Available online at www.sciencedirect.com

ScienceDirect

Procedia Materials Science 10 (2015) 395 – 402

Procedia Materials Science

www.elsevier.com/locate/procedia

2nd International Conference on Nanomaterials and Technologies (CNT 2014)

Fixed-Width Multiplier with Simple Compensation Bias

Saroja S Bhusare[a,*] V S Kanchana Bhaaskaran[b]

[a]*JSS Academy of Technical Education, Bangalore, India.*

[a,b]*VIT University, Chennai, India.*

Abstract

Multiplications in many of the DSP applications are implemented by fixed-width multipliers primarily due to its low hardware complexity, less operation delay time and reduced power consumption. This paper presents an error compensation method for a fixed-width multiplier that receives two n-bit inputs and produces n-bit product. For the generation of error compensation bias, Booth encoder outputs have been employed. In order to compensate for truncation error and to generate the error compensation bias efficiently, truncated bits are divided into two groups and the carry estimation is done through exhaustive simulations. The simulation results reveal that the proposed method reduces the truncation error significantly compared with the direct-truncated multiplier with modest hardware overhead. Results further validate that the overall truncation error is significantly reduced as compared with the other existing method.

Peer-review under responsibility of the International Conference on Nanomaterials and Technologies (CNT 2014)

Keywords: **Fixed-width multiplier, Error-compensation bias, modified Booth multiplier, Truncation error.**

1. Introduction

In many DSP applications, truncation of (n-1) least-significant bits of the resultant (2n-1)-bit product due to the internal word length is essential, to reduce the hardware complexity, operation delay time and power consumption.

* Corresponding author. Tel.: +0-91-9964420351; *.E-mail address:* saroja_sush@yahoo.co.in.

Peer-review under responsibility of the International Conference on Nanomaterials and Technologies (CNT 2014)
doi:10.1016/j.mspro.2015.06.071

This can be done by removing half of the adder cells required for computing the (n-1) least-significant columns which in effect reduces the area by approximately 50%. But it leads to huge truncation error. In the post truncation method, after doing all the operations, (n-1) LSBs are truncated. This is the most accurate method, but it results in increased computation time and more hardware area. In order to address this problem, several fixed-width multiplier schemes have been proposed. In fixed-width multipliers, the adder cells required for the addition of (n-1) LSBs are removed and appropriate error compensation biases are obtained and added to the retained adder cells. In this manner, the truncation error can be significantly reduced.

Various approaches to reduce the hardware and to improve the accuracy of fixed-width multipliers have been proposed in literature. Sunder S Kidambi et al (1996) proposed a technique in which a constant bias is computed and added to the retained adder cells. This technique is advantageous in terms of bias generation. However it has huge truncation error since the bias obtained is constant and does not adapt to the input data. Adaptive schemes proposed by Jer Min Jou et al (1999), Lan-Da Van et al (2000) and Chih-Chyau Yang et al (2005) achieve higher accuracy while comparing to the constant scheme, since the compensation bias adjusts adaptively, according to the input data though at the cost of a minimum hardware overhead. Jer Min Jou et al (1999) have proposed Low-error fixed-width multipliers for sign-magnitude and two's- complement formats. Here the authors have presented a general methodology for designing a Low-error two's-complement fixed-width multiplier. Authors also have investigated on choosing the generalized index. Lan-Da Van and Chih-Chyau Yang (2005) have proposed a more generalized method to design Low-error fixed-width two's-complement multipliers for different cases. Several techniques for the reduction of the truncation error of fixed-width modified Booth multipliers have been proposed by Shyh-Jye Jou et al (2000), Meng_Hung Tsai et al (2003), Kyung-Ju Cho et al (2004) and Jiun-Ping Wang et al (2011). Shyh-Jye Jou et al (2000) have proposed a technique in which statistical and linear regression analysis is used to generate the compensation bias and the authors have extended the approach to modified Booth multiplier. Meng HungTsai et al (2003) have proposed a technique in which statistical and linear regression analysis is employed for generating the compensation bias. However the truncation errors are found to be still large. Kyung-Ju Cho et al (2004) have proposed an efficient method for fixed-width modified Booth multiplier. Here, the authors have used Booth encoder outputs for the generation of compensation bias. A technique which is highly accurate has been proposed by Jiun-Ping Wang et al (2011). In this proposal, an error compensation circuit, which could make the error distribution symmetric to and also centralize in zero error, for reducing both the mean error and mean-square errors simultaneously has been realized.

The modified Booth encoding technique has been employed in several multipliers, for the main reason that it reduces the number of partial products by a factor of 2. This paper presents a method for the generation of error compensation bias for fixed-width modified Booth multiplier. For efficient generation of compensation bias, Booth encoder outputs have been employed. Truncated bits are divided into the *major group and minor group*. Estimation of the carry is done through exhaustive simulation.

The paper is organized as follows. In Section II, a brief review of modified Booth encoding is presented. Section III presents the proposed error compensation technique. Section IV presents the results and performance comparison of the proposed method with the existing techniques. Section V concludes the paper.

2. **Fundamentals of Modified Booth encoding**

Modified Booth multiplier is one of the most popular and widely used multiplier because it realizes the reduction in the number of partial products by a factor of 2. Consider the multiplication of two n-bit signed numbers X and Y, where X is the n-bit multiplicand and Y is the n-bit multiplier. X and Y in two's complement form can be represented as

$$X = -x_{n-1}2^{n-1} + \sum_{i=0}^{n-2} x_i 2^i$$

$$Y = -y_{n-1}2^{n-1} + \sum_{i=0}^{n-2} y_i 2^i \quad (1)$$

In Modified Booth encoding, a *0* must be added to the right of Y, i,e $y_{-1} = 0$. The encoding is done by scanning the bits from right to left and grouping the bits into triplets Three bits of the multiplier $y_{2i+1}y_{2i}y_{2i-1}$ are mapped to y_i^1 and the corresponding operation is shown in table 1. Y can be expressed as,

$$Y=\sum_{i=0}^{n/2-1} M_i\, 2^{2i}$$

$$= \sum_{i=0}^{n/2-1}(-2y_{2i+1} + y_{2i} + y_{2i-1})2^{2i} \quad (2)$$

Table 1. BOOTH ENCODING TABLE

$y_{2i+1}\, y_{2i}\, y_{2i-1}$	operation	y_i^1	one_i	two_i	neg_i	$zero_i$	cor_i
000	0	0	0	0	0	1	0
001	+X	1	1	0	0	0	0
010	+X	1	1	0	0	0	0
011	+2X	2	0	1	0	0	0
100	-2X	-2	0	1	1	0	1
101	-X	-1	1	0	1	0	1
110	-X	-1	1	0	1	0	1
111	0	0	0	0	1	1	0

According to the encoded result shown in table 1, one of the multiple multiplicands -2X, -X, 0, X, 2X is chosen for the generation of each partial product *PPi.* Operation 2X can be implemented by shifting X left by one bit. For the negation operation, each bit of X is inverted and a binary value of *1* is added as the correction bit to the LSB of the next partial product row. The correction bit indicates whether the partial product row is positive ($cor_i=0$) or negative ($cor_i=1$). Fig. 1 shows the partial product matrix of an 8-bit modified Booth multiplier

Position	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PP_0							p_{08}	p_{07}	p_{06}	p_{05}	p_{04}	p_{03}	p_{02}	p_{01}	p_{00}
PP_1					p_{18}	p_{17}	p_{16}	p_{15}	p_{14}	p_{13}	p_{12}	p_{11}	p_{10}		Cor_0
PP_2			p_{28}	p_{27}	p_{26}	p_{25}	p_{24}	p_{23}	p_{22}	p_{21}	p_{20}		Cor_1		
PP_3	p_{38}	p_{37}	p_{36}	p_{35}	p_{34}	p_{33}	p_{32}	p_{31}	p_{30}		Cor_2				
									Cor_3						
	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

Fig.1 Partial product matrix of 8-bit modified Booth multiplier

The Partial product matrix of the modified Booth multiplier can be divided into *most significant part,* MP and *least significant part,* LP as shown in Fig. 2. For the efficient generation of the compensation bias, LP is further divided into LP_{major} and LP_{minor}. While LP_{major} indicates the most significant column of the LP, and LP_{minor} indicates the remaining columns in LP. In the direct-truncated multiplier, truncation is done by directly removing adder cells required for LP resulting in a very large truncation error. In fixed-width multipliers, after removing the adder cells

required for LP, a compensation circuit is added that estimates the carry value from LP to MP in order to reduce the truncation error. In this technique, the adder cells for computing LP_{minor} are removed and the carry generated from LP_{minor} to LP_{major} is estimated using a simple circuit. This estimated carry value is added to LP_{major} and the carry obtained from this addition is the desired compensation bias. Fig. 2 shows the partial product matrix indicating LP and MP.

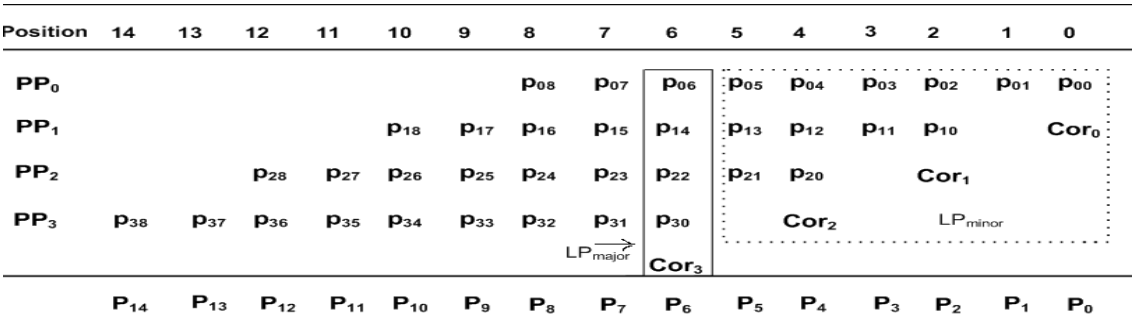

Fig 2 Partial product matrix showing MP and LP

(2n-1)-bit product can be expressed as

$$P = SUM\_MP + SUM\_LP \tag{3}$$

Where SUM_LP can be defined as

$$SUM\_LP = SUM\_LP_{major} + SUM\_LP_{minor} \tag{4}$$

SUM_LP^1 can be defined as

$$SUM\_LP^1 = SUM\_LP \times 2^n \tag{5}$$

$SUM\_LP^1_{major}$ and $SUM\_LP^1_{minor}$ can be expressed as

$$SUM\_LP^1_{major} = 2^{-2}(P_{06} + P_{14} + P_{22} + P_{30} + cor_3) \tag{6}$$

$$\begin{aligned} SUM\_LP^1_{minor} = {} & 2^{-3}(P_{05} + P_{13} + P_{21}) \\ & + 2^{-4}(P_{04} + P_{12} + P_{20} + cor_2) \\ & + 2^{-5}(P_{03} + P_{11}) \\ & + 2^{-6}(P_{02} + P_{10} + cor_1) \\ & + 2^{-7}(P_{01}) + 2^{-8}(P_{00} + cor_0) \end{aligned} \tag{7}$$

3. **Proposed error compensation method**

In this method, the Booth encoder outputs are used for the generation of the compensation bias since the generated partial products depend on the Booth encoder outputs. In this method, the relation between the carry value generated from LP_{minor} to LP_{major} and Booth encoder outputs has been explored through exhaustive simulation. Then, a simple compensation bias circuit has been derived that accepts the Booth encoder outputs as the inputs and generates the approximate carry value. The carry value so obtained is added to LP_{major} and the carry obtained from this addition stage is the desired compensation bias.

The carry may be generated from LP to the MP for any of the following operations X, 2X, -X, -2X. To identify whether the encoded bit y_i^1 has resulted in any operation, we can define

$$y_i^{11} = \begin{cases} 1 & \text{if } y_i^1 \neq 0 \\ 0 & \text{if } y_i^1 = 0 \end{cases} \quad (8)$$

For example, if the coded number $y_2^1\, y_1^1 y_0^1$ is one of the four values 100, 200, $\bar{1}00$ and $\bar{2}00$, then $y_2^{11}\, y_1^{11} y_0^{11}$ will be 100. Table 2 shows the actual carry values generated for 8-bit modified Booth multiplier from LP_{minor} to LP_{major} for all the possible values of $y_2^{11}\, y_1^{11} y_0^{11}$ which are obtained using exhaustive simulation. y_3^{11} is not considered since the partial product corresponding to this bit is not included in LP_{minor}. From table 2, it can be seen that considering $y_2^{11} y_1^{11} y_0^{11}$ = 000, the number of cases for which actual carry is 0 is 1024, and therefore approximate carry value for this case can be approximated as 0. Similarly the carry value can be approximated as 0 for all the cases except for $y_2^{11} y_1^{11} y_0^{11}$ = 111 where carry value is 1. To simplify the bias circuit, for the case y_2^{11} $y_1^{11} y_0^{11}$ = 011 & 110, carry value is approximated as 0. A single carry signal can be used to represent the approximate carry value from LP_{minor} to LP_{major} and this carry can be represented as

$$CARRY1 = y_2^{11} \& y_1^{11} \& y_0^{11} \quad (9)$$

Table 2. ACTUAL CARRIES FROM LP_{minor} to LP_{major} FOR 8-BIT

$y_2^{11} y_1^{11} y_0^{11}$	# of cases	# of cases with carry 0	# of cases with carry 1	# of cases with carry 2	# of cases with carry 3
000	1024	1024	0	0	0
001	3072	3024	48	0	0
010	3072	2880	192	0	0
011	9216	4576	4592	48	0
100	3072	2304	768	0	0
101	9216	4768	4400	48	0
110	9216	4480	4544	192	0
111	27648	4416	18368	4816	48

This can be implemented by a simple three input AND gate. This approximate carry signal is added to LP_{major} and the resulting carry signals from this step are added as a compensation bias to MP. Table 3 shows the actual carry values generated from LP_{minor} to LP_{major} for all the possible values of $y_3^{11} y_2^{11}\, y_1^{11} y_0^{11}$ which are obtained using exhaustive simulation for a 10-bit modified Booth multiplier. y_4^{11} is not considered due to the fact that the partial product corresponding to this bit is not included in LP_{minor}.

Table 3. ACTUAL CARRIES FROM LP_{minor} to LP_{major} FOR 10-BIT

$y_3^{11} y_2^{11} y_1^{11} y_0^{11}$	# of cases	# of cases With carry 0	# of cases With carry 1	# of cases With carry 2	# of cases With carry 3	# of cases With carry 4
0000	4096	4096	0	0	0	0
0001	12288	12240	48	0	0	0
0010	12288	12096	192	0	0	0
0011	36864	18400	18416	48	0	0
0100	12288	11520	768	0	0	0
0101	36864	18592	18224	48	0	0
0110	36864	18304	18368	192	0	0
0111	110592	17344	75456	17744	48	0
1000	12288	9216	3072	0	0	0
1001	36864	19360	17456	48	0	0
1010	36864	19072	17600	192	0	0
1011	110592	17664	74048	18832	48	0
1100	36864	17920	18176	768	0	0
1101	110592	18592	72768	19184	48	0
1110	110592	17664	73472	19264	192	0
1111	331776	12960	153344	151680	13744	48

From Table 3, it can be observed that if $y_3^{11}y_2^{11}y_1^{11}y_0^{11}$ = 0000, the number of cases for which actual carry is 0 is 4096 and therefore the carry value for this case can be approximated as 0. Similarly the carry value can be approximated as 0 for all the cases except for $y_3^{11}y_2^{11}y_1^{11}y_0^{11}$ = 0111, 1011, 1101, 1110 and 1111 where carry value is 1. To simplify the bias circuit, for the combinations $y_3^{11}y_2^{11}y_1^{11}y_0^{11}$ =0011, 0101, 0110, and 1100 carry value is approximated as 0 even though the number of cases for which the carry value is 1 or 0 are equal.. A single carry signal can be used to represent the approximate carry value from LP_{minor} to LP_{major} From table 3, using Karnaugh map simplification, approximate carry signal can be obtained.

Table 4. ACTUAL CARRIES FROM LP_{minor} to LP_{major} FOR 12-BIT

y_4^{11} $y_3^{11}y_2^{11}y_1^{11}y_0^{11}$	# of cases	# of cases with carry 0	# of cases with carry 1	# of cases with carry 2	# of cases with carry 3	# of cases with carry 4	# of cases with carry 5
00000	16384	16384	0	0	0	0	0
00001	49152	49104	48	0	0	0	0
00010	49152	48960	192	0	0	0	0
00011	147456	73696	73712	48	0	0	0
00100	49152	48384	768	0	0	0	0
00101	147456	73888	73520	48	0	0	0
00110	147456	73600	73664	192	0	0	0
00111	442368	69440	303040	69840	48	0	0
01000	49152	46080	3072	0	0	0	0
01001	147456	74656	72752	48	0	0	0
01010	147456	74368	72896	192	0	0	0
01011	442368	70656	299840	71824	48	0	0
01100	147456	73216	73472	768	0	0	0
01101	442368	71136	299456	71728	48	0	0
01110	442368	69376	301824	70976	192	0	0
01111	1327104	46994	617024	615360	47678	48	0
10000	49152	36864	12288	0	0	0	0
10001	147456	77728	69680	48	0	0	0
10010	147456	77440	69824	192	0	0	0
10011	442368	71936	294208	76176	48	0	0
10100	147456	76288	70400	768	0	0	0
10101	442368	75552	287552	79216	48	0	0
10110	442368	70656	296192	75328	192	0	0
10111	1327104	55040	612128	603936	55952	48	0
11000	147456	71680	72704	3072	0	0	0
11001	442368	74656	291648	76016	48	0	0
11010	442368	74368	291072	76736	192	0	0
11011	1327104	52960	612544	607040	54512	48	0
11100	442368	70656	293888	77056	768	0	0
11101	1327104	54944	611456	604992	55664	48	0
11110	1327104	51840	613376	606720	54976	192	0
11111	3981312	30656	835968	2243488	839680	31472	48

Table 4 shows the actual carry values generated from LP_{minor} to LP_{major} for each combination of y_4^{11} $y_3^{11}y_2^{11}y_1^{11}y_0^{11}$ obtained by exhaustive simulation for a 12-bit modified Booth multiplier. Yet again in this case, y_5^{11} is not considered since LP_{minor} does not depend on this bit. For of y_4^{11} $y_3^{11}y_2^{11}y_1^{11}y_0^{11}$ =00000 combination, since the actual carry is 0 for all the cases, carry value for this combination can be approximated as 0. Carry value can be approximated as 0 for $y_4^{11}y_3^{11}y_2^{11}y_1^{11}y_0^{11}$ =00001, 00010, 00100, 01000 and 10000. For the combinations, 00011, 00101, 00110, 01001, 01010, 01100, 10001, 10010, 10100, 11000 carry value can be approximated as 0. For the remaining combinations, carry value can be approximated as 1 except for the combination 11111, where the number of cases for which carry value is 2 is 2243488. Hence two number of carry signals are required to represent the approximate carry value from LP_{minor} to $LP_{major.}$ Again using table 4, with k-map simplification, approximate carry signals can be obtained.

4. Simulation Results and Discussions

The accuracy of different fixed-width multipliers can be measured in terms of the various performance parameters like maximum absolute error, average error and mean-square error.

$$\epsilon_{max} = Max\{|P_s - P_t|\}$$

$$\epsilon_{mean} = Ave\{|P_s - P_t|\}$$

$$\epsilon_{mse} = \sum\{|P_s - P_t|\}^2 / 2^{2n} \quad (10)$$

Where P_s and P_t represent the output of standard and fixed- width multiplier respectively. The maximum absolute error ϵ_{max} indicates the maximum value of $|P_s - P_t|$ for all input combinations. ϵ_{mean} is the average error and ϵ_{mse} represents mean-square error. Table 5 and 6 show the simulation results.

Table 5. COMPARISION OF ERRORS (8-Bit)

Fixed Width Multiplier(n=8)	ϵ_{max}	ϵ_{mean}	ϵ_{mse}
DTM	512	192.25	20775.75
FWM [Meng HungTsai et al (2003)]	298	73.51	8566.25
FWM [Kyung-Ju Cho et al (2004)**]**	128	38.24	2220.5
Proposed	192	78.12	8012.12

Table 6. COMPARISION OF ERRORS (10-Bit)

Fixed Width Multiplier(n=10)	ϵ_{max}	ϵ_{mean}	ϵ_{mse}
DTM	2560	960.25	$1.07\ X\ 10^6$
FWM [Meng HungTsai et al (2003)]	1398	313.628	$1.544\ X\ 10^5$
FWM [Kyung-Ju Cho et al (2004)**]**	768	164.733	$4.11\ X\ 10^4$
Proposed	1024	328.12	$1.419\ X\ 10^5$

DTM represents direct-truncated multiplier and FWM denotes Fixed-Width multiplier. Table 5 and 6 show the simulation results of direct truncated multiplier, Fixed-width multipliers and the proposed technique in terms of maximum absolute error, mean error and mean-square error for the justified comparison. The results depict that the proposed technique has reduced errors compared to direct truncated and the existing technique with considerably smaller hardware overhead. In many DSP and multimedia applications, the final output is obtained by adding a series of products instead of a single multiplication operation. Under such situations, truncation errors will be accumulated which results in a large error. Hence for a multiplier to be accurate, it should have small mean and mean-square error.

Fig.3a, 3b and 3c show the plot comparison of maximum absolute error, mean error and mean-square error respectively incurred by the proposed technique against existing multiplier for 8-bit multipliers. Fig. 4a, 4b and 4c

represent the corresponding results for 10-bit.multipliers From the plot, it can be concluded that the proposed method has less error compared to direct- truncated and the existing technique.

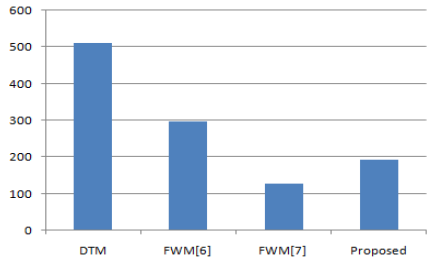

Fig. 3a

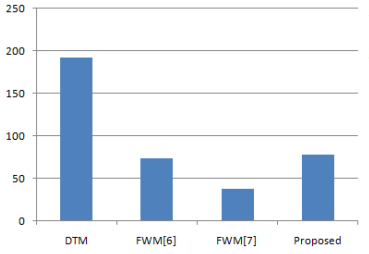

Fig. 3b

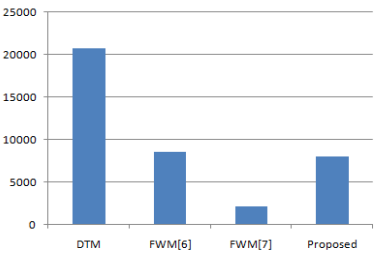

Fig. 3c

Fig.3 Comparison f maximum absolute error, mean error and mean-square error for 8 bit.

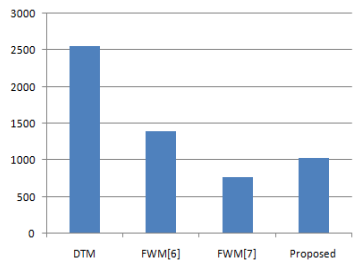

Fig. 4a

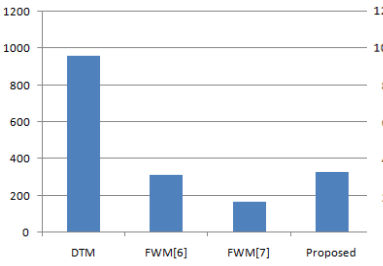

Fig .4b

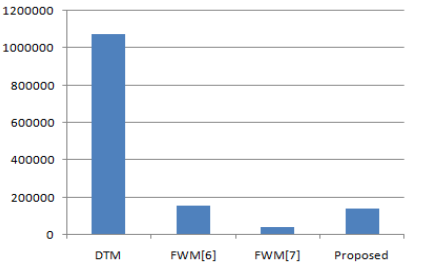

Fig. 4c

Fig.4 Comparison of maximum absolute error, mean error and mean-square error for 10 bit.

5. Conclusion

In this paper, a simple error compensation method for the modified Booth multiplier is proposed. For efficient generation of the compensation bias, the Booth encoder outputs have been used. The truncated bits have been divided into major group and the minor group. The estimation of the carry is done by exhaustive simulation .The results validate significant error reduction capabilities, in terms of the maximum absolute error, the mean error and the mean-square error compared with direct truncated and the other existing technique.

References

Sunder S Kidambi, Fayez El-Guibaly and Andreas 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

Jer Min Jou, Shiann Rong Kuang, and Ren Der Chen, "Design of Low-error Fixed-width Multipliers for DSP applications" , *IEEE Trans. Circuits Syst.* II, Exp. Briefs, vol. 46, no. 6, pp. 836–842, June 1999.

Lan-Da Van, Shuenn-Shyang Wang, and Wu-Shiung Feng, "Design of the Lower-error Fixed width Multiplier and its Application," *IEEE Trans. Circuits Syst.* II, Exp. Briefs, Vol. 47, No. 10, pp. 1112–1118, Oct. 2000.

Lan-Da Van and Chih-Chyau 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

Shyh-Jye Jou and Hui-Hsuan Wang,"Fixed-Width Multiplier for *DSP* Application*", in Proc 2000 Int Conf .Computer Design (ICCD*), *Austin, TX*, pp, 318-322, Sept 2000.

Shyh-Jye Jou, Meng-Hung Tsai and Ya-Lan Tsao, "Low-Error Reduced-Width Booth Multipliers for DSP Applications", *IEEE Trans. Circuits Syst*, Vol. 50, No. 11, pp. 1470–1474, Nov. 2003.

Kyung-Ju Cho, Kwang-Chul Lee, Jin-Gyun Chung, and K Keshab 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.

Jiun-Ping Wang, Shiann-Rong Kuang , and Shish-Chang Liang "High-Accuracy Fixed-Width Modified Booth Multipliers for Lossy Applications*", IEEE Transactions on Very Large Scale Integration Systems ,* Vol 19, No 1, January 2011.