Modified Booth Multiplier To Reduce The Partial Products by Half

Manibharathi.M and Dhivya.M
Department of Electronics and Communication Engineering
Panimalar Engineering College,
Chennai, India
manibharathim@gmail.com

D.Selvaraj
Department of Electronics and Communication Engineering
Panimalar Engineering College
Chennai, India
mails2selvaraj@yahoo.com

***Abstract*— The Modified Booth Encoding has been widely adopted in parallel multipliers since it reduces the partial products by half. The Conventional modified Booth encoding generates an irregular partial product array because of the extra partial product bit at the least significant bit position of each partial product row. But the proposed algorithm is used to generate a regular partial product array with fewer partial product rows. This reduces the complexity of partial product reduction which in turn reduces the area, delay and power of MBE multipliers. Implementation results demonstrate that the proposed MBE multipliers really achieve significant improvement in area, delay and power consumption when compared with conventional MBE multipliers.**

Keywords-Modified Booth, multiplier, partial products,Radix-4 algorithm.

I. INTRODUCTION

The performance of multiplier frequently dominate the system's performance and power dissipation for multimedia and digital signal processing (DSP) applications. Multiplication process consists of three major steps:1) Recoding and generating partial products ;2) Reducing the partial products by partial product reduction scheme to two rows; and 3) Adding the two rows of partial products by using a carry- propagate adder to obtain the final product. There are already many techniques developed in the past years for these steps to improve the performance of multipliers. In this brief, we will focus on the first step (i.e., partial product generation) to reduce the area, delay and power consumption of multipliers.

The main focus of recent multipliers has been on rapidly reducing the partial product rows by using some kind of circuit optimization and identifying the critical paths and signal races. There is no doubt that MBE is efficient when it comes to reducing the partial products. This is because, by applying MBE, the number of partial products to be accumulated is reduced from n to n/2. In Fig.1(a), the conventional MBE algorithm generates n/2+1 partial product rows rather than n/2 due to the extra partial product bit (neg bit) at the least significant bit position of each partial product row for negative encoding, leading to an irregular partial product array and a complex reduction tree. Some approaches [1], [2] have been proposed to generate more regular partial arrays, as shown in Fig.1 (b) and (c) for MBE multipliers. Thus the area, delay and power consumption of a reduction tree, as well as the whole MBE multiplier, can be reduced.

b_p	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PP0						$\overline{s_0}$	s_0	s_0	p_{07}	p_{06}	p_{05}	p_{04}	p_{03}	p_{02}	p_{01}	p_{00}
PP1					1	$\overline{s_1}$	p_{17}	p_{16}	p_{15}	p_{14}	p_{13}	p_{12}	p_{11}	p_{10}	neg_0	
PP2			1	$\overline{s_2}$	p_{27}	p_{26}	p_{25}	p_{24}	p_{23}	p_{22}	p_{21}	p_{20}	neg_1			
PP3	1	$\overline{s_3}$	p_{37}	p_{36}	p_{35}	p_{34}	p_{33}	p_{32}	p_{31}	p_{30}	neg_2					
PP4										neg_3						

(a)

b_p	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PP0						$\overline{s_0}$	s_0	s_0	p_{07}	p_{06}	p_{05}	p_{04}	p_{03}	p_{02}	p_{01}	τ_{00}
PP1					1	$\overline{s_1}$	p_{17}	p_{16}	p_{15}	p_{14}	p_{13}	p_{12}	p_{11}	τ_{10}	c_0	
PP2			1	$\overline{s_2}$	p_{27}	p_{26}	p_{25}	p_{24}	p_{23}	p_{22}	p_{21}	τ_{20}	c_1			
PP3	1	$\overline{s_3}$	p_{37}	p_{36}	p_{35}	p_{34}	p_{33}	p_{32}	p_{31}	τ_{30}	c_2					
PP4										c_3						

(b)

b_p	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PP0						$\overline{s_0}$	s_0	s_0	p_{07}	p_{06}	p_{05}	p_{04}	p_{03}	p_{02}	p_{01}	p_{00}
PP1					1	$\overline{s_1}$	p_{17}	p_{16}	p_{15}	p_{14}	p_{13}	p_{12}	p_{11}	p_{10}	neg_0	
PP2		1	1	$\overline{s_2}$	p_{27}	p_{26}	p_{25}	p_{24}	p_{23}	p_{22}	p_{21}	p_{20}	neg_1			
PP3	$\overline{s_3}$	s_3	t_7	t_6	t_5	t_4	t_3	t_2	t_1	t_0	neg_2					

(c)

Fig.1. Conventional MBE partial product arrays for 8×8 multiplication

The proposed algorithm reduces the partial product from n/2+1 to n/2 by incorporating the last neg bit into the sign extension bit of the first partial product row, and almost no overhead is introduced to the partial generator. More regular

partial product array and fewer partial product rows result in a small and fast reduction tree, so that the area, delay and power of MBE multipliers can further be reduced. Experimental results shows that proposed MBE multipliers with a regular partial product array can achieve significant improvement in area, delay and power consumption when compared with conventional MBE multipliers.

TABLE I

MBE TABLE

b_{2i+1}	b_{2i}	b_{2i-1}	Operation	neg_i	two_i	one_i	p_{ij}
0	0	0	+0	0	0	0	0
0	0	1	+A	0	0	1	a_j
0	1	0	+A	0	0	1	a_j
0	1	0	+2A	0	1	0	a_{j-1}
1	0	0	-2A	1	1	0	$\overline{a_{j-1}}$
1	0	1	-A	1	0	1	$\overline{a_j}$
1	1	0	-A	1	0	1	$\overline{a_j}$
1	1	1	-0	0	0	0	0

II. CONVENTIONAL MBE MULTILIER

Consider the multiplication of two n- bit integer numbers A (multiplicand) and B(multiplier) in two's complement representation, i.e.,

$$A = -a_{n-1}\ 2^{n-1} + \sum_{i=0}^{n-2} a_i\, 2^i \qquad (1)$$

$$B = -b_{n-1}\ 2^{n-1} + \sum_{i=0}^{n-2} b_i\, 2^i$$

In MBE, B in (1) becomes

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

Where b_{-1}=0, and $m_i \in$ {-2,-1, 0, 1, 2}. According to the encoder results from B, the booth selectors choose -2A, -A, 0, A or 2A to generate the partial product rows, as shown in Table I. The 2A in Table I is obtained by left shifting A one bit. Negation operation is achieved by complementing each bit of A (one's complement) and adding "1" to the least significant bit. Adding "1" implemented as a correction bit neg, which implies that the partial product row is negative (neg=1) or positive (neg=0). In addition, partial product rows are represented in 2's complement representation and each row is left shifted two bit position with respect to the previous row, sign extensions are required to align the most significant parts of partial product rows. These extra sign bits will significantly complicate the reduction tree. Therefore, the many sign extension schemes [3]-[6] have been proposed to prevent extending the sign bit of each row to the (2n-1) th bit position.

Fig.1 (a) illustrates the MBE partial products array for an 8×8 multiplier with a sign extension prevention procedure, where s_i is the sign bit of the partial product row PP_i, $\overline{s_i}$ is the complement of s_i, and b_p indicates the bit position. As can be seen in Fig.1 (a), MBE reduces the number of partial product rows by half, but the correction bits result in an irregular partial product array and one additional partial product row. To have a more regular least significant part of each partial product row PP_i , the authors in [1] added the least significant bit p_{i0} with neg_i in advance and obtained new least significant bit τ_{io} and a carry c_i . Note that the both τ_{io} and c_i are generated no later than other partial product bits.Fig.1 (b) shows the 8×8 MBE partial product array generated by the approach proposed in [1]. Since c_i is at the left one bit position of neg_i, the require addition in the reduction tree are reduced. However, the approach does not remove the additional partial product row PP_4.

The problem is removed [2] by directly producing 2's complement representation of the last partial product row $PP_{n/2-1}$while the other partial product are produced so that the last neg bit will not be necessary. An efficient method and circuit are developed [2] to find the 2's complement of the last partial product row in a logarithmic time. As shown in Fig.1(c), the 10-bit last partial product row and its neg bit in Fig.1(a) are replaced by the 10-bit 2's complemented partial product rows ($\overline{s_3}$, s_3,t_7,t_6,…..,t_0) without the last neg bit. Note that one extra "1" is added at the fourteenth bit position to obtain the correct final product. The approach simplifies and speeds up the reduction step, particularly for a multiplier that is in the size of 2 and uses 4-2 compresses [7], [8] to achieve modularity and high performance. However, the approach must additionally develop and design the 2's complement logic, which possibly enlarges the area and delay of the partial product generator and lessens the benefit contributed by removing the extra row.

III. PROPOSED MULTIPLIER

The proposed MBE multiplier combines the advantages of these two approaches presented in [1] and [2] to produce a very regular partial product array as shown in Fig.2. In the partial product array, not only each neg_i is shifted left and replaced by c_i but also the last neg bit is removed by using a simple approach described in detail in the following section.

b_p	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PP0						α_2	α_1	α_0	p_{07}	p_{06}	p_{05}	p_{04}	p_{03}	p_{02}	p_{01}	τ_{00}
PP1					1	$\overline{s_1}$	p_{17}	p_{16}	p_{15}	p_{14}	p_{13}	p_{12}	p_{11}	τ_{10}	c_0	
PP2				1	$\overline{s_2}$	p_{27}	p_{26}	p_{25}	p_{24}	p_{23}	p_{22}	p_{21}	τ_{20}	c_1		

PP3 1 $\bar{s_3}$ p_{37} p_{36} p_{35} p_{34} p_{33} p_{32} τ_{31} τ_{30} c_2

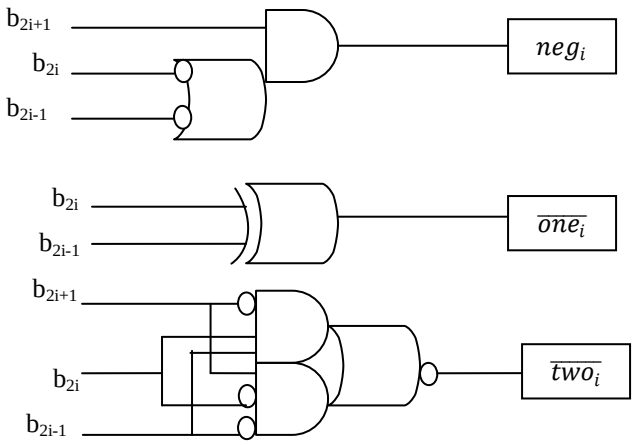
Fig.2. Proposed MBE partial product array for 8×8 multiplication

A. Proposed MBE Multiplier

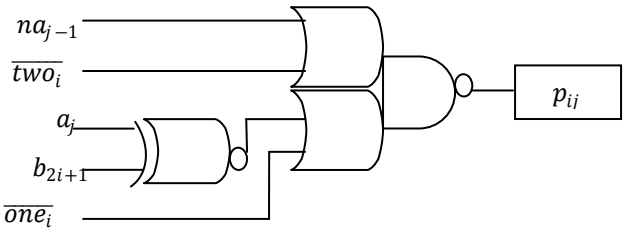
For MB recoding, at least three signals are needed to represent the digit set {-2,-1, 0, 1, 2}. Many different ways have been developed and Table I shows the encoding scheme proposed in [9] that is adopted to implement the proposed MBE multiplier. The Booth encoder and selector circuits proposed in [9] are shown in Fig.3 (a) and (b) respectively. Based on the recoding scheme and the approach proposed in [1], τ_{i0} and c_i in Fig.1 (d) can be derived from the truth table shown in Table II, as follows:

$$\tau_{i0} = one_i \,.\, a_0 = \overline{\overline{one_i} + \bar{a}_0} \qquad (3)$$

$$c_i = neg_i.(\overline{one_i} + \bar{a}_0) = \overline{\overline{neg_i} + one_i \,.\, a_0} \qquad (4)$$



(a)



(b)

Fig.3.MBE encoder and selector proposed in [9]

According to (3) and (4), τ_{i0} and c_i can be produced by one NOR gate and one AOI gate respectively. Moreover, they are generated no later than other partial product bits.

To further remove the additional partial product row $PP_{n/2}$ [i.e., PP4 in Fig.1 (b),] we combine the c_i for i=n/2-1 with the partial product bit p_{i1} to produce the new partial product τ_{i1} and a new carry d_i. Then, the carry d_i can be incorporated into the sign extension bit of PP_0. However, if τ_{i1} and d_i are produced by adding c_i and p_{i1}, their arrival delays

TABLE II

TRUTH TABLE FOR PARTIAL PRODUCT BITS IN THE PROPOSED PARTIAL PRODUCT ARRAY

b_{2i+1}	b_{2i}	b_{2i-1}	τ_{i0}	c_i
0	0	0	0	0
0	0	1	a_0	0
0	1	0	a_0	0
0	1	1	0	0
1	0	0	0	1
1	0	1	a_0	$\overline{a_0}$
1	1	0	a_0	$\overline{a_0}$
1	1	1	0	0

will probably be larger than other partial product bits. Therefore, we directly produce τ_{i1} and di for i=n/2-1 from A, B and the outputs of the Booth encoder (i.e.,neg_i, $two_i$and one_i), as shown in Table II where $\oplus$ and $\overline{\oplus}$ denote the Exclusive-OR and Exclusive-NOR operations, respectively. The logical expression of τ_{i1} and d_i can be written as

$$\tau_{i1} = one_i.\varepsilon + two_i.a_0 = \overline{(\overline{one_i} + \bar{\varepsilon}).(\overline{two_i} + \bar{a}_0)} \qquad (5)$$

$$d_i = \overline{\bar{b}_{2i+1} + a_0}.\left[(b_{2i+1} + a_1).(b_{2i} + a_1).(b_{2i} + b_{2i-1})\right] \qquad (6)$$

Where

$$\bar{\varepsilon} = \begin{cases} a_1, & if\ \overline{a_0.b_{2i+1}} = 0 \\ \overline{a_1}, & otherwise \end{cases} \qquad (7)$$

Since the weight of d_i is 2^n, which is equal to the weight of s_0 at bit position n, d_i can be incorporated with the sign extension bits $\bar{s_0}s_0s_0$ of PP_0. Let $\alpha_2\alpha_1\alpha_0$ be the new bits after incorporating d_i into $\bar{s_0}s_0s_0$; the relations between them are summarized in Table III. As can be seen in Table III the maximum value of $\bar{s_0}s_0s_0$ is 100 so that the addition of $\bar{s_0}s_0s_0$ and d_i .Therefore, the $\alpha_2\alpha_1\alpha_0$ is enough to represent the sum of $\bar{s_0}s_0s_0$ and d_i . According to Table III, α_2, α_1 and α_0 can be expressed as

$$\alpha_2 = \overline{(s_0.\bar{d}_i)} \qquad (8)$$

$$\alpha_1 = s_0.\overline{d_i} = \overline{\alpha_2} \qquad (9)$$

$$\alpha_0 = \overline{s_0 \oplus \overline{d_i}} \qquad (10)$$

TABLE III

TRUTH TABLE FOR NEW SIGN EXTENSION BITS

$\overline{s_0}$	s_0	s_0	d_0	α_2	α_1	α_0
1	0	0	0	1	0	0
1	0	0	1	1	0	1
0	1	1	0	0	1	1
0	1	1	1	1	0	0

The partial product array generated by the proposed approach for the 8×8 multiplier is shown in Fig.2. This regular array is generated by only slightly modifying the original partial product generation circuits and introducing almost no area and delay overhead.

IV. EXPERIMENTAL RESULTS

For comparison, we have implemented conventional MBE multiplier whose partial product arrays are generated. Accept for a few of partial product bits to regularize the partial product array, the other partial product bits are generated by using the similar method and circuits for the multiplier. These multipliers are modeled in Verilog HDL and synthesized by using Xilinx.

TABLE IV

EXPERIMENTAL RESULTS OF MBE MULTIPLIERS

n	Multiplier	Area (μm^2)	Delay (ns)	Power (µW)
8	Conventional	7311	4.19	728.3
	Proposed	5921	3.81	489.0

The implementation results including the hardware area, critical path delay and power dissipation for multipliers with n=8 are listed in Table IV. For example, the 8×8 multplier gives 11.32%,7.0% and 17.4% improvement over the conventional multiplier in area, delay and power reduction respectively.

V. CONCLUSION

In this brief, a simple approach has been proposed to generate a regular partial product array with fewer partial product rows, thereby reducing the area, delay and power of MBE multiplier. The proposed approach can also be extended to 16×16 and 32×32 multiplier.Experimental results have demonstrated that the proposed MBE multipliers can achieve significant improvement in area, delay and power consumption when compared with conventional multipliers.

REFRENCES

[1] W.-C.Yeh and C.-W.Jen,"High-Speed booth encoded parallel multiplier design,"IEEE Trans. Comput.,vol.49,no.7,pp.692-701,Jul.2000.

[2] J.-Y Kang and J.-L.Gaudiot," A simple high-speed multiplier design,"IEEE Trans.Comput.,vol.55,no.10,pp.1253-1258,Oct.2006.

[3] J.Fadavi-Ardekani,"M × N booth encoded multiplier generator using optimized Wallace trees,"IEEE Trans. Very Large Scale Integr.(VLSI) syst.,vol.1,no.2,pp.120-125,Jun.1993.

[4] O.Salomon,J.-M.Green, and H.Klar,"General algorithms for a simplified addition of 2's complement numbers,"IEEE J.Solid-State Circuit,vol.30,no.7,pp.839-844,Jul.1995.

[5] E.de Angel and E.E.Swartzlander,Jr.,"Low power parallel multipliers,"in Worlshop VLSI Signal Process.IX,1996,pp.199-208.

[6] A.A.Farooqui and V.G.Oklobdzija,"General data-path organisation of a MAC unit for VLSI implementation of DSP processors,"in Proc.IEEE Int.Symt.Circuits Syst.,1998,vol.2,pp.260-263.

[7] S.-F.Hsiao,M.-R.Jiang and J.-S.Yeh,"Design of high-speed low-power 3-2 counter and 4-2 compressor for fast multipliers," Electron.Lett.,vol.34,no.4,pp.341-342,Feb.1998.

[8] C.-H.Chang,J.Gu, and MZhang,"Ultra low-voltage low-power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits,"IEEE Trans. Circuits Syst.I,Reg.Papers,vol.51,no.10,pp.1985-1997,Oct.2004.

[9] Z.Huang and M.T.Ercegovac, "High-performance low-power left-to-right array multiplier design,"IEEE Trans. Comput.,vol.54,no.3,pp.272-283,Mar.2005.