A Nonspeculative Maximally Redundant Signed Digit Adder

Ghassem Jaberipur and Saeid Gorgin

Department of Electrical and Computer Engineering, Shahid Beheshti University and
School of Computer Science, Institute for Studies in Theoretical Physics and Mathematics (IPM), Tehran, Iran
{Jaberipur,Gorgin}@sbu.ac.ir

Abstract. Signed digit number systems provide the possibility of constant-time addition, where inter-digit carry propagation is eliminated. Carry-free addition for signed digit number systems is primarily a three-step process. However, the special case of maximally redundant signed digit number systems leads to more efficient carry-free addition. This has been previously achieved by speculative computation of digit-sum values using three parallel adders. We propose an alternative nonspeculative addition scheme that computes the transfer values through a fast combinational logic. The proposed carry-free addition scheme uses a combinational logic, to compute the transfer digit, and the equivalent of two adders. The simulation and synthesis of the two previous works and this work based on 0.13 μm CMOS technology shows that the proposed circuit operates faster and has a lower product of delay × power.

Keywords: Computer arithmetic, Carry-free addition, Signed-digit number system, Maximal redundancy.

1 Introduction

Addition is the basic computer arithmetic operation. Traditional ripple-carry adders are very slow due to the long chain of carry-propagation logic. The latency of an n-digit carry-ripple adder is linearly depending on n (i.e., O (n)) [1]. Carry-accelerating techniques used, for example, in carry look-ahead adders [2] or carry select adders [3] improve the order of latency to O(log n) and O($\sqrt{n}$), respectively. Faster adders are not possible if the sum, as usual, is to be represented in a conventional nonredundant format [4]. However, constant-time (i.e., O(1)) adders may be envisaged if the sum is allowed to be represented in a redundant format [5].

In a radix-r redundant number system, each digit may assume values from a redundant digit set [α, β] whose cardinality is greater than r [6]. The special case of balanced digit set [–α, α] was introduced in the pioneering work of Avizeinis [7] as the signed digit (SD) number systems. The conventional three-step carry-free addition algorithm for SD numbers (see Section 2) has been implemented in hardware, based on different approaches, for improved performance. For example, Fahmy and Flynn offer a maximally redundant SD (MRSD) adder that effectively parallels the three steps of the conventional algorithm [8]. They use three SD adders to speculatively

H. Sarbazi-Azad et al. (Eds.): CSICC 2008, CCIS 6, pp. 235–242, 2008.

compute three digit-sum values with regards to three different transfer-values. Similar result is reported in [9] based on a different approach for transfer computation.

In this paper, we present a new maximally redundant SD adder that nonspeculatively computes the digit-sums and shows more efficiency compared to previous speculative schemes. Here is a roadmap to the rest of the paper. A background on SD number systems and conventional carry-free addition algorithm is provided in Section 2, the work of [8] and [9] are reviewed in Section 3, we offer, in Section 4, the new nonspeculative addition scheme, Section 5 provides the simulation results, and finally we draw our conclusions in Section 6.

2 Signed Digit Number Systems

In the conventional nonredundant number systems, cardinality ξ of the digit set is equal to the radix r (e.g., the digit set [0, 1] for radix 2 or [0, 9] for radix 10). Notwithstanding the conventional radix-complement or diminished-radix-complement number systems, in order to allow negative numbers, one could think of digit sets with signed values (e.g., radix-5 and radix-16 nonredundant digit sets [–2, 2] and [–8, 7], respectively). Allowing $\xi > r$, leads to redundant number systems, where some values may be redundantly represented by more than one digit combination.

Example 1 (decimal redundancy): Consider the decimal digit set {0, 1,···, 9, A, B}, where digits A and B worth 10 and 11, respectively and $\xi = 12 > r = 10$. In this redundant decimal number system, the 3-digit number 110 may also be represented as the 2-digit number AA ($10 \times 10 + 10 = 110$). ◀

The signed digit (SD) number systems, introduced by Avezienis [7], represent a special case of redundant number systems, where the radix-r digit set is $[-\alpha, \alpha]$, and $\alpha \geq r/2$ that leads to $\xi = 2\alpha + 1 > r$. The most useful property of redundant number systems is the possibility of carry-free addition, where the carry propagation chain is limited to a few number of digits [5]. This chain is only one digit long for SD numbers in case of $r \geq 3$ and $\alpha \geq (r+1)/2$ [7], where the carry generated in any position i will not propagate beyond position $i + 1$. The conventional three-step carry-free addition algorithm for SD numbers is presented below as Algorithm 1.

Algorithm 1 (Carry-free SD addition):

Input: Two n-digit radix-r SD numbers $X = x_{n-1}\ldots x_0$ and $Y = y_{n-1}\ldots y_0$, where $-\alpha \leq x_i, y_i \leq \alpha$ for $0 \leq i \leq n-1$.

Output: An $n+1$-digit radix-r SD number $S = s_n\ldots s_0$

Compute the n-digit position-sum as a radix-r SD number $P = p_{n-1}\ldots p_0 = X + Y$, by digit-parallel computation of $p_i = x_i + y_i$ for $0 \leq i \leq n-1$.

Decompose pi to transfer ti+1 and interim sum wi such that $-\alpha + 1 \leq w_i \leq \alpha - 1$ for $0 \leq i \leq n-1$, $p_i = w_i + r \times t_{i+1}$, and $t_{i+1} = -1$, 0, and 1 for $p_i \leq -\alpha$, $-\alpha < p_i < \alpha$, and $p_i \geq \alpha$, respectively.

Compute $s_i = w_i + t_i$, for $0 \leq i \leq n-1$, where no new transfer will be generated by this final addition, and $s_n = t_n$. ◀

Example 2 (Decimal Carry-free addition): Consider the decimal SD digit set [–7, 7], where $\xi = 15 > r = 10$. Fig. 1 illustrates the application of Algorithm 1 on two 4-digit decimal SD numbers. ◄

	i	4	3	2	1	0
	x_i		2	3	−5	4
	y_i		5	6	−6	2
Step I	p_i		7	9	−11	6
Step II	w_i		−3	−1	−1	6
	t_i	1	1	−1	0	
Step III	s_i	1	−2	−2	−1	6

Fig. 1. A decimal SD addition

Each of the Steps I and III of Algorithm 1 contains a digit addition. Also Step II involves a digit comparison of p_i with α. Reducing the number of these three digit operations, if possible, leads to faster carry-free addition. In the next Section two such approaches, which have recently appeared in the literature, are reviewed.

3 Speculative MRSD Addition Schemes

The radix of choice for a redundant number system, besides the special case of radix-10, is practically a power-of-two such that $r = 2^h$. Therefore, given that $\xi > r$, the number of bits for encoding each radix-2^h digit is at least h+1. However, an h+1-bit two's complement encoding of a SD $\in [-\alpha, \alpha]$, allows α to be up to $2^h - 1$ ($\xi = 2\times\alpha+1 \le 2^{h+1} \Rightarrow \alpha \le 2^h - 1$) corresponding to the maximally redundant SD number system. In this case there is only one invalid h +1-bit digit-value due to -2^h. On the other extreme, $\alpha = 2^{h-1}$ (i.e., the lowest α that meets the Avezienis lower bound $(r+1)/2$), corresponds to the minimally redundant SD number system with $2^h - 1$ invalid h +1-bit digit-value in the intervals $[-2^h, -2^{h-1}-1]$ and $[2^{h-1}+1, 2^h-1]$. One can intuitively expect that the minimum number of invalid h+1-bit digit-values in the maximally redundant case lead to the most efficient SD adder.

Two speculative MRSD adders due to [8] and [9] are depicted in Figs. 2 and 3, respectively. Both schemes use h+1-bit two's complement encoding of the signed digits and rely on triple active hardware redundancy on concurrently computing p_i–1, p_i, and p_i+1 in anticipation of the coming transfer value from position i–1. Fahmy and Flynn observed that the h least significant bits of the interim sum w_i are the same as the corresponding bits of p_i and the MSB of w_i and the transfer bits may be computed by combinational logic. However, the other work is based on comparing p_i with 2^{h-1}, instead of comparison with α in Step II of Algorithm 1. Note that, in Figures 2 and 3

the number of h+1-bit operations in the critical delay path is two and one, respectively. The speculative approach with triple hardware redundancy (i.e., three parallel adders), as followed in these designs, is probably the most straight forward approach. However, we present, in the next section, a nonspeculative MRSD adder based on very fast computation of the transfer digit through a combinational logic.

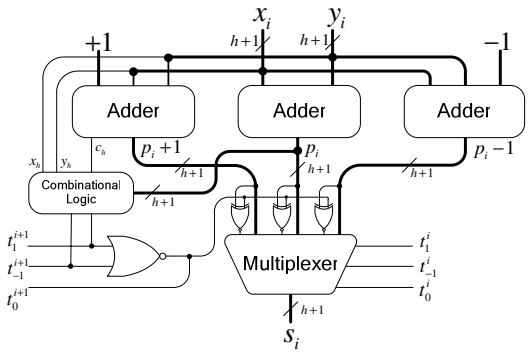

Fig. 2. Position i of MRSD adder of [8]

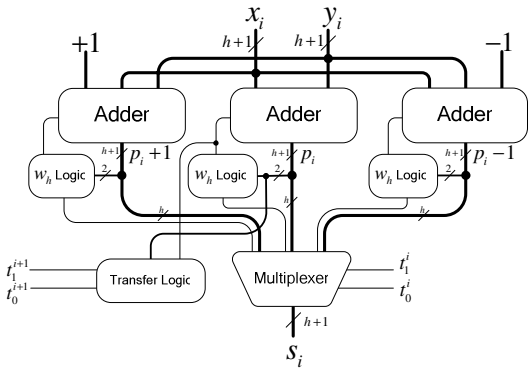

Fig. 3. Position i of MRSD adder of [9]

4 New MRSD Addition Scheme

In SD carry-free addition, the interim sum digit w_i and transfer t_{i+1} can be expressed directly in terms of the radix-2^h digits x_i and y_i (e.g., as functions $w_i = \omega(x_i, y_i)$ and $t_{i+1} = \tau(x_i, y_i)$). However, a direct implementation of these functions in hardware is not practical due to the large number of input bit-variables (i.e., $2\times(h+1)$ bits). Alternatively, and for the sake of efficiency, the strict comparison of p_i and α may be relaxed in the Step II of Algorithm 1. In other words, for a given value of p_i, there may be more than one set of corresponding valid values for w_i and t_{i+1}. Figure 4 illustrates such p_i-values as gray regions, where valid intervals for different transfer values overlap.

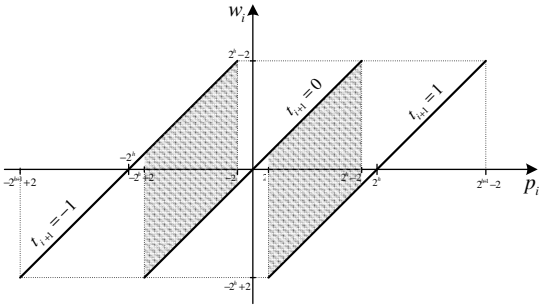

Fig. 4. The overlapping regions of valid values for t_{i+1}

One may take advantage of this imprecision (i.e., choice of different transfer values for a given p_i value), by a careful selection of alternative t_{i+1}-values, to prevent dependency of t_{i+1} on each and all the constituent bits of x_i and y_i. In particular, the case of minimum dependency (i.e., only on X_i^h and Y_i^h, the most significant bits of x_i and y_i) may lead to very fast computation of the transfer. Table 1 shows that t_{i+1} may, indeed, be defined as a function of just X_i^h and Y_i^h, except for five instances of (x_i, y_i) values, where the proposed t_{i+1} leads to an invalid w_i. For example, in the first row of Table 1, $t_{i+1} = 1$ holds for $x_i \geq 0$ and $y_i \geq 0$ except for three cases, where the resultant w_i is not valid (i.e., $w_i \leq -\alpha = -2^h + 1$).

As another example, the strict comparisons of Step II of Algorithm 1 would set $t_{i+1}=0$ for $-2^h + 1 < p_i < 2^h - 1$. However, the transfer value chosen by Table 1 in some of such cases (e.g., for $x_i = y_i = 1$) is 1.

Table 1. Minimum dependency of ti+1 and the exceptions

$X_i^h\ Y_i^h$	Range of x_i and y_i	t_{i+1}	w_i	Exceptions on (x_i, y_i) pair
0 0	$x_i \geq 0, y_i \geq 0$	1	$-2^h + p_i$	(0, 0), (0, 1), and (1, 0)
0 1	$x_i \geq 0, y_i < 0$	0	p_i	$(0, -2^h+1)$
1 0	$x_i < 0, y_i \geq 0$	0	p_i	$(-2^h+1, 0)$
1 1	$x_i < 0, y_i < 0$	–1	$2^h + p_i$	None

To take care of the exceptions in Table 1 let $x_i = X_i^h x_i^{h-1} \ldots x_i^0$, $y_i = Y_i^h y_i^{h-1} \ldots y_i^0$, $p_i = P_i^{h+1} p_i^h \ldots p_i^0$, $w_i = W_i^h w_i^{h-1} \ldots w_i^0$ and $t_{i+1} = T_{i+1}^1 t_{i+1}^0$. Equation sets (1) and (2), blow, can be driven from Table 1, where $\varphi = \overline{x_i^{h-1} + \cdots + x_i^1}\ \overline{y_i^{h-1} + \cdots + y_i^1}\ \overline{x_i^0 y_i^0}$ is a flag indicating the exceptions. Note that x_i, y_i, p_i, w_i, and t_{i+1} are two's complement numbers, where the MSB is distinguished by uppercase variables. Furthermore it is easy to see from Table 1 that $W_i^h = P_i^h$ and $t_{i+1}^0 = 0$ in the two middle rows, $W_i^h = \overline{P_i^h}$ and $t_{i+1}^0 = 1$ in the first and last rows, and the other bits of w_i are the same as corresponding bits of p_i. The reason is that adding ± 2h to pi affects only the bit in position h.

$$T_{i+1}^{1} = X_i^h Y_i^h + \varphi\left(X_i^h + Y_i^h\right), \quad t_{i+1}^{0} = \overline{X_i^h \oplus Y_i^h \oplus \varphi} \tag{1}$$

$$W_i^h = P_i^h \oplus t_{i+1}^{0}, w_i^j = p_i^j, \text{ for } 0 \le j \le h-1 \tag{2}$$

In spite of the latter improvement regarding the implementation of Algorithm 1, there are still two h +1-bit addition steps; namely $p_i = x_i + y_i$ and $s_i = w_i + t_i$. However, due to the following observations the second addition operation may commence as soon as t_i is available.

Using Equation-set (1), t_i is available prior to termination of computation of p_i=$x_i + y_i$. The bits of w_i, except for the MSB, are available as soon as the corresponding bits of p_i are ready.

Fig. 5 illustrates the new MRSD adder based on the above results. The bold line shows the critical delay path, which passes through exactly h full adder cells as well as a half adder and the transfer logic whose details are depicted in Fig. 6. Note that there is also a combinational logic and a multiplexer in the critical delay path of the two previous designs [8, 9]. Therefore, shorter latency for all values of h, with respect to [8], and moderate values of h (e.g., h = 4), with respect to [9], is expected for the critical delay path of Fig. 5. This is confirmed by the outcomes of simulation and synthesis provided in the next section.

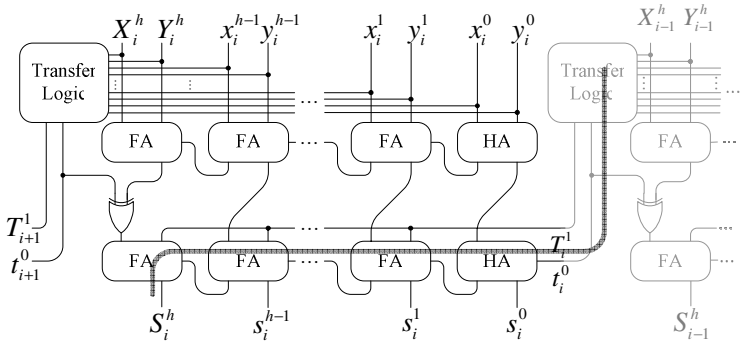

Fig. 5. The new one-step MRSD adder

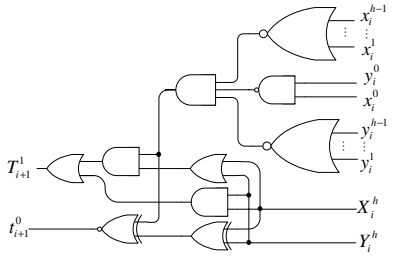

Fig. 6. The transfer logic

5 Results of Simulation and Synthesis

We have implemented the logic of Fig.5 and also those of Figs. 2 and 3, for $h = 4$, in VHDL and run exhaustive tests to ensure correctness. The 5-bit adder in the critical delay path of Fig. 5 and the similar ones in Figs. 2 and 3 are replaced by fast carry look-ahead logic. Subsequently, all the three single digit MRSD adders were synthesized in 0.13 μm CMOS technology. The simulation results are presented in Table 2, where the three designs are compared in terms of latency, power and area. There is also a PDP (product of delay × power) column. All the figures in the bottom row show that the proposed nonspeculative SD adder outperforms the previous works.

Table 2. Simulation results for single digit MRSD adders with h=4

MRSD adder	Delay (ns)	Power (mW)	Area (μm^2)	PDP
Fig.2 [8]	1.30	0.97	985	1.26
Fig. 3 [9]	1.05	1.45	1304	1.52
Fig. 5 (nonspeculative)	0.89	0.77	823	0.68

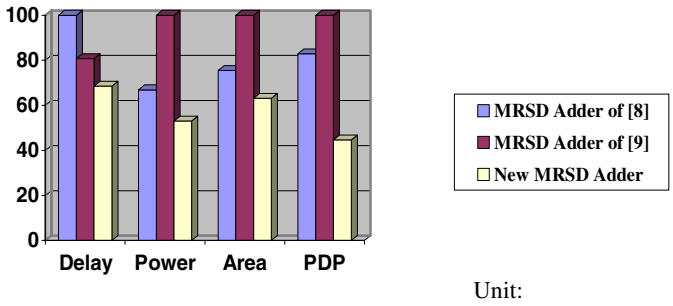

Fig. 7. Illustration of the comparison of three MRSD adders

6 Conclusions

Carry-free addition of radix-2^h signed digit numbers is, in general, a three-step process, where the latency of each step is roughly as long as that of an h +1-bit adder. We propose a nonspeculative maximally redundant radix-2^h signed digit addition scheme with only one h +1-bit digit-adder and an h+1-input combinational logic in the critical delay path. This work is compared with two previously published designs for speculative MRSD adders that use three h+1-bit digit-adders in parallel for improved performance. All the three designs have been simulated by a synthesis tool based on 0.13 μm CMOS technology, where the results are illustrated in Figure 7. Comparing these results with the best of the two previous works, 15.5% less delay is achieved with respect to [9], and 20% less power dissipation, 16.5% less area, and 46% less PDP are the advantages over [8].

Further research is ongoing towards alternative nonspeculative MRSD addition schemes with better performance.

Acknowledgement

This work has been funded in part by grant # from Shahid Beheshti University, and in part by grant # CS1386-3-01 from IPM.

References

1. Parhami, B.: Computer Arithmetic: Algorithms and Hardware Designs, Oxford (2000)
2. Doran, R.W.: Variants of an Improved Carry Look-Ahead Adder. IEEE Trans. Computer 37(9), 1110–1113 (1988)
3. Sklansky, J.: Conditional-Sum Addition Logic. IRE Trans. Elec. Comp. 9(2), 226–231 (1960)
4. Winograd, S., Watson, T.J., Heights, Y.: On the Time Required to Perform Addition. Journal of the ACM (JACM) 12(2), 277–285 (1965)
5. Parhami, B.: Generalized Signed-Digit Number System: A Unifying Framework for Redundant Number Representation. IEEE Trans. on Computer 39(1), 89–98 (1990)
6. Jaberipur, G., Parhami, B.: Stored-Transfer Representations with Weighted Digit-Set Encodings for Ultrahigh-Speed Arithmetic. IET Circuits, Devices, and Systems 1(1), 102–110 (2007)
7. Avizienis, A.: Signed-digit number representations for fast parallel arithmetic. IRE Trans. on Electronic Computers EC-10, 389–400 (1961)
8. Fahmy, H., Flynn, M.J.: The Case for a Redundant Format in Floating-point Arithmetic. In: Proc. 16th IEEE Symposium Computer Arithmetic, pp. 95–102. IEEE Computer Society, Los Alamitos (2003)
9. Jaberipur, G., Ghodsi, M.: High Radix Signed Digit Number Systems: Representation Paradigms. Scientia Iranica 10(4), 383–391 (2003)