ERROR COMPENSATED FIXED WIDTH MODIFIED BOOTH MULTIPLIER FOR MULTIMEDIA APPLICATIONS

R. MOHAMED NIYAS, 1 N. PRABHAKARAN, 2 N. SATHYA, 3 P. VAISHNAVI, 4 B. MADHUSUDHANA REDDY

1, 2, 4, 5 Assistant Professor, Department of Electronics & Communication Engineering, Vel Tech High Tech Dr. Rangarajan & Dr. Sakuntala Engineering College

3 Assistant Professor, Department of Electronics & Communication Engineering, Vel Tech (Owned By RS Trust)

E-mail: 1 mdnys@yahoo.com, 2 captainprabhakar1982@yahoo.co.in, 3 sathya nalayiram@gmail.com, 4 vaishnavi ec07@gmail.com, 5 madhu.barusu@gmail.com

ABSTRACT

Many multimedia and digital signal processing systems are desirable to maintain a fixed format and to allow little accuracy loss to output data. The objective of this paper is to design a fixed width modified booth multiplier with high error performance. And the need to derive an effective error compensation function that makes the error distribution more symmetric and centralized in the error equalized to zero. The compensation circuit is mainly composed of simplified sorting network and this can achieve a tiny mean and mean square error as compared to the other circuits. The odd even sorting networks used for error compensation are composed of appropriately connected comparators. The simplified form of sorting network consist of neither NAND, NOR, AND-OR INVERTER (AOI) and OR-AND-INVERTER (OAI).

In fixed width modified Booth multiplication, to reduce the number of partial products by a factor of two, 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. The n * n fixed width multipliers that generate only the n most significant product bits are utilized to maintain the fixed word size which allows a little accuracy loss to output data. By directly removing the adder cells the hardware complexity reduction and power saving can be achieved and computation of the n least significant bits of 2n –bit output product is also done. But in direct –truncated fixed width multiplier (DTFM) a huge truncation error will be introduced.

Various error compensation techniques are used to effectively reduce the truncation error, which add estimated compensation value to the carry input of the reserved adder cells, have been proposed. Two methods, constant scheme and adaptive scheme are used to produce the error compensation value. The truncation error of constant scheme is very large since it pre-computes the error compensation value and feeds them to the carry input of retained adder cells regardless of the current input data value. But in adaptive scheme [5] high accuracy is achieved by adaptively adjusting the compensation value according to the input data with little higher hardware complexity. However adaptive error compensation scheme are not applied for fixed width modified Booth multipliers directly since it is developed for fixed width array multipliers. Hence several error compensation approaches [6] have been proposed to reduce the truncation error of fixed width modified Booth multipliers. By using statistical analysis and linear regression analysis the compensation value generated significantly reduces the mean error but the maximum absolute error and the mean square error are still large.

1. INTRODUCTION

Multipliers are always the fundamental arithmetic unit of all the multimedia applications and DSP systems for which high processing performance and low power dissipation are the most important objective. 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. The n * n fixed width multipliers that generate only the n most significant product bits are utilized to maintain the fixed word size which allows a little accuracy loss to output data. By directly removing the adder cells the hardware complexity reduction and power saving can be achieved and computation of the n least significant bits of 2n –bit output product is also done. But in direct –truncated fixed width multiplier (DTFM) a huge truncation error will be introduced.
The truncated parts of the product bit matrix of booth multiplication are divided into major group and a minor group depending on their effects on the truncation error. Booth encoded outputs are consumed to make the error compensation value. The fixed width modified Booth multiplier in [7] attains improved performance in terms of greatest absolute error and the mean square error. The smaller mean and mean square error represent that the error distribution is more symmetric and centralized in the error equal to zero. In multimedia and DSP applications the final output are produced from accumulating a series of product rather than from a single multiplication operation hence truncation error is accumulated to produce a huge output error. To avoid this compensation value is computed which requires additional computations, the fixed-width multiplier with small mean and mean square error is largely expected to obtain more accurate output data.

In this paper high error compensated circuit for fixed width modified Booth is proposed. Therefore, the mean and mean square error can be reduced, hence the resultant fixed width multiplier is suitable for different applications. To achieve this goal, at first the partial product matrix of Booth multiplication is improved to decrease the partial product bits in the truncated portion of DTFM. Then the correlation among the Booth encoded outputs and the truncated product error of DTFM is examined and explored to derive an effective error compensation function, which can generate an approximation to the carry value generated by truncated portion of DTFM, to decrease the truncation error and make the error distribution as symmetric and centralized as possible. Finally a simple compensation circuit composed of 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 multipliers achieves much higher accuracy.

2. BOOTH MULTIPLIER

2.1 Modified Booth Multiplier

Consider the multiplication of two n-bit signed number

\[ A = -a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i \]  (1)

\[ B = -b_{n-1}2^{n-1} + \sum_{i=0}^{n-2} b_i 2^i \]  (2)

Table I
Modified Booth Encoding Table

<table>
<thead>
<tr>
<th>(b_{2i+1})</th>
<th>(b_2)</th>
<th>(b_{2i})</th>
<th>Operation</th>
<th>neg</th>
<th>one</th>
<th>zero</th>
<th>cor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>+0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>+1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>-2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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

\[ B = \sum_{i=0}^{n/2-1} M_i 2^i = \sum_{i=0}^{n/2-1} (-2b_{2i} + b_2 + b_{2i+1}) 2^i \]  (3)

Where \(b_{-1} = 0\) and \(M_i \in \{-2,-1,0,1,2\}\).

Based on the encoded results shown in table 1 the Booth encoder and the partial product generation circuit are adopted to choose one of multiple multiplicand -2A, -A, 0, A and 2A for generating each partial product row PPi where 0 ≤ i ≤ n/2-1 and aj bar is the complement of aj (0 ≤ j ≤ n-1). In Table I. Left shifting A by one bit achieves 2A. For negation, each bit of A is inverted and an extra binary value ‘1’ is added to the LSB of next partial product row. Adding ‘1’ can be considered as a correlation bit Cori which indicates that partial product is PPi is positive if cori=0 or negative if cori=1. Further, the sign bit for partial product row must be extended upto (2n-1) bit position because each PPi row is represented in two’s complementation.

Let us consider two 8 bit signed integers A=10101010; B=11100001. The two’s complement of A is a=01010110, the two’s complement of B is b=00011111. Let this b can be given as input to the booth encoder. Outputs of booth encoder are neg=0011, one=0001, two=0100, zero=1010, Cori=0010, the width of all these outputs is i=4 (0≤i≤3); where as i=n/2-1.
The two’s complement of \( A \) i.e. \( a \) and \( \overline{a} \) are given as inputs to partial product generation circuit. \( a = 01010110; \overline{a} = 10101001 \); here four registers are used namely, \( q, r, s, \) and \( t \), to store the partial products generated from the circuit. No. Of partial products \( i = 4 \).
2.3 Partial Product Matrix For 8×8 Modified Booth Multiplication

These additional sign bits affect the performance and power consumption of the multipliers. Hence Fig. 5 illustrates the partial product matrix if 8×8 modified Booth multiplier with sign extension elimination technique to solve this problem. Here the partial product matrix can be divided into MP and LP, where LP is of two types namely, LP major and LP minor. According to DTFM, LP can be directly omitted but huge truncation error will be introduced. Hence in the propagated error compensated multiplier, only the partial product bits in LP minor are removed and carry value is propagated from LP minor to LP major, which is estimated by a simple circuit.

3. PROPOSED FIXED WIDTH MODIFIED BOOTH MULTIPLIER

3.1 Generation Of Error Compensation Function

Let sum (MP) and sum (LP) represent the sum of partial product bits in MP and LP then the output product P can be expressed as

\[ P = A \times B = \text{SUM(MP)} + \text{SUM(LP)} \]  

The sum (LP) can be obtained as

\[ \text{SUM(LP)} = \delta \times 2^n \]  

Where

\[ \delta = \frac{1}{2}\left[p_{0,n-1} + \cdots + p_{n/2-1,1}\right] + \frac{1}{2}\left[p_{0,n-2} + \cdots + p_{n/2-1,0}\right] + \frac{1}{2}\left[p_{0,n-1} + \text{COR}_{0,n-1}\right] \]

\[ + \cdots + \frac{1}{2}\left[p_{0,0} + \text{COR}_{0,0}\right] \]

Let \( \theta_{\text{major}} = p_{0,n-1} + \cdots + p_{n/2-1,1} \)

Can be written as

\[ \delta = \frac{1}{2}\theta_{\text{major}} + \theta_{\text{minor}} \]

Where

\[ \theta_{\text{minor}} = \frac{1}{2}\left[p_{0,n-2} + \cdots + \text{COR}_{0,n-2}\right] + \frac{1}{2}\left[p_{0,0} + \text{COR}_{0,0}\right] \]

If partial products in LP are removed the most accurate error compensation value \( \delta \) is the rounding value of \( \delta \) that can be expressed as

\[ \delta = \frac{1}{2}\left[\theta_{\text{major}} + \theta_{\text{minor}}\right] = \left[\frac{1}{2}(\theta_{\text{major}} + 1) + \theta_{\text{minor}}\right] \]

3.2 Post Truncated Modified Booth Multiplier

Post-truncated modified Booth multiplier (PTM) can generate the most accurate error compensation value \( \hat{\delta} \). By adding an extra ‘1’ at the (n-1) th bit position of partial product matrix shown in fig 2 PTM can be achieved. Inspite of the advantages this results in hardware complexity and power consumption to standard modified Booth multiplier. Hence, a simple error compensation function whose output value approximates the most accurate error compensation value \( \hat{\delta} \) is developed by the following steps. First, the partial product matrix of PTM shown in fig.2 is modified into a new matrix as shown in Figure 3.
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 carry \( \lambda \) at the \((n-2)\)th and \((n-1)\)th bit positions respectively. The expression for \( \varepsilon_{n/2-1,0} \) and \( \lambda \) can be derived from the truth table shown in Table II as follows

\[
\varepsilon_{n/2-1,0} = a_0 \land one_{n/2-1} \quad (11)
\]

\[
\lambda = (a_0 \land one_{n/2-1}) \land zero_{n/2-1} \land b_{n-1} \quad (12)
\]

The expressions for carry \( \overline{\lambda} \) and \( \omega_2, \omega_1, \omega_0 \) can be derived as

\[
\overline{\lambda} = ((a_0 \land one_{n/2-1}) \land zero_{n/2-1} \land b_{n-1}) \quad (13)
\]

\[
\omega_2 = (s_0 \land \overline{\lambda}) \quad (14)
\]

\[
\omega_1 = \omega_2 \quad (15)
\]

\[
\omega_0 = (s_0 \lor \overline{\lambda}) \land \omega_2 \quad (16)
\]

Where \( \lor \) denotes the OR operation. Now generation of \( \overline{lbar} \) (shown as \( \overline{\lambda} \) in product matrix):

\[
\delta = \frac{1}{2} \left[ (\theta_{major} + \overline{\lambda}) + CA(LP') \right] \quad (17)
\]

\[
= \frac{1}{2} \left[ \theta'_{major} + CA(LP'_{min\lor}) \right] \quad (18)
\]
where $\theta'_\text{major} = \theta_\text{major} + \lambda$ represents the approximate carry value propagated from $L'P_{\text{minor}}$ to $L'P_{\text{major}}$. Sum ($L'P_{\text{minor}}$) is the sum of partial product bits in $L'P_{\text{minor}}$; $\sim C$ is substituted to obtain the expression as below

$$S(X) = \frac{1}{N(T_{AB}(X))} \sum_{(A,B) \in T_{AB}(X)} \text{SUM}(L'P_{\text{minor}}) \quad (19)$$

where $N(T_{AB}(X)) = N(T_{B}(X)) \times 2^n$ and $N(T_{B}(X))$ indicate the number of elements in $T_{B}(X)$. As $X = \bar{z}_n 2^{n-1} + z_{n-2} 2^{n-2} + \ldots + z_0 2^0$. $S(X) = \text{floor function}(\sim S(X)/2^{n-1}) = \text{sum of } (\bar{z}_0, zbar[1], zbar[2], zbar[3])$ Finally estimated carry = floor function ($\theta'_\text{major} + S(X)$).

3.3 Proposed Low Error Compensation Circuit

The SC-generator is the simple and efficient circuit for low error compensation. The input of SC generator is $\bar{z}_i$ for $0 \leq i \leq \frac{n}{2}-1$ and it will generate $m$ output bits $\alpha_1, \alpha_2, \ldots, \alpha_m$.

```
\begin{array}{cccccccc}
<table>
<thead>
<tr>
<th>n</th>
<th>n \times 8</th>
<th>n \times 16</th>
<th>n = 1.2</th>
<th>n = 1.4</th>
<th>n = 16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>-</td>
<td>-</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
\end{array}
```

Figure 8: Block Diagram for Proposed Error Compensation Circuit.

SC-generator consists of odd even sorting network. After the estimation of carriers they are fed into $L'P_{\text{minor}}$ to produce the final fixed-width product. SC-generator works with the principle of odd even merge sorting algorithm. The network is given below

```
\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure8.png}
\caption{Block Diagram for Proposed Error Compensation Circuit.}
\end{figure}
```

```
\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure9.png}
\caption{RTL view of Odd-Even Merge Sorting Network.}
\end{figure}
```

```
\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure10.png}
\caption{Simulation Waveform for Odd-Even Merge Sorting Network.}
\end{figure}
```

The inputs of SC generator are $\sim \bar{z}_n = \bar{z}_n$, $\bar{z}_n = 0101$, Here the larger weights are given MSB positions; in such a way they are sorted using the above circuit. The output is $P = 0011$; to choose alpha value from $p[0]$, $p[1]$, $p[2]$, $p[3]$, the formula used is $m = \text{floor function of } ((n/2-1)/2)$, $m = \text{floor function } (1.5)=1$, since $n=8$; $\alpha = p[m] = p[1] = 1$;

```
\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
Messages & Input & Output \\
\hline
$\bar{z}_0$ & 0001 & 0000  \\
$\bar{z}_0$ & 0011 & 0010  \\
$\bar{z}_0$ & 0101 & 0100  \\
$\bar{z}_0$ & 0111 & 0110  \\
$\bar{z}_0$ & 1001 & 1000  \\
$\bar{z}_0$ & 1011 & 1010  \\
$\bar{z}_0$ & 1101 & 1100  \\
\hline
\end{tabular}
\caption{Simulation Waveform for Odd-Even Merge Sorting Network.}
\end{table}
```

Now take four registers $F$, $G$, $H$, $I$ of 16-bit length, frame all the required bits in that registers as per below partial product matrix, then accumulation is done using accumulator. The result obtained are as follows $F=0000100010101100$; $G=0000110000000000$; $H=0110101001000000$; $I=1100000000000000$. 

```
Now finally add lbar and alpha to the SUM[6], then truncate SUM[15:0] to SUM[15:7]. Finally making the widths of SUM, lbar, alpha same by appending zeros so that it will be very easy to add. Now as a result SUM=000001010(9bits); Lbar=000000001; P[2]=alpha=000000000. Now again add all three variables to get a 9bit output out of which most MSB 8 bits as final output. The RTL view of Adding Sum, lbar, alpha using accumulator is shown below in Figure 13. The output is final(8bits)=11100110.

3.4 Power Analysis

The power analysis is done using Quartus simulator and optimum power is calculated. Power in a system is mainly composed of static and dynamic power. The dynamic power is due to the switching activity of the signals and the output capacitance in the circuitry. For n=8 the experimental results are as follows;
4. CONCLUSION

In this paper, error compensated fixed width modified Booth multiplier has been proposed. In the proposed paper, the partial product matrix of Booth multiplication was slightly modified and an effective error compensation function was derived. This compensation function makes the error low; leading the error compensated modified Booth multiplier with low error. In addition, a simplified sorting network has been designed to realize the compensation function.

**REFERENCES:**


