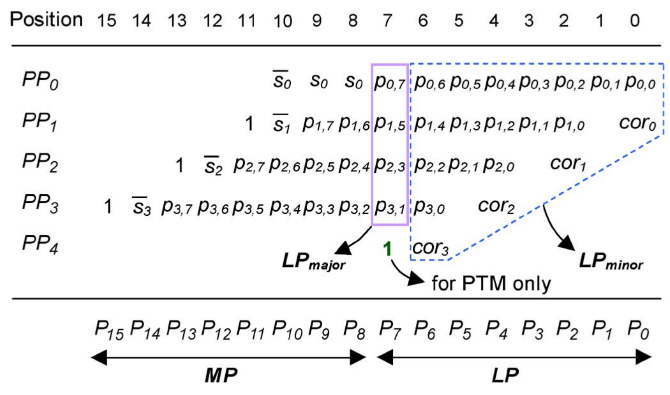


Fig. 1. (a) Modified Booth encoder. (b) Partial product generation circuit.



Position 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
PP_0 $\overline{s_0}$ s_0 s_0 $p_{0,7}$ $p_{0,6}$ $p_{0,5}$ $p_{0,4}$ $p_{0,3}$ $p_{0,2}$ $p_{0,1}$ $p_{0,0}$
PP_1 1 $\overline{s_1}$ $p_{1,7}$ $p_{1,6}$ $p_{1,5}$ $p_{1,4}$ $p_{1,3}$ $p_{1,2}$ $p_{1,1}$ $p_{1,0}$ cor_0
PP_2 1 $\overline{s_2}$ $p_{2,7}$ $p_{2,6}$ $p_{2,5}$ $p_{2,4}$ $p_{2,3}$ $p_{2,2}$ $p_{2,1}$ $p_{2,0}$ cor_1
PP_3 1 $\overline{s_3}$ $p_{3,7}$ $p_{3,6}$ $p_{3,5}$ $p_{3,4}$ $p_{3,3}$ $p_{3,2}$ $p_{3,1}$ $p_{3,0}$ cor_2
PP_4 1 cor_3
LP_{major}
for PTM only
LP_{minor}
P_{15} P_{14} P_{13} P_{12} P_{11} P_{10} P_9 P_8 P_7 P_6 P_5 P_4 P_3 P_2 P_1 P_0
MP
LP

Fig. 2. Partial product matrix for 8 × 8 modified Booth multiplication.

carry value propagated from LP_{minor} to LP_{major}. Next, an effective and simple error compensation function, which takes the outputs of Booth encoders as inputs and then generates the approximate carry value, is derived to reduce the truncation error and make the error distribution as symmetric and centralized as possible. Finally, an uncomplicated and fast compensation circuit is constructed to form a high-accuracy fixed-width multiplier.

A. *Proposed Error Compensation Function*

Let $SUM(MP)$ and $SUM(LP)$ represent the sum of partial product bits in MP and LP, respectively, then the $2n$-bit output product P can be expressed as

$$P = A \times B = SUM(MP) + SUM(LP). \tag{3}$$

The $SUM(LP)$ in (3) can be calculated as

$$SUM(LP) = \delta \times 2^n \tag{4}$$

where

$$\delta = \frac{1}{2}(p_{0,n-1}+\ldots+p_{n/2-1,1}) + \frac{1}{2^2}(p_{0,n-2}+\ldots+cor_{n/2-1}) + \ldots + \frac{1}{2^{n-1}}(p_{0,1}) + \frac{1}{2^n}(p_{0,0}+cor_0). \tag{5}$$

Let $\theta_{\text{major}} = p_{0,n-1}+\ldots+p_{n/2-1,1}$, (5) can be rewritten as

$$\delta = \frac{1}{2}\theta_{\text{major}} + \theta_{\text{minor}} \tag{6}$$

where

$$\theta_{\text{minor}} = \frac{1}{2^2}(p_{0,n-2}+\ldots+cor_{n/2-1}) + \ldots + \frac{1}{2^n}(p_{0,0}+cor_0). \tag{7}$$

If the partial products in LP are removed, the most accurate error compensation value $\hat{\delta}$ is the rounding value of δ that can be expressed as

$$\hat{\delta} = \left[\frac{1}{2}\theta_{\text{major}} + \theta_{\text{minor}}\right]_r = \left\lfloor \frac{1}{2}(\theta_{\text{major}}+1) + \theta_{\text{minor}} \right\rfloor \tag{8}$$

where $[t]_r$ and $\lfloor t \rfloor$ denote the rounding and floor operations for t. However, only the post-truncated modified Booth multiplier (PTM) can generate the most accurate error compensation value $\hat{\delta}$. A PTM can be achieved by adding an extra "1" at the $(n-1)$th bit position of partial product matrix as shown in Fig. 2, generating the carry value by LP_{minor}, and finally outputting the most significant n product bits. Unfortunately, the PTM results in the approximate hardware complexity and power consumption to the standard modified Booth multiplier. Therefore, we attempt to develop a simple error compensation function whose output value approximates the most accurate error compensation value $\hat{\delta}$ as follows.

Firstly, we modify the partial product matrix of PTM shown in Fig. 2 into a new matrix as shown in Fig. 3 to decrease the partial product bits in LP_{minor} so that the carry value generated by LP_{minor} can be estimated more accurately and easily. The new matrix is obtained by adding the least significant bit $p_{n/2-1,0}$ of $PP_{n/2-1}$ and $cor_{n/2-1}$ in advance to generate a sum $\varepsilon_{n/2-1,0}$ and a carry λ at the $(n-2)$th and $(n-1)$th bit positions, respectively. The logic expressions for $\varepsilon_{n/2-1,0}$ and λ can be derived and simplified from the truth table shown in Table II as follows:

$$\varepsilon_{n/2-1,0} = a_0 \wedge one_{n/2-1} \tag{9}$$

$$\lambda = (\overline{a_0 \wedge one_{n/2-1}}) \wedge \overline{zero_{n/2-1}} \wedge b_{n-1} \tag{10}$$

where $\wedge$ denotes AND operation. Since the weight of the extra "1" located at the $(n-1)$th bit position of the PTM is equal to the weight of λ, they can be added up to generate a sum $\overline{\lambda}$ (i.e., the complement of λ) and a carry λ which must be propagated to the nth bit position. Then, the carry λ can be incorporated with the sign extension bits $\overline{s_0}s_0s_0$ of PP_0 to produce the new partial product bits $\omega_2\omega_1\omega_0$ based on the truth table shown in Table III. According to Table III, the simplified logic expression for carry $\overline{\lambda}$ and $\omega_2\omega_1\omega_0$ can be expressed as

$$\overline{\lambda} = \overline{[(\overline{a_0 \wedge one_{n/2-1}}) \wedge \overline{zero_{n/2-1}} \wedge b_{n-1}]} \tag{11}$$

$$\omega_2 = \overline{(s_0 \wedge \overline{\lambda})} \tag{12}$$

$$\omega_1 = \overline{\omega_2} \tag{13}$$

$$\omega_0 = \overline{[(s_0 \vee \overline{\lambda}) \wedge \omega_2]} \tag{14}$$

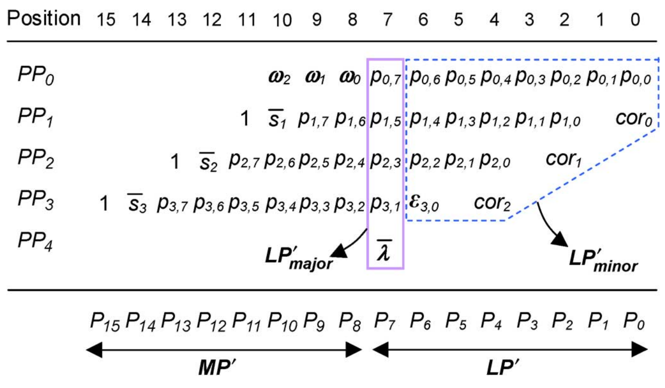

Position 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
PP0 ω2 ω1 ω0 p0,7 p0,6 p0,5 p0,4 p0,3 p0,2 p0,1 p0,0
PP1 1 s̄1 p1,7 p1,6 p1,5 p1,4 p1,3 p1,2 p1,1 p1,0 cor0
PP2 1 s̄2 p2,7 p2,6 p2,5 p2,4 p2,3 p2,2 p2,1 p2,0 cor1
PP3 1 s̄3 p3,7 p3,6 p3,5 p3,4 p3,3 p3,2 p3,1 ε3,0 cor2
PP4 λ̄
LP'major
LP'minor
P15 P14 P13 P12 P11 P10 P9 P8 P7 P6 P5 P4 P3 P2 P1 P0
MP'
LP'

Fig. 3. The proposed 8 × 8 modified Booth partial product matrix.

TABLE II
TRUTH TABLE FOR $\varepsilon_{n/2-1,0}$ AND λ

b_{n-1}	b_{n-2}	b_{n-3}	$two_{n/2-1}$	$one_{n/2-1}$	$zero_{n/2-1}$	$cor_{n/2-1}$	$p_{n/2-1,0}$	$\varepsilon_{n/2-1,0}$	λ
0	0	0	0	0	1	0	0	0	0
0	0	1	0	1	0	0	a_0	a_0	0
0	1	0	0	1	0	0	a_0	a_0	0
0	1	1	1	0	0	0	0	0	0
1	0	0	1	0	0	1	1	0	1
1	0	1	0	1	0	1	$\overline{a_0}$	a_0	$\overline{a_0}$
1	1	0	0	1	0	1	$\overline{a_0}$	a_0	$\overline{a_0}$
1	1	1	0	0	1	0	0	0	0

TABLE III
TRUTH TABLE FOR SIGN EXTENSION BITS $\omega_2\omega_1\omega_0$

$\overline{s_0}$	s_0	s_0	λ	ω_2	ω_1	ω_0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	1	0	0
1	0	0	1	1	0	1

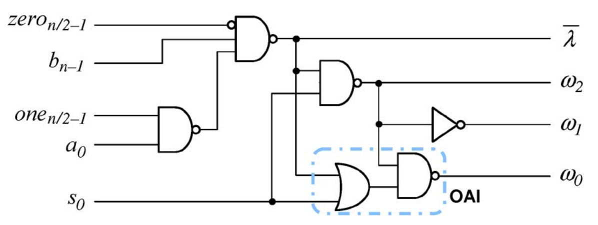


Fig. 4. The circuit to produce $\overline{\lambda}$ and $\omega_2\omega_1\omega_0$.

where $\vee$ denotes OR operation. The corresponding circuit for generating $\overline{\lambda}$, ω_2, ω_1, and ω_0 is depicted in Fig. 4.

As shown in Fig. 3, we can define an error compensation function $\widetilde{\delta}$ as follows:

$$\widetilde{\delta} = \left\lfloor \frac{1}{2}\left[(\theta_{\text{major}} + \overline{\lambda}) + CA\left(LP'_{\text{minor}}\right)\right]\right\rfloor = \left\lfloor \frac{1}{2}\left[\theta'_{\text{major}} + CA\left(LP'_{\text{minor}}\right)\right]\right\rfloor \tag{15}$$

where $\theta'_{\text{major}} = \theta_{\text{major}} + \overline{\lambda}$ and $CA(LP'_{\text{minor}})$ represents the approximate carry value propagated from LP'_{minor} to LP'_{major}. Subsequently, the correlation between $CA(LP'_{\text{minor}})$ and the outputs of Booth encoder (i.e., $zero_i$ for $0 \le i \le n/2-1$) will be explored. From Table I, it can be seen that if the encoder output $zero_i$ is one, the partial product bits of PP_i in LP'_{minor} of Fig. 3 must be equal to zero. To represent the different combination of $zero_i$ for $0 \le i \le n/2-1$, a generalized index $\varPhi$ is defined as

$$\varPhi = \overline{zero_{n/2-1}} \times 2^{n/2-1} + \overline{zero_{n/2-2}} \times 2^{n/2-2} + \ldots + \overline{zero_0} \times 2^0. \tag{16}$$

According to (16), the range of $\varPhi$ is from 0 to $2^{n/2}-1$. For example, $\varPhi = 10$ if $n = 8$ and $(zero_3, zero_2, zero_1, zero_0) = (0,1,0,1)$. Because a specific $\varPhi$ can be produced by different B, we define $\tau_B(\varPhi)$ and $\tau_{AB}(\varPhi)$ as the set of input B and the set of input pair $(A;B)$ that can produce the specific $\varPhi$. Let $N(\tau_B(\varPhi))$ indicate the number of elements in $\tau_B(\varPhi)$, then $N(\tau_{AB}(\varPhi))$ is equal to $N(\tau_B(\varPhi)) \times 2^n$ and all $(A;B)$ pairs in $\tau_{AB}(\varPhi)$ can be utilized to calculate the average value of $SUM(LP'_{\text{minor}})$ for a specific $\varPhi$, denoted as $\widetilde{S}(\varPhi)$, as follows:

$$\widetilde{S}(\Phi) = \frac{1}{N\left(\tau_{AB}(\Phi)\right)} \sum_{(A;B)\in\tau_{AB}(\varPhi)} SUM\left(LP'_{\text{minor}}\right) \tag{17}$$

where $SUM(LP'_{\text{minor}})$ is the sum of partial product bits in LP'_{minor}. If the value of $\widetilde{S}(\varPhi)$ in (17) is normalized by 2^{n-1}, then $\widetilde{S}(\varPhi)/2^{n-1}$ can substitute for $CA(LP'_{\text{minor}})$ and thus (15) can be rewritten as

$$\widetilde{\delta} = \left\lfloor \frac{1}{2}\left(\theta'_{\text{major}} + \widetilde{S}(\varPhi)/2^{n-1}\right)\right\rfloor = \lfloor \beta \rfloor. \tag{18}$$

Subsequently, we will further investigate the correlation between $\varPhi$ and $\widetilde{S}(\varPhi)/2^{n-1}$ and then design the corresponding error compensation circuit.

Taking the 8 × 8 fixed-width modified Booth multiplier as an example, we compute the value of $\widetilde{S}(\varPhi)/2^{n-1}$ for all possible indices $\varPhi$ through performing exhaustive simulation. Table IV shows the value of $\widetilde{S}(\varPhi)/2^{n-1}$ for different index $\varPhi$. Furthermore, $\widetilde{\delta}$ can be computed by considering all combinations of θ'_{major} and $\widetilde{S}(\varPhi)/2^{n-1}$ according to (18). Let $(\varPhi, \theta'_{\text{major}})$ denote the combination of $\varPhi$ and θ'_{major}, then β and $\widetilde{\delta}$ for all possible $(\varPhi, \theta'_{\text{major}})$ with $n = 8$ can be calculated and listed in Table V. For instance, when $(\varPhi, \theta'_{\text{major}}) = (2,1)$, $2\beta = \widetilde{S}(2)/2^{n-1} + 1 = 0.5104 + 1 = 1.5104$ so that $\widetilde{\delta} = \lfloor\beta\rfloor = \lfloor 0.7552 \rfloor = 0$. As can be seen from Table V and Fig. 3, θ'_{major} is always an integer. Moreover, $\widetilde{\delta}$ is also an integer. Therefore, based on the results shown in Table V, the contribution of $\widetilde{S}(\varPhi)/2^{n-1}$ to $\widetilde{\delta}$ can be approximated to an integer $s(\varPhi)$ as follows:

$$s(\varPhi) = \left\lfloor \widetilde{S}(\varPhi)/2^{n-1} \right\rceil. \tag{19}$$

Table IV shows $s(\varPhi)$ for different $\varPhi$ when $n = 8$. In addition, Table IV also shows the relation between $\varPhi$ and $\Re$, where $\Re$ is

TABLE IV
VALUES OF $\widetilde{S}(\Phi)/2^{n-1}$, $s(\Phi)$, AND $\Re$ FOR DIFFERENT INDEX Φ

Φ	$\widetilde{S}(\Phi)/2^{n-1}$	$s(\Phi)$	$\Re$	Φ	$\widetilde{S}(\Phi)/2^{n-1}$	$s(\Phi)$	$\Re$
0	0	0	0	8	0.1666	0	1
1	0.5026	0	1	9	0.6692	0	2
2	0.5104	0	1	10	0.6770	0	2
3	0.9990	0	2	11	1.1656	1	3
4	0.5416	0	1	12	0.7082	0	2
5	0.9886	0	2	13	1.1552	1	3
6	0.9964	0	2	14	1.1630	1	3
7	1.5036	1	3	15	1.6704	1	4

the summation of $\overline{zero_i}$ for $0 \le i \le n/2-1$. That is, $\Re$ can be expressed as

$$\Re = \sum_{i=0}^{n/2-1} \overline{zero_i}. \tag{20}$$

For example, $\Re = 3$ if $n = 8$, $\Phi = 11$, and $(\overline{zero_3}, \overline{zero_2}, \overline{zero_1}, \overline{zero_0}) = (1, 0, 1, 1)$. According to $\Re$ and $s(\Phi)$ shown in Table IV, their relation can be concluded as

$$s(\Phi) = \lfloor (\Re - 1)/2 \rfloor. \tag{21}$$

By the similar method, $s(\Phi)$ for each $\Re$ with different n ($n =$ 10, 12, 14, and 16) can also be obtained and shown in Table VI. Substituting (21) into (18), we get

$$\widetilde{\delta} = \left\lfloor \frac{1}{2}\left(\theta'_{\text{major}} + \lfloor (\Re - 1)/2 \rfloor\right) \right\rfloor. \tag{22}$$

That is, $\widetilde{\delta}$ is equal to the carries of adding up θ'_{major} and $\lfloor (\Re - 1)/2 \rfloor$.

B. Proposed Low Error Compensation Circuit

According to (22), θ'_{major} and $\lfloor (\Re - 1)/2 \rfloor$ can be compressed with the partial product bits in MP' through the compression tree structure to generate the final fixed-width product. Since the partial product bits in LP'_{major} and MP' can be simultaneously produced by partial product generation circuits, the problem becomes how to design a simple and efficient circuit, denoted as SC-generator, to produce $\lfloor (\Re - 1)/2 \rfloor$ quickly. By (20) and Table VI, the inputs of SC-generator are $\overline{zero_i}$ for $0 \le i \le n/2-1$ and it will generate m output bits $\alpha_1, \alpha_2, \cdots,$ and α_m as shown in Fig. 5, where $m = \lfloor (n/2 - 1)/2 \rfloor$ and $s(\Phi)$ will be equal to $\alpha_1 + \alpha_2 + \cdots + \alpha_m$. Due to the subtraction operation in $\lfloor (\Re - 1)/2 \rfloor$, it is difficult to generate $\alpha_1, \alpha_2, \cdots, \alpha_m$ by adder cells directly. Instead of adder cells, the proposed SC-generator is composed of a sorting network based on the following observation. We assume that $\overline{zero_i}$ for $0 \le i \le n/2-1$ can be sorted and the sorted outputs are ρ_j for $0 \le j \le n/2-1$. Moreover, if the largest bits (i.e., bits equal to "1") are gathered to the less significant positions (i.e., $\rho_{n/2-1} \le \rho_{n/2-2} \le \cdots \le \rho_1 \le \rho_0$), then $\alpha_k = \rho_{2k}$ for $1 \le k \le m$. That is, the problem of designing a SC-generator can be translated into the design of a sorting network that sorts $\overline{zero_{n/2-1}} \cdots \overline{zero_1}\,\overline{zero_0}$ into $\rho_{n/2-1} \cdots \rho_1 \rho_0$ and $\alpha_k = \rho_{2k}$.

TABLE V
VALUES OF β AND $\widetilde{\delta}$ FOR EACH COMBINATION $(\Phi, \theta'_{\text{major}})$ WITH $n = 8$

(Φ, θ'_{major})	β	$\widetilde{\delta}$	(Φ, θ'_{major})	β	$\widetilde{\delta}$
(0, 1)	0.5000	0	(10, 1)	0.8385	0
(1, 1)	0.7513	0	(10, 2)	1.3385	1
(1, 2)	1.2513	1	(10, 3)	1.8385	1
(2, 1)	0.7552	0	(11, 0)	0.5828	0
(2, 2)	1.2552	1	(11, 1)	1.0828	1
(3, 1)	0.9995	0	(11, 2)	1.5828	1
(3, 2)	1.4995	1	(11, 3)	2.0828	2
(3, 3)	1.9995	1	(11, 4)	2.5828	2
(4, 1)	0.7708	0	(12, 0)	0.3541	0
(4, 2)	1.2708	1	(12, 1)	0.8541	0
(5, 1)	0.9943	0	(12, 2)	1.3541	1
(5, 2)	1.4943	1	(12, 3)	1.8541	1
(5, 3)	1.9943	1	(13, 0)	0.5776	0
(6, 1)	0.9982	0	(13, 1)	1.0776	1
(6, 2)	1.4982	1	(13, 2)	1.5776	1
(6, 3)	1.9982	1	(13, 3)	2.0776	2
(7, 1)	1.2518	1	(13, 4)	2.5776	2
(7, 2)	1.7518	1	(14, 0)	0.5815	0
(7, 3)	2.2518	2	(14, 1)	1.0815	1
(7, 4)	2.7518	2	(14, 2)	1.5815	1
(8, 0)	0.0833	0	(14, 3)	2.0815	2
(8, 1)	0.5833	0	(14, 4)	2.5815	2
(8, 2)	1.0833	1	(15, 0)	0.8352	0
(9, 0)	0.3346	0	(15, 1)	1.3352	1
(9, 1)	0.8346	0	(15, 2)	1.8352	1
(9, 2)	1.3346	1	(15, 3)	2.3352	2
(9, 3)	1.8346	1	(15, 4)	2.8352	2
(10, 0)	0.3385	0	(15, 5)	3.3352	3

TABLE VI
RELATION BETWEEN $s(\Phi)$ AND $\Re$ FOR DIFFERENT n

$\Re$	$s(\Phi)$				
	$n=8$	$n=10$	$n=12$	$n=14$	$n=16$
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	1	1	1	1	1
4	1	1	1	1	1
5	-	2	2	2	2
6	-	-	2	2	2
7	-	-	-	3	3
8	-	-	-	-	3

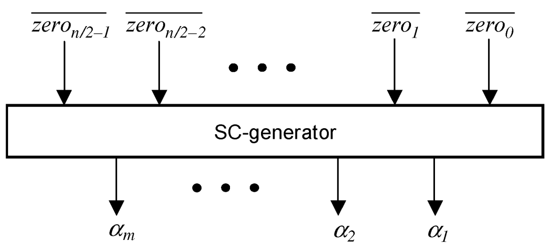


Fig. 5. The inputs and outputs of proposed SC-generator.

There are two kinds of well-known comparison-based sorting networks [19], the bitonic and the odd-even merge sorting networks, suited to hardware implementation. Since the odd-even merge sorter has the same number of compare levels as the bitonic sorter but requires fewer comparators, thus we adopt and simplify the odd-even merge sorting network to realize the

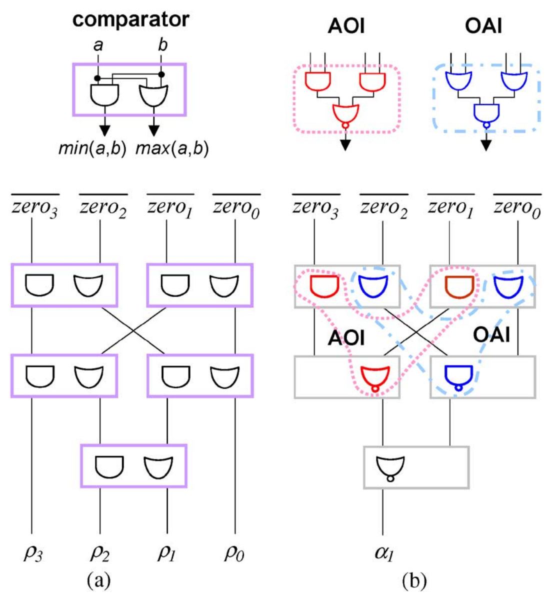


Fig. 6. (a) Odd-even merge sorting network for $n = 8$. (b) Proposed SC-generator for $n = 8$.

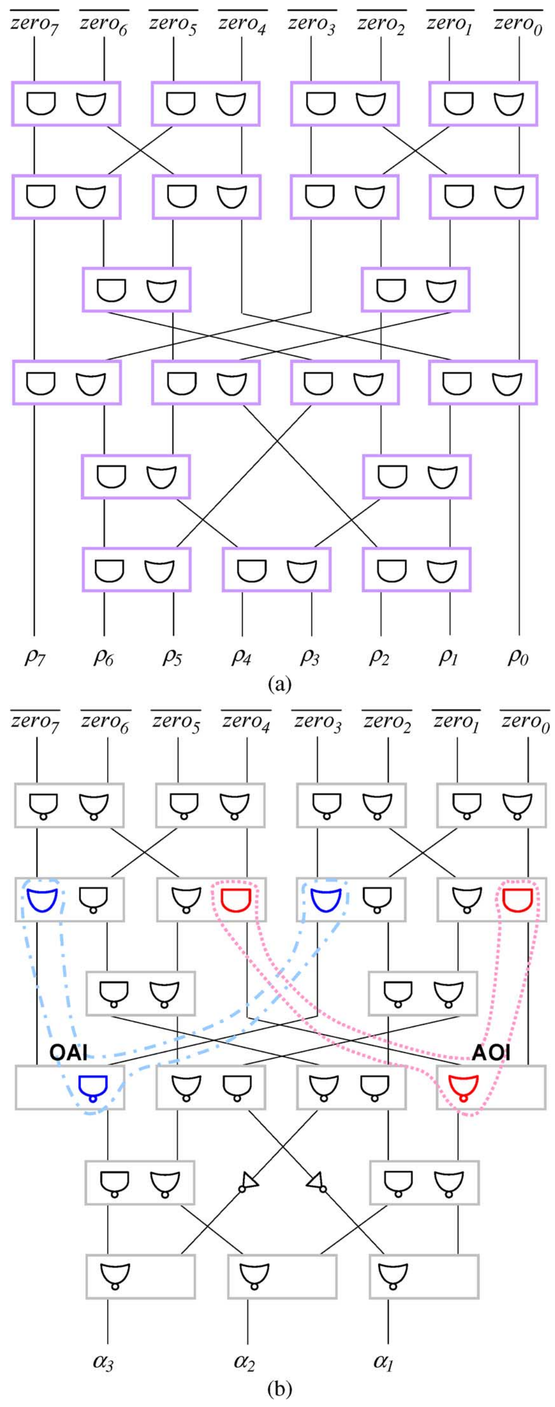


Fig. 7. (a) The odd-even merge sorting network for $n = 16$. (b) The proposed SC-generator for $n = 16$.

SC-generator. Figs. 6(a) and 7(a) illustrate the odd-even merge sorting networks [20] for the case of $n = 8$ and 16, respectively. These sorting networks are composed of appropriately connected comparators. Each comparator takes in two input bits and either passes them directly or switches them. With inputs a and b, the outputs $max(a,b)$ and $min(a,b)$ of comparator correspond to $(a \vee b)$ and $(a \wedge b)$, respectively. The structure of a comparator is shown in Fig. 6(a). After sorting for $n = 16$, α_1, α_2, and α_3 are equal to the sorted outputs ρ_2, ρ_4, and ρ_6 as shown in Fig. 7(a). Since ρ_0, ρ_1, ρ_3, ρ_5, and ρ_7 are unnecessary for a SC-generator, thus the logic gates only for producing these outputs can be removed to simplify the SC-generator. Besides, the sorting networks can be further simplified by using NAND, NOR, AND-OR INVERTER (AOI), and OR-AND-INVERTER (OAI) gates as shown in Figs. 6(b) and 7(b) for $n = 8$ and 16, respectively. The SC-generator for different n can be constructed in a similar fashion. After the estimated carries $\alpha_1, \alpha_2, \cdots$, and α_m are generated by SC-generator, they are fed into the $LP'_{\rm major}$ to produce the final fixed-width product. Fig. 8 illustrates the final partial product matrix of the proposed fixed-width Booth multiplier for $n = 8$. In it, all the partial product bits in $LP'_{\rm minor}$ are removed and replaced by the SC-generator. In addition, the carries generated by $LP'_{\rm minor}$ are also replaced by the outputs of SC-generator.

C. Error Performance

To analyze the accuracy of the different fixed-width modified Booth multipliers, the error metrics in terms of the normalized maximum absolute error ($\varepsilon_{\max}$), the normalized mean error ($\varepsilon_{\rm mean}$), and the normalized mean-square error ($\varepsilon_{\rm mse}$) are expressed as

$$\begin{aligned}\varepsilon_{\max} &= \mathrm{Max}\left\{|\mathrm{P}-\mathrm{P_t}|\right\}/2^{\mathrm{n}}\\ \varepsilon_{\rm mean} &= \mathrm{Ave}\{\mathrm{P}-\mathrm{P_t}\}/2^{\mathrm{n}}\\ \varepsilon_{\rm mse} &= \mathrm{Ave}\left\{(\mathrm{P}-\mathrm{P_t})^2\right\}/2^{2\mathrm{n}}\end{aligned} \quad (23)$$

where Max, Ave, P, and P_t represent the maximal operator, the average operator, the output product of the standard modified Booth multiplier, and the output product of the fixed-width modified Booth multiplier, respectively. The exhaustive simulation results in terms of $\varepsilon_{\max}$, $\varepsilon_{\rm mean}$, and $\varepsilon_{\rm mse}$ for different fixed-width modified Booth multipliers with $n = 8$, 10, 12, 14, and 16 are listed in Table VII, where JFM, CFM, and SFM denote the Jou's fixed-width multiplier [14], Cho's fixed-width multiplier [15], and Song's fixed-width multiplier [16], respectively. The results show that the proposed error compensation

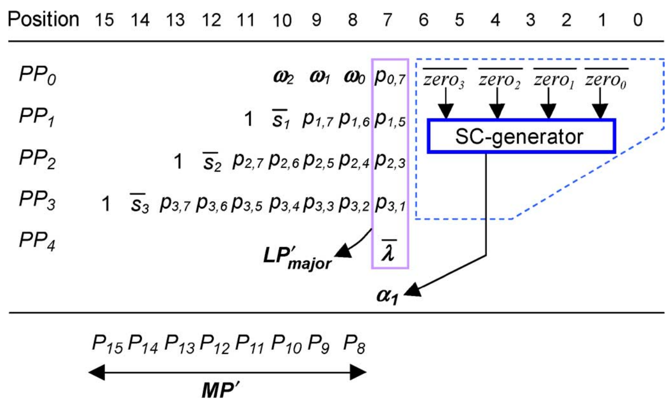


Fig. 8. Final partial product matrix of proposed fixed-width modified Booth multiplier for $n = 8$.

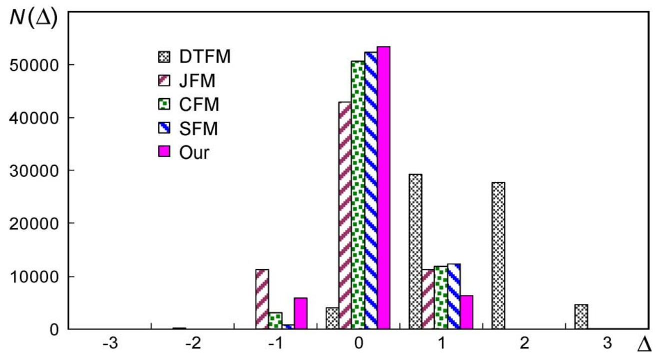


Fig. 9. The error distribution of different fixed-width multipliers for $n = 8$.

TABLE VII
ERROR PERFORMANCE OF DIFFERENT FIXED-WIDTH MULTIPLIERS

n	*multiplier*	ε_{max}	ε_{mean}	ε_{mse}
8	PTM	0.5000	0	0.0833
	DTFM	4.0000	1.5010	2.6880
	JFM	1.7305	0.0010	0.2717
	CFM	1.5000	0.1328	0.1664
	SFM	1.1641	0.1758	0.1459
	Our	1.1680	0.0078	0.1367
10	PTM	0.5000	0	0.0833
	DTFM	5.0000	1.8752	4.0563
	JFM	2.1299	0.0002	0.3368
	CFM	1.5000	0.1211	0.1713
	SFM	1.3652	0.2028	0.1713
	Our	1.5000	–0.0039	0.1542
12	PTM	0.5000	0	0.0833
	DTFM	6.0000	2.25006	5.7068
	JFM	2.5300	0.00006	0.4019
	CFM	2.0000	0.12695	0.1950
	SFM	1.5649	0.22032	0.1960
	Our	1.6667	0.00195	0.1671
14	PTM	0.5000	0	0.0833
	DTFM	7.0000	2.625015	7.6390
	JFM	2.9300	0.000015	0.4670
	CFM	2.0000	0.124023	0.2088
	SFM	1.7650	0.231404	0.2192
	Our	2.0000	–0.000977	0.1821
16	PTM	0.5000	0	0.0833
	DTFM	8.0000	3.000004	9.8525
	JFM	3.3300	0.000004	0.5321
	CFM	2.5000	0.125488	0.2235
	SFM	1.9650	0.238366	0.2409
	Our	2.1667	0.000488	0.1961

circuit not only leads the fixed-width modified Booth multiplier to a very small mean error but also significantly reduces the mean-square error when compared to other circuits. For 16-bit and 8-bit fixed-width modified Booth multipliers, the proposed circuit can achieve at least 12.3% and 6.3% reductions in mean-square error over the previous circuits, respectively.

To further show the strength of the proposed circuit, the difference Δ between the output products of different fixed-width multipliers and the PTM are computed. Let P_r be the output product of the PTM, the error Δ is calculated as $\Delta = (P_r - P_t)/2^n$ and the number of error Δ, denoted as $N(\Delta)$, for different fixed-width multipliers with different n are listed in Table VIII. In addition, the error histograms of these fixed-width multipliers for $n = 8$ are also illustrated in Fig. 9, where the horizontal axis and the vertical axis represent Δ and $N(\Delta)$, respectively. As can be seen from Table VIII and Fig. 9, the proposed circuit can make the error distribution of fixed-width modified Booth multiplier be symmetric to and centralized in $\Delta = 0$ since a high-accuracy compensation function has been developed according to the modified partial product matrix of Booth multiplication shown in Fig. 3 as mentioned earlier. Accordingly, the mean and mean-square errors can be significantly reduced simultaneously so that the proposed fixed-width modified Booth multiplier is well-suited to different applications. In addition, two real-life applications are picked to further exhibit the accuracy of the proposed circuit in the next section.

IV. EXPERIMENTAL RESULTS

To compare the characteristics of different error compensation circuits, we have implemented several fixed-width modified Booth multipliers with Dadda tree [21], [22] for $n = 8$, 10, 12, 14, and 16 in Verilog HDL. These fixed-width Booth multipliers were synthesized through utilizing Synopsys Design Compiler with the Artisan TSMC 0.13 μm CMOS standard cell technology library. The synthesized netlists were then fed into Cadence SOC Encounter to accomplish the placement and routing [23]. Delay estimations were obtained behind RC extraction from the placed and routed netlists. Moreover, Synopsys Nanosim is adopted to perform full transistor-level power simulation at a clock frequency of 125 MHz and 1.2 V with 1000 random input patterns. The implementation results including hardware area (Area), critical path delay (Delay), and power consumption (Power) of different fixed-width Booth multipliers are shown in Table IX. The results exhibit that the area, delay, and power consumption of the proposed multiplier are slightly larger than JFM, but very approximate CFM and are less than SFM.

Additionally, two practical applications including 9/7 forward and inverse discrete wavelet transform (DWT/IDWT) [24] in JPEG2000 image compression and RGB to YUV conversion [25] are adopted to evaluate the error performance of these fixed-width Booth multipliers. For the DWT/IDWT, we take ten 64 × 64 gray images for experiment and the coefficients are set to the integer values represented by 16-bit two's complement form. By performing multiplication in two-dimensional

TABLE VIII
ERROR DISTRIBUTION OF DIFFERENT FIXED-WIDTH MULTIPLIERS

n	*multiplier*	$N(\Delta)$											
		$\Delta=-3$	$\Delta=-2$	$\Delta=-1$	$\Delta=0$	$\Delta=1$	$\Delta=2$	$\Delta=3$	$\Delta=4$	$\Delta=5$	$\Delta=6$	$\Delta=7$	$\Delta=8$
8	DTFM	0	0	0	4066	29240	27624	4544	62	0	0	0	0
	JFM	0	78	11196	42916	11276	70	0	0	0	0	0	0
	CFM	0	0	3120	50594	11820	2	0	0	0	0	0	0
	SFM	0	0	828	52360	12348	0	0	0	0	0	0	0
	Our	0	0	5804	53416	6316	0	0	0	0	0	0	0
12	DTFM	0	0	0	1.70e+05	2.86e+06	7.46e+06	5.22e+06	1.02e+06	0.04e+06	139	0	0
	JFM	8	1.36e+05	3.52e+06	9.46e+06	3.53e+06	1.35e+05	8	0	0	0	0	0
	CFM	0	8	1.17e+06	1.23e+07	3.30e+06	3143	0	0	0	0	0	0
	SFM	0	0	0.46e+06	1.22e+07	4.16e+06	153	0	0	0	0	0	0
	Our	0	12	1.93e+06	1.29e+07	1.96e+06	116	0	0	0	0	0	0
16	DTFM	0	0	0	6.31e+06	2.07e+08	1.08e+09	1.74e+09	1.02e+09	2.22e+08	1.54e+07	2.32e+05	252
	JFM	6.41e+05	8.24e+07	9.86e+08	2.16e+09	9.86e+08	8.24e+07	6.44e+05	0	0	0	0	0
	CFM	0	1.89e+05	3.74e+08	3.01e+09	9.07e+08	2.89e+06	2	0	0	0	0	0
	SFM	0	3970	1.84e+08	2.90e+09	1.20e+09	2.08e+06	0	0	0	0	0	0
	Our	0	4.45e+05	5.75e+08	3.14e+09	5.78e+08	4.19e+05	0	0	0	0	0	0

TABLE IX
EXPERIMENTAL RESULTS OF DIFFERENT DADDA TREE FIXED-WIDTH MULTIPLIERS

n	*multiplier*	Area (μm^2)	Delay (ns)	Power (mW)
8	PTM	3279	4.060	0.523
	DTFM	1497	2.975	0.200
	JFM	1816	3.803	0.269
	CFM	2008	3.543	0.313
	SFM	2193	4.249	0.328
	Our	2005	3.563	0.306
10	PTM	4912	4.576	0.849
	DTFM	2341	3.422	0.349
	JFM	2760	4.691	0.463
	CFM	2921	4.152	0.488
	SFM	3161	4.886	0.522
	Our	2918	4.240	0.476
12	PTM	6864	5.279	1.322
	DTFM	3266	4.215	0.508
	JFM	3745	5.138	0.662
	CFM	4025	4.732	0.695
	SFM	4274	5.163	0.726
	Our	4038	4.762	0.720
14	PTM	9135	5.699	1.830
	DTFM	4429	4.677	0.775
	JFM	4987	5.553	0.947
	CFM	5281	5.133	1.013
	SFM	5622	5.953	1.082
	Our	5296	5.239	1.028
16	PTM	11726	6.375	2.447
	DTFM	5673	4.952	1.012
	JFM	6392	6.318	1.260
	CFM	6666	5.830	1.304
	SFM	7034	6.417	1.375
	Our	6683	5.760	1.358

three-level DWT/IDWT with 16-bit fixed-width Booth multipliers, the reconstructed pixel values are compared with the original gray images. For RGB to YUV conversion, twenty images (each requires 7 077 888 multiplication operations) are picked for experiment and the coefficients are scaled so that they can be represented by 12-bit two's complement form. The final outputs generated by these fixed-width Booth multipliers are compared with the outputs obtained by the floating-point arithmetic operation. The output quality in terms of peak signal-to-noise ratio (PSNR) for different fixed-width Booth multipliers and different applications (DWT/IDWT and RGB to YUV conversion) are listed in Tables X and XI, respectively. The results exhibit that the proposed fixed-width Booth multipliers improve the average PSNR by at least 2.0 dB and 1.1 dB over that of the previous fixed-width Booth multipliers, respectively.

TABLE X
PSNR RESULTS FOR DWT/IDWT

Images	DTFM	JFM	CFM	SFM	Our
1.raw	27.45	44.94	44.84	42.45	46.76
2.raw	27.09	45.96	44.28	42.53	46.09
3.raw	27.76	44.55	45.77	42.17	47.75
4.raw	27.52	45.21	44.95	42.28	47.07
5.raw	27.58	44.73	45.32	42.45	47.37
6.raw	27.17	44.53	45.41	42.37	47.37
7.raw	27.87	43.72	45.74	42.59	47.80
8.raw	27.61	44.67	45.27	42.28	47.28
9.raw	27.55	44.48	45.20	42.47	47.36
10.raw	27.59	44.44	45.15	42.46	47.22
Average	27.54	44.72	45.19	42.40	47.21

V. CONCLUSION

In this paper, a high-accuracy fixed-width modified Booth multiplier has been proposed. In the proposed multiplier, the partial product matrix of Booth multiplication was slightly modified and an effective error compensation function was derived accordingly. This compensation function makes the error distribution be more symmetric to and centralized in the error equal to zero, leading the fixed-width modified Booth multiplier to very small mean and mean-square errors. In addition, a simplified sorting network has been designed to realize the compensation function. Implementation results showed that the proposed fixed-width modified Booth multiplier can achieve a tiny mean error and a considerable reduction in mean-square error without increasing the hardware overhead. For 16-bit and 8-bit fixed-width modified Booth multipliers, the proposed error compensation circuits offer at least 12.3% and 6.3% reductions in mean-square error over the previous circuits, respectively. Additionally, experimental results of two real-life applications also demonstrated that the proposed multiplier can improve the output quality in terms of PSNR by

TABLE XI
PSNR RESULTS FOR RGB TO YUV CONVERSION

Images	DTFM	JFM	CFM	SFM	Our
01.bmp	25.96	42.05	44.29	43.15	45.43
02.bmp	25.76	43.23	44.94	44.07	46.45
03.bmp	26.01	42.64	44.27	42.97	45.42
04.bmp	25.86	41.73	44.63	43.21	45.88
05.bmp	25.61	42.04	44.26	43.42	45.42
06.bmp	26.06	42.56	44.33	43.05	45.40
07.bmp	26.06	42.48	44.54	43.07	45.55
08.bmp	26.05	42.03	43.87	43.08	44.90
09.bmp	26.31	42.35	44.36	43.16	45.47
10.bmp	25.92	42.02	44.68	43.09	45.72
11.bmp	25.76	42.16	44.32	43.40	45.49
12.bmp	25.38	42.41	44.14	43.68	45.16
13.bmp	25.98	42.12	44.40	43.05	45.46
14.bmp	25.55	42.24	43.63	43.50	44.65
15.bmp	25.60	42.05	44.37	43.77	45.51
16.bmp	25.79	42.02	44.56	43.57	45.58
17.bmp	25.96	41.81	44.61	42.98	45.65
18.bmp	26.08	42.39	44.43	42.84	45.40
19.bmp	26.07	42.13	44.38	42.82	45.27
20.bmp	25.94	42.20	44.66	43.23	45.75
Average	25.91	42.23	44.38	43.26	45.47

at least 2.0 dB and 1.1 dB on average, respectively. As a result, the proposed multipliers are applicable to lossy applications to reduce the area and power consumption of the whole system while maintaining good output quality.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their many constructive comments and suggestions in improving this paper. They would also like to thank the contributions of the National Chip Implementation Center (CIC), Taiwan, for their support in technology data.

REFERENCES

[1] G. O. Young, A. Inoue, R. Ohe, S. Kashiwakura, S. Mitarai, T. Tsuru, and T. Izawa, “A 4.1-ns compact 54 × 54 multiplier utilizing sign-select Booth encoders,” *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1676–1682, Nov. 1997.

[2] F. Elguibaly, “A fast parallel multiplier-accumulator using the modified Booth algorithm,” *IEEE Trans. Circuits Syst. II, Reg. Papers*, vol. 47, no. 9, pp. 902–908, Sep. 2000.

[3] W.-C. Yeh and C.-W. Jen, “High-speed Booth encoded parallel multiplier design,” *IEEE Trans. Computers*, vol. 49, no. 7, pp. 692–701, July 2000.

[4] K. Choi and M. Song, “Design of a high performance 32 × 32-bit multiplier with a novel sign select Booth encoder,” in *Proc. IEEE Int. Symp. Circuits and Systems*, 2001, vol. 2, pp. 701–704.

[5] Y. C. Lim, “Single precision multiplier with reduced circuit complexity for signal processing applications,” *IEEE Trans. Computer*, vol. 41, no. 10, pp. 1333–1336, Oct. 1992.

[6] M. J. Schulte and E. E. Swartzlander, Jr., “Truncated multiplication with correction constant,” in *Proc. VLSI Signal Processing, VI*, New York, 1993, pp. 388–396.

[7] S. S. Kidambi, F. El-Guibaly, and A. Antoniou, “Area-efficient multipliers for digital signal processing applications,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 43, no. 2, pp. 90–94, Feb. 1996.

[8] J. M. Jou, S. R. Kuang, and R. D. Chen, “Design of low-error fixed-width multipliers for DSP applications,” *IEEE Trans. Circuits Syst. I, Exp. Briefs*, vol. 46, no. 6, pp. 836–842, June 1999.

[9] L. D. Van, S. S. Wang, and W. S. Feng, “Design of the low error fixed-width multiplier and its application,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 47, no. 10, pp. 1112–1118, Oct. 2000.

[10] L. D. Van and C. C. Yang, “Generalized low-error area-efficient fixed-width multipliers,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 8, pp. 1608–1619, Aug. 2005.

[11] J.-S. Wang, C.-N. Kuo, and T.-H. Yang, “Low-power fixed width array multipliers,” in *Proc. Int. Symp. Low Power Electron. Des.*, 2004, pp. 307–312.

[12] Y.-C. Liao, H.-C. Chang, and C.-W. Liu, “Carry estimation for two’s complement fixed-width multipliers,” in *Proc. IEEE Workshop Signal Processing Systems*, 2006, pp. 345–350.

[13] A. G. M. strollo, N. Petra, and D. D. Caro, “Dual-tree error compensation for high performance fixed-width multipliers,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 8, pp. 501–507, Aug. 2005.

[14] S. J. Jou, M.-H. Tsai, and Y.-L. Tsao, “Low-error reduced-width Booth multipliers for DSP applications,” *IEEE Trans. Circuits Syst. I, Fudam. Theory Appl.*, vol. 50, no. 11, pp. 1470–1474, Nov. 2003.

[15] K.-J. Cho, K.-C. Lee, J.-G. Chung, and K. K. Parhi, “Design of low-error fixed-width modified Booth multiplier,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 5, pp. 522–531, May 2004.

[16] M.-A. Song, L.-D. Van, and S.-Y. Kuo, “Adaptive low-error fixed-width Booth multipliers,” *IEICE Trans. Fundamentals*, vol. E90-A, no. 6, pp. 1180–1187, Jun. 2007.

[17] E. de Angel and E. E. Swartzlander, Jr., “Low power parallel multipliers,” in *Workshop VLSI Signal Process., IX*, 1996, pp. 199–208.

[18] A. A. Farooqui and V. G. Oklobdzija, “General data-path organization of a MAC unit for VLSI implementation of DSP processors,” in *Proc. IEEE Int. Symp. Circuits Syst.*, 1998, vol. 2, pp. 260–263.

[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: MIT Press, 1990.

[20] Z. Hong and R. Sedgewick, “Notes on merging networks,” in *Proc. ACM Symp. Theory Comput.*, 1982, pp. 296–302.

[21] L. Dadda, “Some schemes for parallel multipliers,” *Alta Frequenza*, vol. 34, pp. 349–359, 1965.

[22] K. C. Bickerstaff, E. E. Swartzlander, Jr., and M. J. Schulte, “Analysis of column compression multipliers,” in *Proc. 15th IEEE Symp. Computer Arithmetic*, 2001, pp. 33–39.

[23] CHIP Implementation Center, CIC, Taiwan, CIC Referenced Flow for Cell-Based IC Design, Document no. CIC-DSD-RD-08-01, 2008.

[24] K. Andra, C. Chakrabarti, and T. Acharya, “A VLSI architecture for lifting-based forward and inverse wavelet transform,” *IEEE Trans. Signal Process.*, vol. 50, no. 4, pp. 966–977, April 2002.

[25] A. Garimella, M. V. V. Satyanarayana, P. S. Murugesh, and U. C. Niranjan, “ASIC for digital color image watermaking,” in *Proc. IEEE Signal Processing Education Workshop*, 2004, pp. 292–296.

Jiun-Ping Wang received the B.S. degree in information and computer engineering from Chung-Yuan Christian University, Chung-Li, Taiwan, in 2003. He is currently working toward the Ph.D. degree at National Sun Yat-Sen University, Kaohsiung, Taiwan

His research interests include VLSI design, computer arithmetic, and low-power design.

Shiann-Rong Kuang (M’09) received the B.S. degree from National Central University, Taiwan, in 1990, and the M.S. and Ph.D. degrees from National Cheng Kung University, Taiwan, in 1992 and 1998, respectively, all in electrical engineering.

He is currently an Assistant Professor in the Department of Computer Science and Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan. His current research interests include VLSI/CAD and low-power SoC design.

Shish-Chang Liang received the M.S. degree in computer science and engineering from National Sun Yat-Sen University, Kaohsiung, Taiwan, in 2007.

His research interests include low-power arithmetic and VLSI design.