Low-Error Reduced-Width Booth Multipliers for DSP Applications

Shyh-Jye Jou, Meng-Hung Tsai, and Ya-Lan Tsao

***Abstract*—A low-error reduced-width Booth multiplier using a proper compensation vector is proposed. The compensation vector is dependent on the input data. The compensation value will thus be adaptively adjusted when the input data are different. Design results from a 16 × 16 to 16 Booth multiplier show that the gate counts and critical path delay of the new reduced-width multipliers is 50.94% and 66.04% of the post-truncation reduced-width multiplier. A module generator of our proposed architecture is developed that will generate C code and Verilog code for each reduced-width multiplier. Pulse-shaping filter-system applications used in CATV transceivers show promising performance with 50.04% hardware reduction and 33.82% reduction in the critical path delay.**

***Index Terms*—Compensation vector, digital signal-processing (DSP) application, reduced-width Booth multiplier.**

I. Introduction

A standard two's-complement multiplier performs the following operations to obtain the $(m+n)$-bit product proportional derivative (PD) from an m-bit multiplicand $X$and an n-bit multiplier Y

$$
\begin{aligned}
\mathrm{PD} = X \cdot Y &= \left(-X_{m-1}2^{m-1} + \sum_{i=0}^{m-2} X_i 2^i\right)\\
&\quad \times \left(-Y_{n-1}2^{n-1} + \sum_{j=0}^{n-2} Y_j 2^j\right)\\
&= X_{m-1}Y_{n-1}2^{m+n-2} + \sum_{i=0}^{m-2}\sum_{j=0}^{n-2}(X_iY_j2^{i+j}) + 2^{m-1}\\
&\quad \times \left(-2^{n-1} + \sum_{j=0}^{n-2} \overline{X_{m-1}Y_j}2^j + 1\right)\\
&\quad + 2^{n-1}\left(-2^{m-1} + \sum_{i=0}^{m-2} \overline{X_iY_{n-1}}2^i + 1\right)\\
&= \left(-\mathrm{PD}_{m+n-1}2^{m+n-1} + \sum_{i=\tau}^{m+n-2} \mathrm{PD}_i 2^i\right)\\
&\quad + \sum_{i=0}^{\tau-1} \mathrm{PD}_i 2^i = \mathrm{MP} + \mathrm{LP}. \qquad (1)
\end{aligned}
$$

Equation (1) is the famous Baugh–Wooley [1] multiplier multiplication algorithm. The multiplication case with $m,\ n,$ and τ equal 6, 8, and 6, respectively, is shown in Fig. 1. In many digital signal-processing (DSP) applications, truncating several least-significant bits (LSBs) of the products (the internal word length problem) is very common to reduce hardware complexity and operation delay time as well as power consumption in hardware realization. A multiplier with a product of p bits ($p < m+n$ and $p \geq \max(m,n)$) is called an $m \times n$ to p reduced-width multiplier. A fixed-width multiplier is a special case of the reduced-width multiplier with $m = n = p$. In a reduced-width multiplier, using the post-truncation method, after doing all the operations, τ LSBs are truncated to obtain the product. This is the most accurate method. However, the post-truncation method wastes computation effort in terms of both computation time and hardware area. Therefore, several works focusing on low-error fixed-width or reduced-width array multipliers [2]–[10] were developed to solve this problem. Fixed constants [2]–[4] or input-data-dependent compensation vectors [5]–[10] are used to reduce the error. The data-dependent compensation vector works better than the fixed constant because the correction value depends on the input data value.

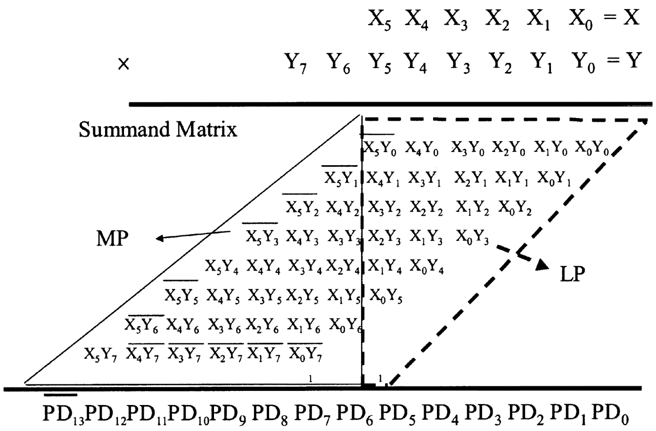

$X_5\ X_4\ X_3\ X_2\ X_1\ X_0 = X$
×
$Y_7\ Y_6\ Y_5\ Y_4\ Y_3\ Y_2\ Y_1\ Y_0 = Y$
Summand Matrix
$\overline{X_5Y_0}\ X_4Y_0\ X_3Y_0\ X_2Y_0\ X_1Y_0\ X_0Y_0$
$\overline{X_5Y_1}\ X_4Y_1\ X_3Y_1\ X_2Y_1\ X_1Y_1\ X_0Y_1$
$\overline{X_5Y_2}\ X_4Y_2\ X_3Y_2\ X_2Y_2\ X_1Y_2\ X_0Y_2$
MP
$\overline{X_5Y_3}\ X_4Y_3\ X_3Y_3\ X_2Y_3\ X_1Y_3\ X_0Y_3$
$X_5Y_4\ X_4Y_4\ X_3Y_4\ X_2Y_4\ X_1Y_4\ X_0Y_4$
LP
$\overline{X_5Y_5}\ X_4Y_5\ X_3Y_5\ X_2Y_5\ X_1Y_5\ X_0Y_5$
$\overline{X_5Y_6}\ X_4Y_6\ X_3Y_6\ X_2Y_6\ X_1Y_6\ X_0Y_6$
$X_5Y_7\ \overline{X_4Y_7}\ \overline{X_3Y_7}\ \overline{X_2Y_7}\ \overline{X_1Y_7}\ \overline{X_0Y_7}$
1
1
$\overline{PD_{13}}PD_{12}PD_{11}PD_{10}PD_9\ PD_8\ PD_7\ PD_6\ PD_5\ PD_4\ PD_3\ PD_2\ PD_1\ PD_0$

Fig. 1. The 6 × 8 two's complement multiplication operation.

TABLE I
RADIX-4 BOOTH CODER (X: MULTIPLICAND; Y: MULTIPLIER)

Y_{j+1}	Y_j	Y_{j-1}	Execution
0	0	0	None (0)
0	0	1	X
0	1	0	X
0	1	1	2X
1	0	0	-2X
1	0	1	-X
1	1	0	-X
1	1	1	None (0)

A faster low-error reduced-width Booth multiplier is proposed to improve the limited performance of reduced-width array multipliers. This reduced-width Booth multiplier has similar error performance as the array multiplier. However, it has a smaller dc offset and the operational speed is much faster. Section II will describe the proposed low-error reduced-width Booth multiplier. Section III will show the application. Conclusions are made in Section IV.

II. Low-Error Reduced-Width Booth Multiplier

Booth multiplication that is based on string recoding can multiply a pair of two's complement numbers without concern about the signs of the numbers [11]. In the radix-4 Booth multiplier, only $\lfloor (n+1)/(2) \rfloor$ multiplicand multiples are required to be summed together due to the string recoding (Table I). In this approach, $\lfloor x \rfloor$ stands for the largest integer less than or equal to the real number x. A block diagram of a parallel Booth multiplier is shown in Fig. 2(a). The Booth multiplier operation is shown in Fig. 2(b) for the 6 × 8 case. In this architecture, the encoder is used to perform string recoding and shift the multiplicand if necessary. The operations required to sum the multiplicand multiples are the same as that in the array multiplier case. Let $\alpha_{\tau-1}$ denote the sum of carriers from the column corresponding to $\mathrm{PD}_{\tau-1}$. A good

Manuscript received November 28, 2002; revised February 24, 2003. This work was supported in part by National Science Council under Grant NSC90-2218-E-008-006 and in part by MediaTek Inc. This paper was recommended by Associate Editor A. B. da Silva.

The authors are with the Department of Electrical Engineering, National Central University, Chung-Li, Taiwan, R.O.C.

Digital Object Identifier 10.1109/TCSI.2003.817779

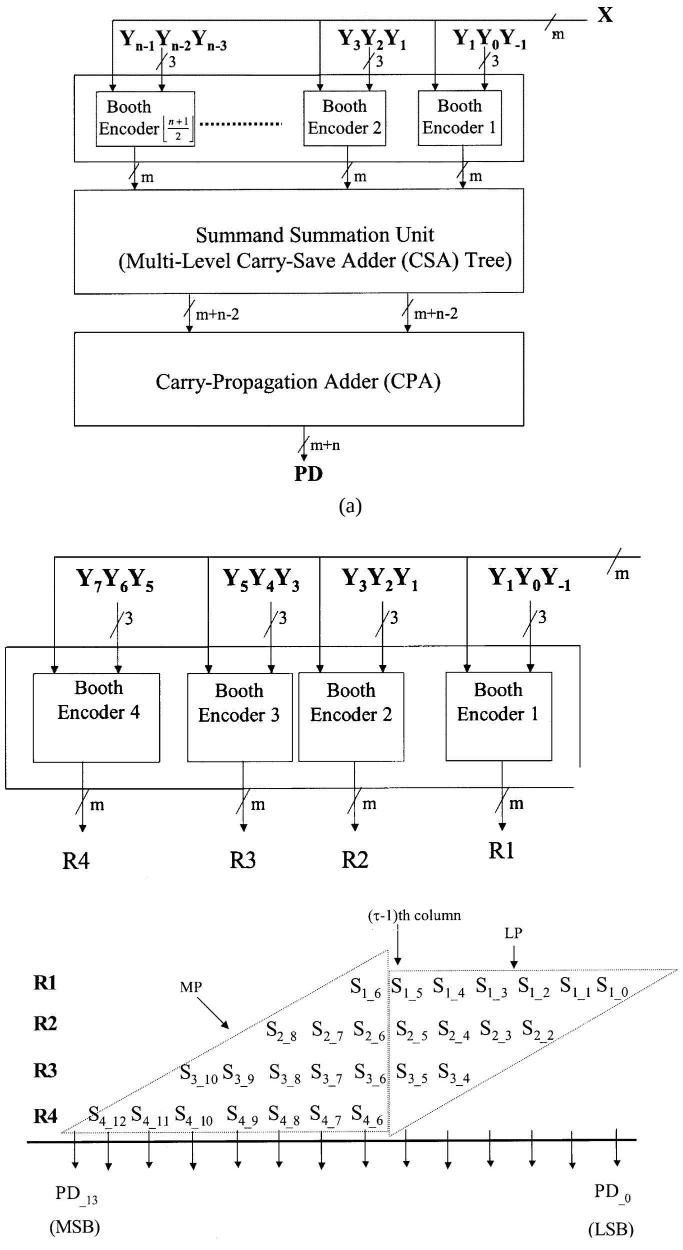

X
m
$Y_{n-1}Y_{n-2}Y_{n-3}$
$Y_3Y_2Y_1$
$Y_1Y_0Y_{-1}$
3
Booth Encoder $\lfloor\frac{n+1}{2}\rfloor$
Booth Encoder 2
Booth Encoder 1
m
Summand Summation Unit (Multi-Level Carry-Save Adder (CSA) Tree)
m+n-2
Carry-Propagation Adder (CPA)
m+n
PD
(a)
$Y_7Y_6Y_5$
$Y_5Y_4Y_3$
$Y_3Y_2Y_1$
$Y_1Y_0Y_{-1}$
Booth Encoder 4
Booth Encoder 3
Booth Encoder 2
Booth Encoder 1
R4
R3
R2
R1
(τ-1)th column
LP
MP
R1
R2
R3
R4
S_{1_6} S_{1_5} S_{1_4} S_{1_3} S_{1_2} S_{1_1} S_{1_0}
S_{2_8} S_{2_7} S_{2_6} S_{2_5} S_{2_4} S_{2_3} S_{2_2}
S_{3_10} S_{3_9} S_{3_8} S_{3_7} S_{3_6} S_{3_5} S_{3_4}
S_{4_12} S_{4_11} S_{4_10} S_{4_9} S_{4_8} S_{4_7} S_{4_6}
$PD_{\_13}$
(MSB)
$PD_{\_0}$
(LSB)
(b)

Fig. 2. (a) Block diagram of parallel Booth multiplier. (b) Example of 6 × 8 parallel Booth multipliers.

estimation of $\alpha_{\tau-1}$ will reduce the error caused by removing the LP segment. According to the multiplication operation shown in Fig. 2(b), for $\tau = 6$ we have

$$\begin{aligned}\alpha_5 = \lfloor & 2^{-1}(S_{1\_5}+S_{2\_5}+S_{3\_5})+2^{-2}(S_{1\_4}+S_{2\_4}+S_{3\_4})\\ &+2^{-3}(S_{1\_3}+S_{2\_3})+2^{-4}(S_{1\_2}+S_{2\_2})\\ &+2^{-5}S_{1\_1}+2^{-6}S_{1\_0}\rfloor. \end{aligned} \tag{2}$$

In general

$$\begin{aligned}\alpha_{\tau-1} = \Big\lfloor & 2^{-1}\left(S_{1\_\tau-1}+S_{2\_\tau-1}+\cdots+S_{\lceil\frac{\tau}{2}\rceil\_\tau-1}\right)\\ &+2^{-2}\left(S_{1\_\tau-2}+S_{2\_\tau-2}+\cdots+S_{\lceil\frac{\tau-1}{2}\rceil\_\tau-2}\right)\\ &+\cdots+2^{-(\tau-1)}S_{1\_1}+2^{-\tau}S_{1\_0}\Big\rfloor = \lfloor 2^{-1}\beta+\lambda\rfloor\end{aligned}$$

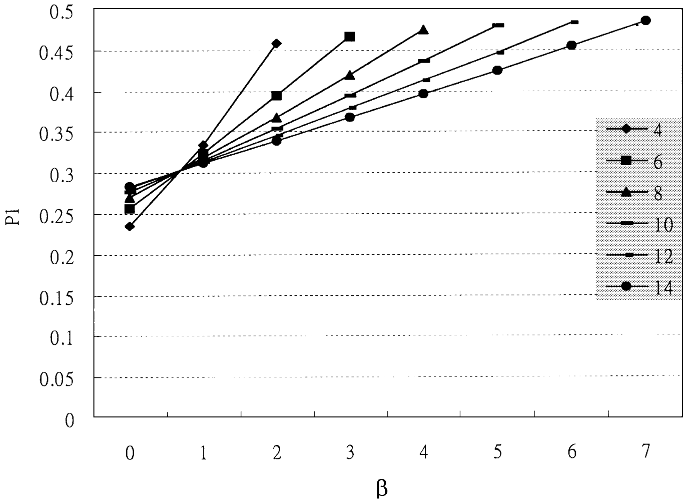


Fig. 3. The $\boldsymbol{p1(S_{i\_j})}$ distribution of Booth multiplier for $\boldsymbol{\tau = 4}$ to $\mathbf{12}$.

and

$$\begin{aligned}\beta &= S_{1\_\tau-1}+S_{2\_\tau-1}+\cdots+S_{\lceil\frac{\tau}{2}\rceil\_\tau-1}, \quad \lambda = 2^{-2}\\ &\times\left(S_{1\_\tau-2}+S_{2\_\tau-2}+\cdots+S_{\lceil\frac{\tau-1}{2}\rceil\_\tau-2}\right)\\ &+\cdots+2^{-(\tau-1)}S_{1\_1}+2^{-\tau}S_{1\_0}\end{aligned} \tag{3}$$

where $\lceil x\rceil$ stands for the smallest integer that is larger than or equal to the real number x. The value of β is the total number of "1" in the column corresponding to $\mathrm{PD}_{\tau-1}$. If we can express λ in terms of β and τ, we can obtain the compensation value in terms of only β and τ. In the following derivation, it is assumed that the probability of each input data bit equaling 1 is 0.5 and the probability that the recoded bit S_{i_j} is equal to 1 is denoted as $P1(S_{i\_j})$. Because of string recoding, the analytical analysis of $\alpha_{\tau-1}$ is more complicated than that for the array multiplier. This is because S_{i_j} is not the original input data bit pattern. Given a $\beta, P1(S_{i\_j})$ distribution using simulation for τ from 4 to 12 is shown in Fig. 3. Note that the probability distribution of β is only a function of τ. This means that the 9×8 to 9 and 10×11 to 13 cases have the same probability distribution of β. Using the $P1(S_{i\_j})$ concept we can rewrite λ as

$$\lambda = \sum_{k=1}^{\tau-1}\frac{1}{2^{k+1}}\times P1(S_{i\_j})\times\left\lceil\frac{\tau-k}{2}\right\rceil. \tag{4}$$

By using linear regression line analysis [13], $P1(S_{i\_j})$ can be approximated as a first-order polynominal

$$P1(S_{i\_j}) = \frac{0.41}{\tau}\times\beta+0.58(0.01\times\tau+0.37). \tag{5}$$

Taking (5) and (4) into (3), and using the round-off method (plus 0.5), produces

$$\begin{aligned}\alpha_{\tau-1} = \Bigg\lfloor & 2^{-1}\beta+\left(\sum_{k=1}^{\tau-1}\frac{1}{2^{k+1}}\times\left(\frac{0.41}{\tau}\times\beta\right.\right.\\ &\left.\left.+\ 0.58(0.01\times\tau+0.37)\times\left\lceil\frac{\tau-k}{2}\right\rceil\right)\right)+0.5\Bigg\rfloor\end{aligned} \tag{6}$$

where $\alpha_{\tau-1}$ is now the compensation value. The occurrence probability for different β values is produced using computer simulation.

TABLE II
PROBABILITY OF $\alpha_{\tau-1}$ WITH DIFFERENT VALUES OF β AND τ IN REDUCED-WIDTH BOOTH MULTIPLIERS

τ	β+2	β+1	β	β-1	β-2	β-3	Expected Value
4	0	2.34%	85.94%	11.72%	0	0	β-0.09
6	1.27%	36.35%	56.88%	5.49%	0	0	β+0.33
8	2.11%	37.06%	53.05%	7.75%	0.04%	0	β+0.33
10	3.23%	36.78%	50.30%	9.54%	0.14%	0	β+0.33
12	4.38%	36.24%	47.97%	11.09%	0.31%	3.58E-7	β+0.33
14	5.52%	35.66%	45.88%	12.38%	0.55%	1.20E-5	β+0.33

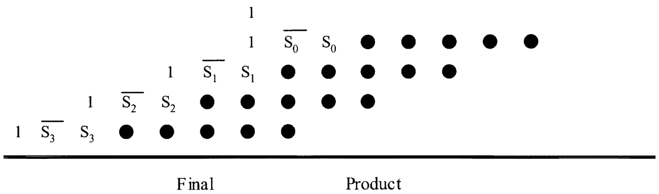
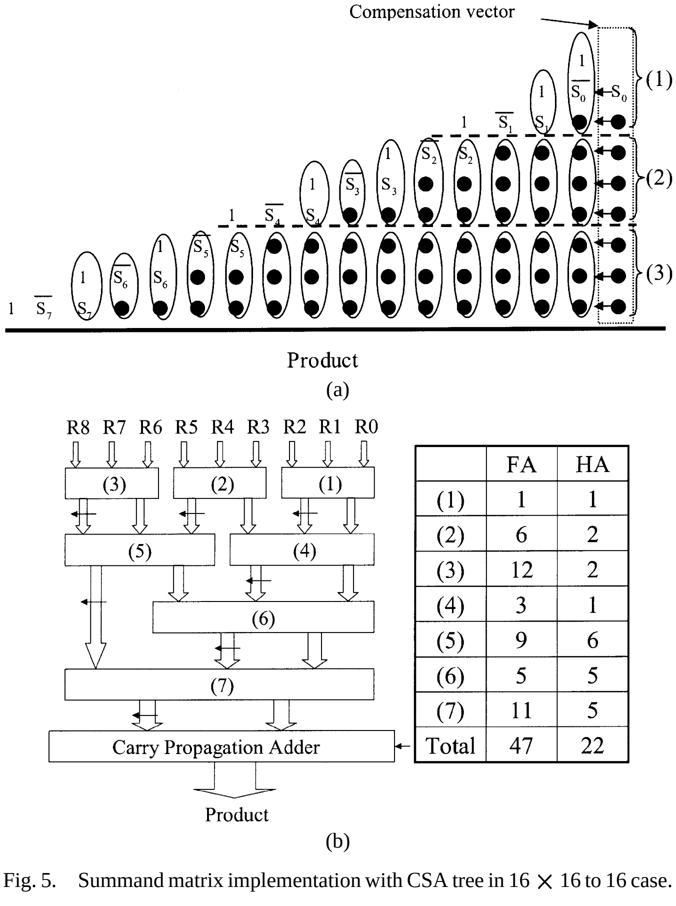


Fig. 4. Summand matrix diagram with sign extension in 6 × 8 Booth multiplier.

TABLE III
ERROR COMPARISONS FOR THE PROPOSED AND POST-TRUNCATION METHODS IN BOOTH MULTIPLIERS

m_n_p	τ	Ave_Error		Max_Error		SNR (dB)	
		Proposed	Post-Trun.	Proposed	Post-Trun.	Proposed	Post-Trun.
4_6_6	4	5	7	15	15	23.07	20.51
8_9_11	6	23	30	85	63	51.49	49.78
12_11_15	8	107	126	443	255	74.38	73.61
10_11_11	10	477	509	2181	1023	49.37	49.45
13_14_15	12	2084	2045	10363	4095	72.68	73.51
16_16_16	16	38315	32761	218235	65535	77.49	79.52



	FA	HA
(1)	1	1
(2)	6	2
(3)	12	2
(4)	3	1
(5)	9	6
(6)	5	5
(7)	11	5
Total	47	22

Fig. 5. Summand matrix implementation with CSA tree in 16 × 16 to 16 case.

TABLE IV
GATE COUNTCOMPARISON FOR DIFFERENT 16 × 16 TO 16 BOOTH MULTIPLIER ARCHITECTURES

	Post-Truncation	Proposed
Encoder, Sign-extension (Gate)	1216	630
Summand summation unit (Gate) [using CSA tree]	715	345
Carry propagation adder (Gate) [using ripple adder]	140	80
Total area (Gate)	2071	1055
Critical path delay (ns)	14.40	9.51

TABLE V
ERROR COMPARISONS FOR THE PROPOSED REDUCED-WIDTH BOOTH AND τ — 1 MULTIPLIER

τ	Ave_Error		Max_Error		SNR(dB)	
	Proposed	τ −1	Proposed	τ −1	Proposed	τ −1
6	23	25	85	75	21.39	21.03
8	107	149	443	393	32.24	30.18
10	477	786	2181	2223	43.35	40.01
12	2084	3926	10363	10905	54.62	50.43
14	8978	17985	48005	51808	66.01	61.33
16	38315	82034	218235	246961	77.49	72.98

TABLE VI
ERROR COMPARISONS BETWEEN REDUCED-WIDTH BOOTH AND TWO'S COMPLEMENT ARRAY MULTIPLIERS

τ	Ave_Error		Ave_Error_DC		Max_Error		SNR (dB)	
	Booth	Array	Booth	Array	Booth	Array	Booth	Array
4	5	5	0.25	-3.75	15	17	11.06	10.21
6	23	24	0.25	-15.75	85	89	21.39	21.09
8	107	106	0.25	-63.75	443	441	32.24	32.33
10	477	456	0.25	-255.75	2181	2105	43.35	43.74
12	2084	1907	0.25	-1023.80	10363	9785	54.62	55.39

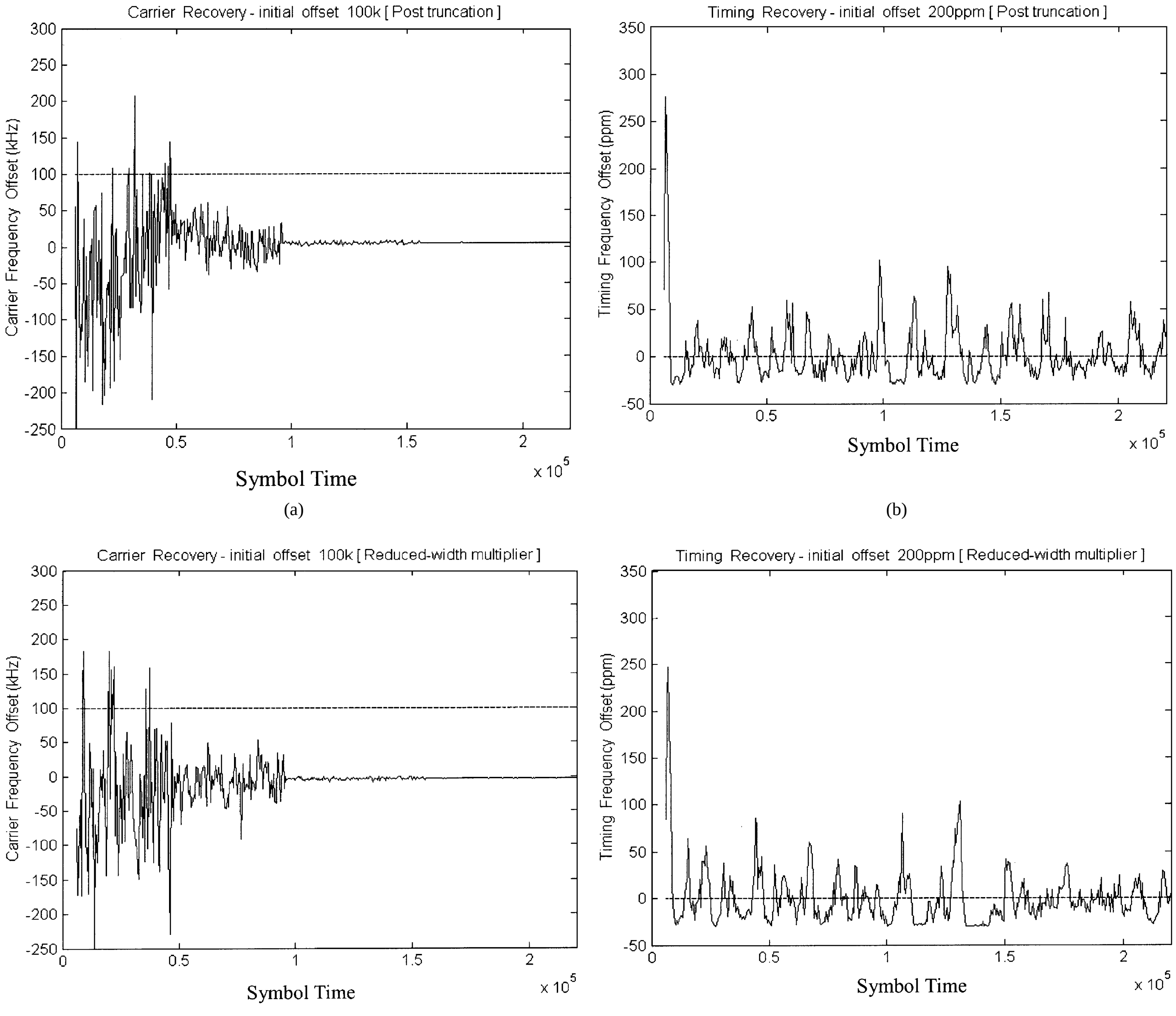
The probabilities of different $\alpha_{\tau-1}$ value for different τ are shown in Table II. It is clear that the best compensation value is β.

A compact encoding circuit [14] is used for the encoder. To reduce the area complexity and power dissipation, an improvement of sign extension method is required [15]. Using a 6 × 8 Booth multiplier, the sign extension portion (S) of the summand matrix can be expressed as

$$\begin{aligned} S &= S_0 \sum_{i=6}^{13} 2^i + \left(S_1 \sum_{i=6}^{11} 2^i \right) \cdot 2^2 + \left(S_2 \sum_{i=6}^{9} 2^i \right) \cdot 2^4 \\ &\quad + \left(S_3 \sum_{i=6}^{7} 2^i \right) \cdot 2^6 \\ &= (2^7 + \overline{S_0} 2^6) + (2^9 + \overline{S_1} 2^8) + (2^{11} + \overline{S_2} 2^{10}) \\ &\quad + (2^{13} + \overline{S_3} 2^{12}) + 2^6 \end{aligned} \tag{7}$$

where S_0, S_1, S_2, and S_3 are the sign of the first to fourth rows of the summand matrix, respectively. The summand matrix is shown in Fig. 4. The summand summation unit is processed using the carry saved adder (CSA) tree and the carry propagation adder (CPA).

In this proposed reduced-width Booth multiplier architecture, only the columns in the summand summation unit corresponding to PD_{m+n-1} to $\mathrm{PD}_{\tau-1}$ are required. Therefore, the complexity of the



(a) (b)

(c) (d)

Fig. 6. The performance of (a) carrier recovery and (b) timing recovery with post-truncation and the performance of (c) carrier recovery and (d) timing recovery with the proposed reduced-width.

encoder and summand summation unit is reduced by nearly half for a fixed-width multiplier. The bit products in column $\tau - 1$ are used as the compensation vector and are directly applied to column τ. By doing so, no additional hardware circuits are needed to generate the compensation vector. This method generates a data-dependent compensation vector and it guarantees that if one of the multiplicand X or multiplier Y is zero, then, the output product is also zero. The proposed method in [12] is for a fixed-width Booth multiplier and the compensation is a fixed constant obtained based on a statistical average. Let FP be the product from the reduced-width multiplier and PD the product from a standard multiplier. The maximum error (Max_Error), the average error $(\text{Ave\_Error}, \text{Ave\_Error\_dc})$, and the SNR equation for all input pairs are listed below:

$$\text{Max\_Error} = \text{Max}(|\text{PD} - FP|) \tag{8}$$

$$\text{Ave\_Error} = \frac{\sum |\text{PD} - FP|}{2^{m+n}}$$

$$\text{Ave\_Error\_dc} = \frac{\sum (\text{PD} - FP)}{2^{m+n}} \tag{9}$$

$$E_{\text{SNR}} = 10 \log \frac{E[|\text{PD}|^2]}{E[|\text{PD} - FP|^2]}. \tag{10}$$

The performance comparisons with the post-truncation multiplier are listed in Table III. Designs using a standard cell library (0.35-μm CMOS technology) and synthesized using a synthesis tool were carried out for the proposed 16×16 to16 Booth multiplier. The summand matrix needs four CSA tree levels as shown in Fig. 5. Although the compensation vector must be applied to column 16, through careful arrangement (putting them into the first bit of the CSA carry outputs and the CPA carry in), the CSA tree level does not increase. The total hardware complexity is listed in Table IV. The results show that the proposed low-error reduced-width Booth mulitplier area is 50.94% and the critical path delay is 66.04% of the post-truncation Booth multiplier. To further compare the performance of the proposed reduced-width Booth multiplier, the $\tau - 1$ method was used to implement a reduced-width Booth multiplier. The $\tau - 1$ method uses one extra column, the $(\tau - 1)$th column as indicated in Fig. 2(b), to compute the product. The hardware requirement is therefore quite the same as the proposed method. The performance comparisons are listed in Table V. It is clear that the proposed design is better than that for $\tau - 1$ method and the difference in SNR is 4.51 dB of 16 ×16 to 16 case.

The performance comparisons of the proposed reduced-width Booth multiplier and the two's complement reduced-width array multiplier are listed in Table VI. They have similar error performance. The dc bias is very small and is constant in Booth architecture. This is a very good characteristic in practical applications. The area of the proposed reduced-width Booth multiplier is 1.3 times that of the two's complement reduced-width array multiplier. However, the operating speed of the reduced-width Booth multiplier is 1.69 times faster than that of the two's complement reduced-width array multiplier. If high-speed is required, faster architectures can be used in the CPA of the proposed Booth multiplier. For example, the multilevel carry look-ahead adder could be used instead of the carry ripple adder. The results show that the critical path delay drops from 9.51 to 6.47 ns and the area increases from 1055 gates to 1270 gates. From the operational speed point of view, the reduced-width Booth multiplier is a very good choice for two's complement operations.

III. Applications

A module generator for the low-error reduced-width Booth multiplier was developed. By giving the m, n, and p parameters of the multiplier, the C subroutine or the Verilog RTL code is automatically generated. In this way, the behavior of the low-error reduced-width Booth multiplier can cooperate with entire system application simulations. The proposed low-error reduced-width Booth multipliers are used in the pulse-shaping filters (37 taps) of a quadrature amplitude modulation (QAM) mode CATV transceiver [16]. In the original CATV transceiver system level simulation, the multipliers in the pulse shaping filters are post-truncation multiplier. The bit numbers were determined after simulations ($m = 14,\ n = 4,$ and $p = 14$ in the transmitter and $m = n = p = 14$ in the receiver). Fig. 6(a) and (b) shows the carrier recovery and timing recovery performances in this system. The proposed low-error reduced-width Booth multipliers were used to perform simulations with the same bit numbers. The simulation results are shown in Fig. 6(c) and (d). Both Booth multiplier designs exhibited the same performance. In both designs, the carrier recovery (timing recovery) worked with an initial frequency offset ± 200 kHz (symbol rate offset ± 200 ppm) and with a final phase jitter within 2.6 °(2.0°). In pulse-shaping filter hardware implementation, the proposed low-error reduced-width Booth multiplier saves 50.04% of the hardware area and 33.82% of the critical path delay with standard cell library of 0.35-μm CMOS technology.

IV. Conclusion

The error-performance analysis of a reduced-width Booth multiplier was carried out. The compensation vector for the low-error reduced-width multiplier was derived and verified by simulation. Designs using this input-data-dependent compensation method were carried out on Booth multipliers. Standard cell library of 0.35-μm CMOS technology was used to design 16×16 to16 multipliers with different structures. In the low-error reduced-width Booth multiplier, the gate count was 50.94% and the critical path 66.04% of the post-truncation Booth multiplier. A module generator for the proposed architectures was developed that generates C code and Verilog code for each reduced-width multiplier. Therefore, the behavior and RTL simulation and design of a system can be performed if the proposed low-error reduced-width multiplier is included.

References

[1] C. R. Baugh and B. A. Wooley, "A two's complement parallel array multiplication algorithm," *IEEE Trans. Comput.*, vol. C-22, pp. 1045–1047, Dec. 1973.

[2] Y. P. Lim, "Single-precision multiplier with reduced circuit complexity for signal processing applications," *IEEE Trans. Comput.*, vol. 41, pp. 1333–1336, Oct. 1992.

[3] M. J. Schulte and E. E. Swartzlander Jr., "Truncated multiplication with correction constant," in *VLSI Signal Processing, VI*. New York: IEEE Press, 1993, pp. 388–396.

[4] S. S. Kidambi, F. El-Guibaly, and A. Antoniou, "Area-efficient multipliers for digital signal processing applications," *IEEE Trans. Circuits Syst. II*, vol. 43, pp. 90–95, Feb. 1996.

[5] E. J. King and E. E. Swartzlander Jr., "Data-dependent truncation scheme for parallel multipliers," in *Proc. 31st Asilomar Conf. Signals, Systems, Computers*, 1998, pp. 1178–1182.

[6] E. E. Swartzlander Jr., "Truncated multiplication with approximate rounding," in *Proc. 33rd Asilomar Conf. Signals, Systems, Computers*, 1999, pp. 1480–1483.

[7] J. M. Jou, S. R. Kuang, and R. D. Chen, "Design of low-error fixed-width multipliers for DSP applications," *IEEE Trans. Circuits Syst. II*, vol. 46, pp. 836–842, June 1999.

[8] S. J. Jou and H. H. Wang, "Fixed-width multiplier for DSP application," in *IEEE Int. Symp. Computer Design*, Sept. 2000, pp. 318–322.

[9] L. D. Van, S. S. Wang, and W. S. Feng, "Design of the lower error fixed-width multiplier and its application," *IEEE Trans. Circuits Syst. II*, vol. 47, pp. 1112–1118, Oct. 2000.

[10] L. D. Van and S. H. Lee, "A generalized methodology for lower-error area-efficient fixed-width multipliers," *Proc. IEEE Int. Symp. Circuits and Systems*, pp. 65–68, May 2002.

[11] A. D. Booth, "A signed binary multiplication technique," *Quart. J. Mech. Appl. Math.*, pt. 2, vol. 4, pp. 236–240, 1951.

[12] K. H. Lee and C. S. Rim, "A hardware reduced multiplier for low power design," in *Proc. 2nd IEEE Asia Pacific Conf. ASICs*, 2000, pp. 331–334.

[13] D. C. Montgomery and E. A. Peak, *Introduction to Linear Regression Analysis*. New York: Wiley, 1982.

[14] R. Fried, "Minimizing energy dissipation in high-speed multipliers," in *Proc. IEEE Int. Symp. Low Power Electronics and Design*, 1997, pp. 214–219.

[15] E. de Angel and E. E. Swartzlander Jr., "Low power parallel multipliers," in *VLSI Signal Processing, IX*. New York: IEEE Press, 1996, pp. 199–208.

[16] S. J. Jou, C. H. Kuo, M. T. Shiau, J. Y. Heh, and C. K. Wang, "VLSI implementation of timing recovery and carrier recovery for QAM/VSB dual mode," in *Proc. Int. Symp. VLSI Technology, Systems.Applications*, Taipei, Taiwan, R. O.C., June 1999, pp. 159–162.