Fixed-Width Group CSD Multiplier Design

Yong-Eun KIM†, Kyung-Ju CHO††(a), Nonmembers, Jin-Gyun CHUNG†††, Member, and Xinming HUANG††††, Nonmember

SUMMARY This paper presents an error compensation method for fixed-width group canonic signed digit (GCSD) multipliers that receive a $W$-bit input and generate a $W$-bit product. To efficiently compensate for the truncation error, the encoded signals from the GCSD multiplier are used for the generation of the error compensation bias. By Synopsys simulations, it is shown that the proposed method leads to up to 84% reduction in the number of partial products, the group canonic signed digit (CSD) encoding [2] and partial product sharing algorithm. This multiplier provides efficient design when the multiplications are performed only with a few predetermined coefficients which are time-varying in periodical order. In these applications, multipliers should have programmability. When a few coefficients share a multiplier, modified Booth encoding, which halves the number of partial products, is generally used. To further reduce the number of partial products, the group canonic signed digit (GCSD) multiplier in [1] was recently proposed based on the variation of canonic signed digit (CSD) encoding [2] and partial product sharing algorithm. This multiplier provides an efficient design when the multiplications are performed only with a few predetermined coefficients (e.g., FFT).

In many multimedia and digital signal processing (DSP) applications, it is desirable to maintain fixed-width property through multiplication operations to avoid quick growth in word size. For example, the $(2W-1)$-bit product obtained from two $W$-bit operands is quantized to $W$-bits by eliminating the $(W-1)$-least-significant bits (LSBs). In practice, fixed-width multipliers can be designed based on Baugh-Wooley, modified Booth and CSD algorithms. In typical fixed-width multipliers, the adder cells required for the computation of the $(W-1)$-LSBs are eliminated and error compensation biases are introduced to the retained adder cells.

In order to reduce the truncation error, various error compensation methods for fixed-width multipliers have been proposed [3]–[9]. Error compensation biases of these methods can be classified into constant biases [3] and adaptive biases [4]–[9]. While a constant bias is generated independent of the truncated partial product bits and is fixed for a given input word size, an adaptive bias is determined depending on each input. Thus, adaptive bias methods achieve better computation accuracy than the constant bias method.

In this paper, we propose an error compensation method for low-error fixed-width GCSD multiplier. To efficiently compensate for the truncation error with reduced hardware complexity, the encoded signals from the GCSD multiplier are used for the generation of error compensation bias.

This paper is organized as follows. After a brief review of the GCSD multiplier in Sect. 2, we propose an error compensation bias design method for GCSD multipliers in Sect. 3. In Sect. 4, two application examples of the proposed fixed-width multiplier design method are presented. Finally, short statements conclude this paper.

2. GCSD Multiplier

Figure 1 shows the $N$-point radix-$2^4$ single-path delay feedback (SDF) FFT architecture [10]. In the first and the third multiplication blocks, three coefficients $\{\cos(\pi/8), \cos(\pi/4), \text{and} \sin(\pi/8)\}$ are multiplied by an complex input signal in periodical order as

$$
\begin{bmatrix}
y_0 \\
y_1 \\
y_2 \\
y_3
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
\cos(\pi/8) - j\sin(\pi/8) \\
\cos(\pi/4) - j\sin(\pi/4) \\
\sin(\pi/8) - j\cos(\pi/8)
\end{bmatrix}
\begin{bmatrix}
x_r + jx_i
\end{bmatrix}.
$$

(1)

In general, the multiplications in (1) can be implemented using a programmable multiplier such as the modified Booth multiplier. If the coefficient word-length is $W$, the number of the partial products obtained by the modified Booth algorithm is $W/2$.

To further reduce the number of partial products, the following coefficient grouping algorithm can be used [1]:

Manuscript revised December 2, 2009.

The author is with Vehicle-IT Fusion Research Center, Korea
Automotive Technology Institute, 74 Yongjeong-Ri, Pungse-

The author is with Division of Electronic Eng., and Informa-
tion and Communication Research Center, Chonbuk National
University, Deokjindong 1-ga Deokjin-gu, Jeonju, 561–756, Korea.

The author is with the Dept. of Electrical & Computer Engr.,
Worcester Polytechnic Institute, Worcester, MA, USA.

a) E-mail: kjcho@chonbuk.ac.kr
DOI: 10.1587/transinf.E93.D.1497
1. If the number of given 2’s complement coefficients is $N_c$ with the word length of $N_w$ bits, the coefficients are arranged as an $N_c \times N_w$ table.

2. The coefficients in the table are converted to CSD encoded representations.

3. Starting from the first column, a group is defined such that each row in a group contains at most one nonzero digit. A group should contain as many columns as possible so that the number of groups is minimized.

By applying the grouping algorithm to the three coefficients in (1) with $N_w = 14$, the CSD coefficient table with 5 groups is obtained as shown in Table 1. Each group $G_i$ generates a corresponding partial product $P_i$. Thus, the multiplication result ($Y$) is obtained as

$$Y = P_4 + 2^{-2}P_3 + 2^{-4}P_2 + 2^{-6}P_1 + 2^{-8}P_0.$$  (2)

In Table 1, the number of partial products required by the grouping algorithm is only 5, while the modified Booth encoding requires 7 ($= N_w/2$) partial products. Thus the grouping algorithm reduces the number of partial products by 2, which can lead to lower power consumption and higher speed. Each group includes at least two columns by the grouping algorithm since CSD coding does not allow any consecutive nonzero digits. Thus, the number of partial products generated by the grouping algorithm is always less than or equal to that of the modified Booth encoding.

If the nonzero digit locations of two groups are the same as in $G_4$ and $G_1$ in Table 1, the two groups can share PP generation circuits. The sign difference in the first rows of $G_4$ and $G_1$ can be taken care of later by additional complementing circuits. For any row in a group that contains only 0’s, the corresponding PP is 0. In this case, the zero digits in the row can be changed to nonzero digits to share the partial product generation circuits, since the output value can be easily changed back to 0 using a control signal. By the partial product sharing algorithm in [1], a new representation of each group in Table 1 can be obtained using control signals as shown in Table 2, where $S_i$, $N_i$, and $Z_i$ are shift, negation, and zero control signals, respectively.

In conventional approach, the coefficient look-up table (LUT) has 14 columns if the coefficient word-length is 14. In GCSD multipliers, encoded values are stored instead of binary coefficients. In [11], a look-up table (LUT) reduction method for modified Booth encoded coefficients was proposed. By similar approach, the number of columns of Table 2 can be reduced to 7. Thus, in this case, LUT size is reduced by 50% compared with conventional approaches.
lay by 41%, 45% and 12%, respectively, compared with the conventional modified Booth multiplier [1].

3. Fixed-Width GCSD Multiplier

For the coefficient Table 1 with $N_w = 14$, the corresponding partial product array for the GCSD multiplier can be obtained as shown in Fig. 2. The partial product array can be divided into $MP$ and $LP$ as shown in Fig. 2, where $MP$ and $LP$ mean more significant and less significant parts, respectively. Then, if $S_{MP}$ and $S_{LP}$ represent the sums of the elements in $MP$ and $LP$, respectively, we can express $(2W - 1)$-bit ideal product $P_t$ as

$$P_t = S_{MP} + S_{LP}. \quad (3)$$

In typical fixed-width multipliers, the adder cells required for $S_{LP}$ are omitted and appropriate biases are introduced to the retained adder cells based on a probabilistic estimation. Thus, the $W$-bit quantized product $P_Q$ can be expressed as

$$P_Q = S_{MP} + \sigma \times 2^{-(W-1)}, \quad (4)$$

where $\sigma$ means the error compensation bias. Note that $\sigma$ approximates the carry signals propagated from $LP$ to $MP$.

To generate error compensation bias more efficiently, the truncated bits in $LP$ can be further divided into $LP_{\text{major}}$ and $LP_{\text{minor}}$ depending on their effects on the truncation error. Then, $S_{LP}$ can be expressed as

$$S_{LP} = S_{LP_{\text{major}}} + S_{LP_{\text{minor}}}. \quad (5)$$

As an example, in Fig. 2, $S_{LP_{\text{major}}}$ and $S_{LP_{\text{minor}}}$ can be expressed as

$$S_{LP_{\text{major}}} = p_{0,12} + p_{1,6} + p_{2,4} + p_{3,2} + p_{4,0},$$

$$S_{LP_{\text{minor}}} = 2^{-1}(p_{0,11} + p_{1,5} + p_{2,3} + p_{3,1}) + 2^{-2}(p_{10,0} + p_{14,4} + p_{12,2} + p_{13,0} + N_i Z_5) + \cdots + 2^{-12}(p_{0,0} + N_0), \quad (6)$$

where $p_{i,j}$ means the $j$th partial product bit of partial product $P_t$.

Theoretically, the most accurate error compensation bias can be obtained by true rounding as

$$\sigma_{\text{true}} = \left[ \frac{S_{LP}}{2^r} \right], \quad (7)$$

where $[t]$ denotes the rounding operation of $t$.

Let $2^{-L_i}$ be the weight of the LSB of the partial product $P_i$. Also, let $M_i$ be the required number of shift-left bit positions for the partial product $P_i$. As an example, for $G_0$ in Table 1, coefficient $\sin(\pi/8)$ has a nonzero digit at the last column of $G_0$. Thus $M_0$ is defined to be 0 for $\sin(\pi/8)$ since no shift operation is required in this case. On the other hand, $\cos(\pi/8)$ has a nonzero digit at the second column of $G_0$. Thus for $\cos(\pi/8)$, $M_0$ is defined to be 4 since shift-left by 4 bit positions are required in this case. Using Table 1, Table 2 and Fig. 2, the possible values of $LP_{\text{minor}}(P_i)$ can be obtained depending on the control signals as shown in Fig. 3. If $N_i = 1$, the input signals are negated as can be seen in Fig. 3. When $Z_i = 0$, partial product bits are changed to 0.

Assume that each bit $x_i$ of input $X$ has the uniform probability distribution. Then, the expected value of $x_i$ is

$$E[x_i] = 1/2. \quad (8)$$

Then, it can be shown that the expected value of $S_{LP_{\text{minor}}}(P_i)$ can be computed as

$$E[S_{LP_{\text{minor}}}(P_i)] = \begin{cases} 0, & Z_i = 0 \\ 2^{-1}(1 - 2^{-f(L_i - M_i)}), & Z_i \neq 0, N_i = 0 \\ 2^{-1}(1 + 2^{-f(L_i - M_i)}), & Z_i \neq 0, N_i \neq 0, \end{cases} \quad (9)$$

where

$$f(L_i - M_i) = \begin{cases} L_i - M_i, & \text{for } L_i > M_i \\ 0, & \text{for } L_i \leq M_i. \end{cases} \quad (10)$$

The error compensation bias of fixed-width modified Booth multiplier in [8] is defined as follows:

$$\sigma_{\text{[8]}} = C_E[S_{LP_{\text{major}}} + C_A[S_{LP_{\text{minor}}}], \quad (11)$$

where $C_E[t]$ and $C_A[t]$ represent the carry value and the approximate carry value of $t$, respectively. In (11), $C_A[S_{LP_{\text{minor}}}]$ is the approximate carry value $a_{\text{carry}}$ propagated from $LP_{\text{minor}}$ to $LP_{\text{major}}$. In [8], the expect value of $S_{LP_{\text{minor}}}(P_i)$ is identified as

$$E[S_{LP_{\text{minor}}}(P_i)]_{[8]} = \begin{cases} 0, & y''_{\text{[8]}} = 0 \\ 2^{-1}, & y''_{\text{[8]}} = 1. \end{cases} \quad (12)$$

where $y''_{\text{[8]}}$ corresponds to $Z_i$ signal in this paper. The approximate carry signal in [8] is defined as the rounded value of

![Fig. 2](image-url)
of the truncated bits in \( P_i \).

Let the partial products included in \( L_{P_{\text{minor}}} \) be \( P_K, P_{K-1}, \ldots, P_0 \). Since the LSB of \( P_i \) always has a smaller weight than that of \( P_{i-1} \) under any coefficient conditions, the following relation holds:

\[
0 \leq f(L_K - M_K) < f(L_{K-1} - M_{K-1}) < \cdots < f(L_0 - M_0).
\]

Thus, using (15) and CSD property, the maximum value of \( E[S_\cdot L_{P_{\text{minor}}}] \) can be obtained as

\[
\max\{E[S_\cdot L_{P_{\text{minor}}}]\} = 2^{-1} \times \max\left[ 1 + 2^{-f(L_K - M_K)} \right] + \left( 1 + 2^{-f(L_{K-1} - M_{K-1})} \right) + \cdots + (1 + 2^{-f(L_0 - M_0)})
\]

\[
= 2^{-1} \times (K + 1 + 2^0 + 2^{-2} + 2^{-4} + \cdots + 2^{-2K})
\]

\[
= 2^{-1} \times (K + 2 + 2^{-2} + \cdots + 2^{-2K}).
\]

The carry signals generated from \( L_{P_{\text{minor}}} \) are determined by the integer part of the terms inside the parenthesis in (16). Thus, \( 2^{-f(L_K - M_K)} \) can have an effect on the carry signals generated from \( L_{P_{\text{minor}}} \) to \( L_{P_{\text{major}}} \) but \( \sum_{i=0}^{K-1} 2^{-f(L_i - M_i)} \) has no effect on the carry signals.
In addition, for the partial product \( P_K \), when \( M_K \) is larger than or equal to \( L_K \), the partial product bits inside the \( LP_{\text{minor}} \) are filled with 0’s. Thus, when the partial product \( P_K \) is negated (i.e., \( N_K = 1 \)), a carry signal is propagated to \( LP_{\text{major}} \).

Based on these observations, \( E[S_{\text{LP}_{\text{minor}}}] \) can be easily computed as

\[
E[S_{\text{LP}_{\text{minor}}}] = 2^{-1} \times \{g(L_K - M_K) \cdot 2N_K + g(L_K - M_K) \cdot Z_K \cdot N_K + Z_{K-1} + \cdots + Z_0\},
\]

where

\[
g(L_K - M_K) = \begin{cases} 1, & \text{for } L_K \leq M_K \\ 0, & \text{for } L_K > M_K. \end{cases}
\]

In this paper, the rounded value of \( E[S_{\text{LP}_{\text{minor}}}] \) is defined as the approximate carry value propagated from \( LP_{\text{minor}} \) to \( LP_{\text{major}} \). As an example, from Fig. 3 (d), it can be seen that \( L_3 (= 2) \) is larger than \( M_3 (= 0 \text{ or } 1) \). Thus, by (17), the approximate carry value is decided as

\[
a_{\text{carry}} = \left[ \frac{1}{2} (Z_3 + Z_2 + Z_1 + Z_0) \right],
\]

where \([x]\) defines the smallest integer greater than or equal to \( x \). In general, (19) can be implemented as shown in Fig. 4 (a). However, since the number of nonzero \( Z_i \) signals in Table 2 is either 3 or 4, the values of \( a_{\text{carry}} \) signals for the three coefficients are always 2. Thus, in this case, no additional hardware is required for the generation of the approximate carry signals as shown in Fig. 4 (b).

The proposed error compensation bias is computed using \( S_{\text{LP}_{\text{major}}} \) and \( a_{\text{carry}} \). When the number of nonzero \( Z_i \) signals is odd, the effect of rounding error can be large in the computation of \( a_{\text{carry}} \) signal. To alleviate this problem, we propose an error compensation bias for fixed-width GCSD multipliers as follows:

If \( N_{\text{NZPP}} \) = odd, then

\[
\sigma_{\text{prop}} = \left[ \frac{S_{\text{LP}_{\text{major}}} + a_{\text{carry}}}{2} \right],
\]

else

\[
\sigma_{\text{prop}} = \left[ \frac{S_{\text{LP}_{\text{major}}} + a_{\text{carry}}}{2} \right],
\]

where \( N_{\text{NZPP}} \) is the number of nonzero \( Z_i \) signals.

In GCSD multipliers, the number of the different coefficients in a group is assumed to be small. Thus, the coefficient selection signals (or, address) need only a few bits. Using this property, the approximate carry signals can be designed using the address bits. For example, if the word length \( N_w \) of the coefficients in Table 1 is 12, the approximate carry value can be obtained as follows:

\[
a_{\text{carry}} = \begin{cases} 2, & \text{for } \cos(\pi/8), \ a_1a_0 = 00, \\ 2, & \text{for } \cos(\pi/4), \ a_1a_0 = 01, \\ 1, & \text{for } \sin(\pi/8), \ a_1a_0 = 10, \end{cases}
\]

where \( a_1a_0 \) means the address for the coefficients. Since the maximum value of \( a_{\text{carry}} \) is 2 in (21), \( a_{\text{carry}} \) can be represented as

\[
a_{\text{carry}} = 2a_{\text{carry}}_1 + a_{\text{carry}}_0. \tag{22}
\]

From (21), the following expression can be obtained using Karnaugh map:

\[
a_{\text{carry}}_1 = a_1, \\quad a_{\text{carry}}_0 = a_1a_0. \tag{23}
\]

In general, the implementation using address bits requires smaller area compared with the implementation based on coefficient control signals when the number of coefficients is small.

4. Performance Comparisons

To evaluate the performance of the proposed fixed-width GCSD multiplier, we compute the maximum absolute error \( e_{\text{max}} \), the average of absolute error \( e_{\text{avg}} \) and the variance of error \( e_{\text{var}} \) for all the possible \( 2^W \) input values of \( X \) as follows

\[
\begin{align*}
\varepsilon_{\text{abs}} &= |P_I - P_Q|, \\
\varepsilon_{\text{max}} &= \max(|\varepsilon_{\text{abs}}|), \\
\varepsilon_{\text{avg}} &= 2^{-W} \sum \varepsilon_{\text{abs}}, \\
\varepsilon_{\text{var}} &= 2^{-W} \sum (\varepsilon - \varepsilon_{\text{avg}})^2.
\end{align*}
\]

Table 3 shows the simulation results of the fixed-width GCSD multiplier for the input word size \( W = 14 \). Let \( M_{\text{true}} \) and \( M_{\text{post}} \) denote the fixed-width multiplier by the true rounding and post-truncation, respectively. For the computation of \( M_{\text{true}} \) and \( M_{\text{post}} \), all the possible bits are required during the multiplication and the final product is obtained by rounding or truncating the least significant \((W - 1)\)-bits from the exact \((2W - 1)\)-bit result. Also, Table 4 compares the Synopsys simulation results using MagnaChip 0.18-μm CMOS technology. Notice that, compared with the fixed-width modified Booth multiplier, the proposed GCSD fixed-width multiplier can reduce about 10% average error. In addition, the proposed multiplier leads to 29%, 36% and 9% reduction in area, power consumption and propagation delay, respectively.
As another example, the proposed algorithm is applied to the following coefficients used in the pulse-shaping filter design for CDMA [11]:

\[ \begin{align*}
a_1a_0(00): & \quad 1111111010, \\
a_1a_0(01): & \quad 1111111000, \\
a_1a_0(10): & \quad 1111111111, \\
a_1a_0(11): & \quad 1111111100. 
\end{align*} \] (25)

Figure 5 shows the partial product array corresponding to the GCSD multiplier. Table 5 compares the error performances of several methods and Table 6 compares the Synopsys simulation results. In this case, although the error performances of the proposed method and the method in [8]

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Comparison of the error performances for FFT applications.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \varepsilon_{\text{max}} \times 2^{(W-1)} )</td>
</tr>
<tr>
<td>( M_{\text{cue}} )</td>
<td>0.5 (1)</td>
</tr>
<tr>
<td>( M_{\text{post}} )</td>
<td>0.9999 (2)</td>
</tr>
<tr>
<td>( M_{[8]} )</td>
<td>2.6328 (5.27)</td>
</tr>
<tr>
<td>( M_{\text{prop}} )</td>
<td>1.2070 (2.41)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4</th>
<th>Synopsys simulation results for FFT applications.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Area (cell)</td>
</tr>
<tr>
<td>([8]_{\text{fixed}} )</td>
<td>8159 (1)</td>
</tr>
<tr>
<td>GCSD</td>
<td>7577 (0.93)</td>
</tr>
<tr>
<td>GCSD_{\text{fixed}}</td>
<td>5779 (0.71)</td>
</tr>
</tbody>
</table>

| Fig. 5 | Partial product array of the GCSD multiplier for pulse-shaping filters. |

<table>
<thead>
<tr>
<th>Table 5</th>
<th>Comparison of the error performances for pulse-shaping filters.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \varepsilon_{\text{max}} \times 2^{(W-1)} )</td>
</tr>
<tr>
<td>( M_{\text{cue}} )</td>
<td>0.5 (1)</td>
</tr>
<tr>
<td>( M_{\text{post}} )</td>
<td>0.9980 (2)</td>
</tr>
<tr>
<td>( M_{[8]} )</td>
<td>2.2188 (4.44)</td>
</tr>
<tr>
<td>( M_{\text{prop}} )</td>
<td>1 (2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 6</th>
<th>Synopsys simulation results for pulse-shaping filters.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Area (cell)</td>
</tr>
<tr>
<td>([8]_{\text{fixed}} )</td>
<td>6116 (1)</td>
</tr>
<tr>
<td>GCSD</td>
<td>2343 (0.38)</td>
</tr>
<tr>
<td>GCSD_{\text{fixed}}</td>
<td>1338 (0.22)</td>
</tr>
</tbody>
</table>
are almost the same, the proposed multiplier leads to 78%, 84% and 53% reduction in area, power consumption and propagation delay, respectively.

5. Conclusions

In this paper, an efficient error compensation method for fixed-width GCSD multipliers is proposed. To compute the error compensation bias more accurately, the encoded signals from the GCSD multiplier are used for the bias generation.

The simulation results show that the proposed method leads to significant reduction in area, power consumption, and delay time compared with the fixed-width modified Booth multipliers.

References