An Area Efficient Redundant Binary (RB) Adder Using Modified RB Addition Rule

Jong-Wook Hong, Byung-In Mun, Tae-Young Lee and Yong-Surk Lee
Dept. of Electrical and Computer Engineering, Yonsei University

Abstract

The authors propose an area efficient redundant binary adder (RBA) using modified redundant binary (RB) addition rule. The proposed RBA reduces the number of transistors as compared with that of previously reported RBAs while maintaining fast speed.

I. Introduction

High speed multiplication is one of the key requirements for real-time signal processing systems. Because of their carry propagation free (CPF) addition characteristic [1], the multipliers using redundant binary (RB) arithmetic to sum up their binary partial products have been proved suitable for fast multiplication [2-3].

In RB multipliers, the RB adder tree which adds RB partial products (RBPPs) occupies a significant portion of the whole RB multiplier considering hardware complexity and critical path delay. Therefore, the optimization of the RBA has a significant effect on the overall RB multiplier performance. We propose here a new fast RBA with fewer transistors than the previously reported RBAs using modified RB addition rule.

II. Modified RB Addition Rule

Table 1 shows the conventional RB addition rule proposed by Takagi [1]. Takagi's RB addition rule determines a carry-out and intermediate sum by examining whether the both of the previous lower digits, x_{i-1} and y_{i-1}, are non-negative or not. If the both are non-negative, it means that there is no possibility of -1 carry occurrence, but possibility of 1 carry occurrence. The rule flexibly determines a carry-out and intermediate sum by the above examination using the redundancy of signed digit number system to eliminate continuous carry propagation.

To simplify internal logic of RBA, we propose a modified RB addition rule and it is shown in Table 2. The modified RB addition rule distinguishes the case that one of the previous lower digits is -1 and the other is 1 from the other cases because in the former cases, there is no possibility of -1 or 1 carry occurrence although they are classified as one of the possible candidates for -1 carry occurrence in the conventional RB addition rule.

III. Proposed RBA

To design RBA, we adopt one of the recently proposed RB recoding scheme [2]. Table 3 shows the mapping of each binary pair to corresponding values by the adopted RB recoding scheme. Table 4 shows the classification of all the possible input operand combinations corresponding to their carry-out and intermediate sum using the modified RB addition rule.

Table 5 shows the logic functions of the designed RBA derived from Table 2 and Table 4. The logic and circuit diagram of the designed RBA are given in Fig 1. As shown in Fig 1, the designed RBA is implemented by transmission gate (TG) logic. For efficiently buffering cascaded XOR gates, we use one of the recently reported XOR gates [4] at the first stage of the cascaded XOR gates.

IV. Result

The delay time and transistor count of the proposed RBA are compared with the previously reported RBAs [2-3] in Table 6. The delay time is measured by HSPICE simulation using the load condition of fan-out 1 and 0.6 ㎛ CMOS parameters.

V. Conclusion

A new fast RBA with the reduced number of transistors is proposed. The number of transistors of the proposed RBA is reduced more than 12 % as compared with that of the previous RBAs without degradation of its speed.

Reference

[1] N. Takagi, et. al , *IEEE Trans. Comput.*, vol. C-34, pp. 789-796, Sept. 1985.

[2] H. Makino, et al., *IEEE J. Solid-State Circuits*, vol. 31, no. 6, pp. 773-782, 1996.

[3] X. Huang, et. al., *IEEE Trans., Circuits Syst. I*, vol. 41, pp. 33-39, Jan. 1994.

[4] Jyh-Ming Wang, et. al., *IEEE J. Solid-state Circuits*, vol. 29, no. 7, pp. 780-786, 1994.

Table 1. conventional RB addition rule

x_iy_i	00	01	01	$0\bar{1}$	$0\bar{1}$	11	$\bar{1}1$	$1\bar{1}$
$x_{i-1}y_{i-1}$	-	neither is $\bar{1}$	at least one is $\bar{1}$	neither is $\bar{1}$	at least one is $\bar{1}$	-	-	-
c_i	0	1	0	0	$\bar{1}$	1	$\bar{1}$	0
u_i	0	$\bar{1}$	1	$\bar{1}$	1	0	0	0

Table 2. modified RB addition rule

x_iy_i	01	01	$0\bar{1}$	$0\bar{1}$	01	01	$0\bar{1}$	$0\bar{1}$
$x_{i-1}y_{i-1}$	except $1\bar{1}$ or $\bar{1}1$				when $1\bar{1}$ or $\bar{1}1$			
	neither is $\bar{1}$	at least one is $\bar{1}$	neither is 1	at least one is 1	$1\bar{1}$	$\bar{1}1$	$1\bar{1}$	$\bar{1}1$
c_i	1	0	0	$\bar{1}$	0	1	$\bar{1}$	0
u_i	$\bar{1}$	1	$\bar{1}$	1	1	$\bar{1}$	0	$\bar{1}$

Table 3. The value of an RB digit

(x_i, y_i)	value
(0, 0)	0
(0, 1)	-1
(1, 0)	1
(1, 1)	0

Table 4. Classification of the input operands

case	(x_i^+, x_i^-) (y_i^+, y_i^-)	c_{i-1}^-	Intermediate carry (c_i^+, c_i^-)	sum (u_i^+, u_i^-)
1	(0, 0) (0, 0) (0, 0) (1, 1)	any	(0, 0)	(0, 0)
2	(1, 1) (1, 1) (0, 0) (1, 1)	any	(1, 1)	(0, 0)
3	(0, 1) (1, 0)	any	(0, 0)	(0, 0)
4	(1, 0) (0, 1)	any	(1, 1)	(0, 0)
5	(0, 0) (1, 0) (1, 1) (1, 0) (1, 0) (0, 0) (1, 0) (1, 1)	1	(0, 0)	(1, 0)
		0	(1, 0)	(0, 1)
6	(0, 0) (0, 1) (1, 1) (0, 1) (0, 1) (0, 0) (0, 1) (1, 1)	1	(0, 1)	(1, 0)
		0	(0, 0)	(0, 1)
7	(1, 0) (1, 0)	any	(1, 0)	(0, 0)
8	(0, 1) (0, 1)	any	(0. 1)	(0, 0)

Table 5. Logic functions of the designed RBA

Proposed RBA

$$g_i = \overline{(x_i^+ \oplus x_i^-)} \oplus \overline{(y_i^+ \oplus y_i^-)}$$

$$c_i^- = \overline{(y_i^+ \oplus y_i^-)}x_i^- + \overline{(y_i^+ \oplus y_i^-)}y_i^-$$

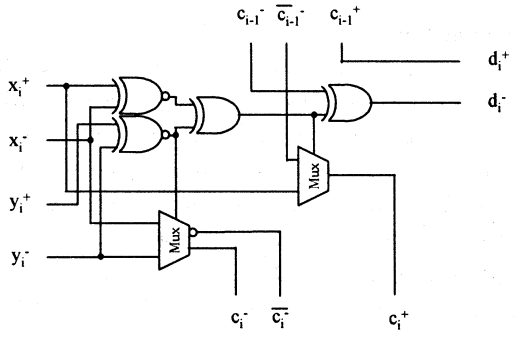
$$c_i^+ = g_i\overline{c_{i-1}^-} + \overline{g_i}x_i^+$$

$$u_i^+ = g_ic_{i-1}^-$$

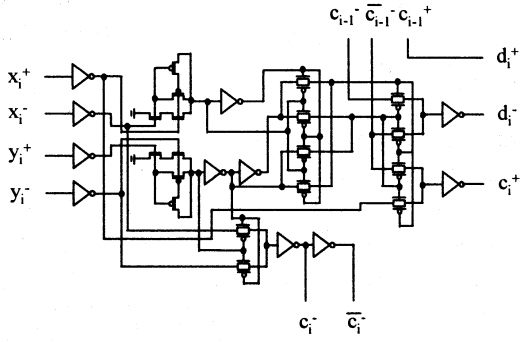
$$u_i^- = g_i\overline{c_{i-1}^-}$$

$$d_i^+ = c_{i-1}^+$$

$$d_i^- = g_i \oplus c_{i-1}^-$$



(a) logic diagram



(b) Circuit diagram

Fig 1. Logic and circuit diagram of the designed RBA

Table 6. performance comparison for different RBAs

RBAs	RBA [2]	RBA [3]	proposed RBA
delay [ns]	1.21	1.30	1.24
Tr count	56	62	50