Indonesian Journal of Electrical Engineering and Computer Science
Vol. 4, No. 3, December 2016, pp. 665 ~ 669
DOI: 10.11591/ijeecs.v4.i3.pp665-669

Reverse Conversion of Signed-Digit Number Systems: Transforming Radix-Complement Output

Madhu Sudan Chakraborty[1], Abhoy Chand Mondal*[2]
[1]Department of Computer Science, Indas Mahavidyalaya, PO: Indas, Bankura, WB, India 722205
[2]Department of Computer Science, Burdwan University, Burdwan, WB, India 713104
*Corresponding author, e-mail: abhoy_mondal@yahoo.co.in

Abstract

Although the speed advantage of using signed – digit number systems seemed to have been reduced significantly by reverse conversion owing to the carry – propagation, in this paper, it was shown that if typical reverse conversion algorithms were employed for signed – digit number systems, then no further carry propagation needed to transform the output from radix – complement form to other conventional forms. As a result the instantaneous delay caused by the reverse conversion of signed – digit number systems might be compensated by speed gain at later stages.

Keywords: *signed-digit number systems, reverse conversion, carry propagation, conventional representation, digit-parallel transformation*

1. Introduction

Signed-digit number system (SDNS) as introduced in [1], is an unconventional number system which is defined on a digit-set containing zero, positive and negative integers [2]. Basically the most important feature of SDNS is its ability to support carry-free/digit-parallel addition and complementation which act as the key to speed up many common arithmetic operations [2]. This is why SDNSs have found scope for applications over a wide area: from general purpose microprocessor to digital signal processing (DSP). SDNSs offer regularity in circuit design and seem to be suitable for hardware implementation [2]. In addition, in line with quest for low power electronic circuits [2, 3], investigations have shown that VLSI implementations of some arithmetic operations using SDNSs may consume lower energy [4-6].

However, as the accustomed bus architectures for DSP and operations of standard peripheral devices are still based on two's-complement/ natural binary number representation, conversion is required from unconventional form to the conventional forms [2], known as the reverse conversion. Reverse conversion has been widely viewed as a major performance bottleneck for SDNSs [7-10], like that for any other unconventional number systems [2], [11]. Even the speed advantage of using SDNDs seems to have reduced significantly as there is no absolutely carry-free scheme for reverse conversion of SDNSs [9], [12-13]. The problem of carry propagation may persist even after the instantaneous output of reverse conversion is generated. Commonly, the instantaneous output of reverse conversion of SDNS appears in radix-complement form (RCF) and in the literature of computer arithmetic reverse conversion of SDNS is ordinarily viewed as the SDNS-to-RCF transformation problem [9]. However, as the instantaneous radix-complement output often needs to be converted to sign-magnitude form (SMF) and sometimes even to diminished radix-complement form (DRCF) either for further internal processing or for user-interface, the implications of reverse conversion for the speed/ performance of SDNSs need to be studied as a whole in terms of SDNS – to – RCF – to – SMF/ DRCF transformation. The problem is that even after expressing the numbers in RCF, the traditional RCF to SMF/DRCF conversion method causes further carry-propagation. Obviously, the straightforward transformation of the output of reverse conversion from RCF to SMF/ DRCF may further reduce the computing speed of the system. Some formulae to directly convert the output of reverse conversion of SDNS without carry-propagation were introduced in [14]. However, neither the methodological realization of the proposed formula was shown nor it was attempted to be proved in [14]. In addition even if being found correct, the conversion formula

Received September 7, 2016; Revised October 25, 2016; Accepted November 20, 2016

proposed in [14] is applicable only when the instantaneous output of reverse conversion of SDNS appears in SMF, which is obviously not the common instantaneous output [9].

In the following it will be shown that if typical reverse conversion algorithms are employed for SDNS, then no further carry propagation is needed to transform the output from RCF to SMF/ DRCF. As a result the instantaneous delay caused by reverse conversion of SDNSs may be compensated by speed gain at later stages. In this regard, the rest of the paper is organized as follows: In section 2 initially a 1-bit conversion tag (OBCT) is defined on each digit position of the conventional radix-complement input. OBCTs are computed in digit-serial manner from least-significant-position to the most-significant-position and then a scheme is developed for transforming conventional numbers from RCF to SMF using OBCTs in digit-parallel manner. In section 3 it is shown that as OBCTs are essentially pre-computed for typical reverse conversion scheme for SDNS, the RCF output can be straightly transformed to the other conventional representations, SMF and DRCF, without further carry propagation. In section 4 the proposed scheme is explained with an example. Finally, the proposed work is concluded with section 5.

2. Conversion of Conventional Numbers from RCF to SMF

Let $X1 = x1_{n-1}x1_{n-2}........x1_0$ $(n \geq 2)$ be an ordinary radix – r number represented in RCF.

2.1. Proposed Scheme

A conversion scheme to transform X1 into the equivalent SMF, say Y, having the same radix and equal number of digits as $Y = y_{n-1}y_{n-2}........y_0$ is proposed:

1. OBCT ($s1_i$) corresponding to each $x1_i$ is defined as:
1.1. Initially: $s1_{-1} = 0$
1.2. For $0 \leq i \leq n-2$ do
1.2.1. If $x1_i > 0$ then $s1_i = 1$
1.2.2. Otherwise, $s1_i = s1_{i-1}$
1.3. Compute: $s1_{n-1}$ as:
1.3.1. If $x1_{n-1} = 0$ then $s1_{n-1} = 0$
1.3.2. Otherwise, $s1_{n-1} = 1$
2. For $0 \leq i \leq n-2$, y_i digits are computed as:
   2.1. If $s1_{n-1} = 0$ then $y_i = x1_i$
   2.2. If $s1_{n-1} = 1$ and $s1_{i-1} = 0$ then $y_i = (-x1_i) \bmod r$
   2.3. If $s1_{n-1} = 1$ and $s1_{i-1} = 1$ then $y_i = r - 1 - x1_i$
3. Do:
   3.1. If $s1_{n-2} = 0$ and $s1_{n-1} = 1$ then output: “Overflow”
   3.2. Otherwise, set: $y_{n-1} = x1_{n-1}$
4. Stop

2.2. Proof

If $X1 \geq 0$ then $x1_{n-1} = 0$; otherwise, $x1_{n-1} = r - 1$. When $X1 \geq 0$, $Y = x1_{n-1}\ x1_{n-2}........x1_0$ where $x1_{n-1} = 0$. Obviously, the proposed conversion scheme works correctly.

When $X1 < 0$, its magnitude can be denoted as: $|X1| = W + 1$ such that $W = w_{n-2}....w_i....w_0$ where

$$w_i = \overline{r-1} - x1_i \ \forall\ i \in [0, n-2].$$

$$\text{Let } Z = Y1 - W \tag{1}$$

where $Y1 = y_{n-2}y_{n-3}.....y_0$.

For proving the correctness of the proposed conversion scheme when $X1 < 0$, it is to be shown that:

$$Z = 1 \tag{2}$$

Considering all possible cases, the result of digit-by-digit computations of Z as defined in (1) from least-significant-digit to most-significant-digit is represented in Table 1 $\forall\ i \in [0, n-2]$ using the following notations:

b_i = borrow forwarded from i^{th} position in Z
dz_i = digit at i^{th} position of Z
D/C means don't care condition and N.Z means non zero.
Initially i = 0, $b_{-1} = 0$.

Table 1. Digit-Serial Computing for Z as defined in (1)

Case No	Inputs			Outputs			
	$s1_{i-1}$	b_{i-1}	$x1_i$	y_i	$s1_i$	$(y_i - w_i) - b_{i-1}$	
						dz_i	b_i
1	0	0	0	0	0	1	1
2	0	0	N.Z	$r - x1_i$	1	1	0
3	0	1	0	0	0	0	1
4	0	1	N.Z	$r - x1_i$	1	0	0
5	1	0	D/C	$r - 1 - x1_i$	1	0	0

Table 1 shows that $0 \leq y_i \leq r - 1 \; \forall \; i \in [0, n - 2]$. As presented in Table 1, computing for Z starts with either case 1 or case 2 and then goes through case 3, case 4 or case 5. In this connection, all possible execution sequences (APES) are shown in Figure 1 as an APES graph.

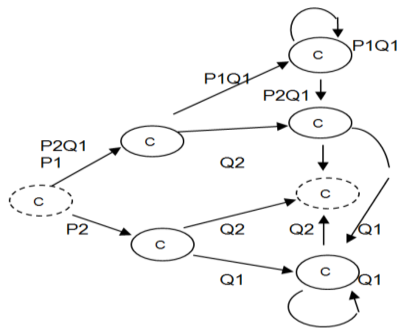

Figure 1. APES Graph Defined on Table 1

In the APES graph vertex Cj represents case no j as in Table 1 $\forall$ j $\in$ [1, 5]. In addition, in order to represent the flow of control more precisely without any loss of generality, vertex C0 and vertex C6 have been introduced to denote the unique start case and stop case respectively. However no computing is done both at C0 and C6. The directed edge Cm → Cn labeled with Pj, Qk or PjQk $\forall$ j, k $\in$ [1, 2] refers a transition to switch the control from vertex Cm to vertex Cn if condition Pj or Qk or both are satisfied respectively where:

P1: Is $x1_i = 0$? ; P2: Is $x1_i \neq 0$? ; Q1: Is $i < n - 2$?; Q2: Is $i \geq n - 2$? Any transition originated at vertex C0 or heading to vertex C6 keeps i unchanged and all other transitions increase i by 1. Then computations as per table 1 is performed and the control is switched to the next vertex through the matching transition. It may be noted that as using radix = r and number of digits = n, the number $-r^{n-1}$ can be correctly represented in RCF but not in SMF, $X1 = -r^{n-1}$ (where $s1_{n-1} = 1$ and $s1_{n-2} = 0$) is prohibited as shown in step 3. Consequently, transitions C1→C6 and C3→C6 are disallowed. Therefore for any valid input $dz_0 = 1$, $dz_i = 0 \; \forall \; i \in [1, n - 2]$ and $b_{n-2} = 0$ regardless of the execution sequence. It means, even when X1 < 0, Z = 1. Obviously the proposed scheme works correctly.

3. Implications of the Proposed Scheme in the Context of Typical Reverse Conversion

Let $D = d_{n-1}d_{n-2}....d_0$ be a radix – r signed – digit number defined on any valid digit set. OBCT (t_i) corresponding to each d_i is defined as:

1. Initially: $t_{-1} = 0$
2. For $0 \leq i \leq n - 1$:

2.1. If $d_i \neq 0$ then $t_i = 1$;

2.2. Otherwise, $t_i = t_{i-1}$.
As expressed in section 2 let X1 be the equivalent RCF of D and signal $s1_i$ is defined corresponding to $x1_i$, $\forall$ i ϵ [0, n – 1].

3.1. Corollary 1: $t_i = s1_i$ ∀ i ϵ [0, n – 1]

If possible assume that $t_i \neq s1_i$ for some i ϵ [0, n – 1] and let j be the smallest value of i such that $t_j \neq s1_j$. It means either $t_j = 0$ and $s1_j = 1$ or $t_j = 1$ and $s1_j = 0$.

Case 1: When $t_j = 0$ and $s1_j = 1$

$t_j = 0$ means $t_{j-1} = 0$, $t_{j-2} = 0$, ..., $t_{-1} = 0$

As j be the smallest value where t_j and $s1_j$ mismatches, $t_{j-1} = 0$ implies $s1_{j-1} = 0$. It means a j – digit partial signed – digit number as $d_{j-1}d_{j-2}....d_0$ who's most significant digit is 0 and even all other digits are 0 is equivalent to a j-digit partial radix-complement number as $x1_{j-1}x1_{j-2}....x1_0$ who's most significant digit is non-zero and all other digits are 0, which is a contradiction. Obviously, case 1 does not hold.

Case 2: When $t_j = 1$ and $s1_j = 0$

Proceeding similar to case 1 it can be shown that case 2 does not hold.

Clearly $t_i = s1_i$ ∀ i ϵ [0, n – 1]

Suppose that $X2 = x2_{n-1}x2_{n-2}........x2_0$ (n ≥ 2) denotes the SMF of a given number whose RCF is given by X1. Let OBCT ($s2_i$) is defined corresponding to $x2_i$ ∀ i ϵ [0, n – 1] as below:

1.1. Initially: $s2_{-1} = 0$
1.2. For $0 \leq i \leq n - 2$:
1.2.1. If $x2_i > 0$ then $s2_i = 1$
1.2.2. Otherwise, $s2_i = s2_{i-1}$
1.3. Compute: $s2_{n-1}$ as:
1.3.1. If $x2_{n-1} = 0$ then $s2_{n-1} = 0$
1.3.2. Otherwise, $s2_{n-1} = 1$

3.2. Corollary 2: $s1_i = s2_i$ ∀ i ϵ [0, n – 1]

Proceeding similar to corollary 1, corollary 2 can be proved.

Some algorithms for reverse conversion of SDNSs are based on direct or indirect sign-detection of partial signed-digit numbers [8-9], [14]. Let p_i denotes the sign of $(i+1)^{th}$ partial signed-digit number i. e. $d_id_{i-1.......}d_0$ ∀ i ϵ [0, n – 1]. In this connection, positive, negative and zero sign may be represented as 01, 11 and 00 respectively [15], [16]. Clearly, for both positive and negative sign the least significant bits (LSBs) are 1 whereas for zero the LSB is 0. So the LSBs for signs of partial signed-digit numbers can serve as t_i signals as presented in this section.

In this paper, it has been shown that the output of reverse conversion of SDNS can be transformed from RCF to SMF in digit-parallel manner. As a number represented in conventional form can be transformed from SMF to DRCF merely by complementing each digit excluding the sign-digit in parallel, the output of reverse conversion can also be transformed from RCF to DRCF in digit-parallel manner.

4. Example

For a given binary signed-digit number D = $\bar{1}010\bar{1}$ the instantaneous output of reverse conversion in two's-complement form [2], [9] is generated as: X1 = 10011. Suppose that X1 is to be further converted into binary SMF. Assume that some reverse conversion scheme based on sign-detection of partial signed-digit numbers has been employed [8-9], [14] that gives: $p_{-1} = 0$, $p_0 = \bar{1}$, $p_1 = \bar{1}$, $p_2 = 1$, $p_3 = 1$, $p_4 = \bar{1}$. Obviously $s_{-1} = 0$, $s_0 = 1$, $s_1 = 1$, $s_2 = 1$, $s_3 = 1$, $s_4 = 1$. Then two's-complement to binary SMF conversion in digit-parallel mode gives: $y_0 = 1$, $y_1 = 0$, $y_2 = 1$, $y_3 = 1$, $y_4 = 1$. Therefore Y = 11101 is the required binary SMF, obtained from the- complement equivalent of the given binary signed-digit number without further carry propagation.

5. Conclusion

Signed-digit number system [1] is well known for supporting high-speed computations and in this context carry-free addition and complementation provides the basic motivation [2]. However, being an unconventional number system, signed-digit number system ultimately needs reverse conversion [2] that necessarily involves carry propagation [9], [12-13]. Commonly the instantaneous output of reverse conversion appears in radix-omplement form [9]. Although the traditional method for conversion from radix-complement form to the other conventional forms needs further carry propagation, in this paper, it has been shown that typical reverse conversion schemes for signed-igit number system can be extended for performing carry-free conversion of instantaneous radix-complement output into other conventional forms. This feature of typical reverse conversion schemes for signed-digit number system not only prevents the occurrence of the situation which otherwise may pave an way for cumulated delays resulted by more than one carry propagation but also attempts to compensate the instantaneous delay caused by carry propagation in reverse conversion.

References

[1] Avizienis A. Signed-Digit Number Representation for Fast Parallel Arithmetic. *IRE Transactions on Electronic Computers.* 1961*;* 10(3): 389-400.
[2] Parhami B. Computer Arithmetic: Algorithms and Hardware Design. First Edition. Oxford: University Press. 2009.
[3] Al A, Reaz MBI, Jalil J, Ali MABM. An Improved A Low Power CMOS TIQ Comparator Flash ADC. *Indonesian Journal of Electrical Engineering.* 2014; 12(7): 5204-5210.
[4] Smitha KG, Fahmy AH, Vinod AP. *Redundant Adders Consume Less Energy*. Proceedings of IEEE APC on Circuits and Systems. Singapore. 2006; 1: 422-425.
[5] Crookes D, Jiang M. Using Signed - Digit Arithmetic for Low Power Multiplication. *Electronics Letters.* 2007*;* 43(11): 613-614.
[6] Phatak DS, Kahle S, Kim H, Lue J. *Hybrid Signed Digit Representation for Low Power Arithmetic Circuits*. Proceddings of Low Power Workshop in Conjunction with ISCA. Barcelona. 1998.
[7] He Y, Chang CH. A Power-Delay Efficient Hybrid Carry - Lookahead/ Carry-Select Based Redundant Binary to Two's Complement Converter. *IEEE Transactions on Circuits and Systems – I.* 2008*;* 55(1): 336 – 346.
[8] Sahoo SK, Gupta A, Asati AR, Shekhar C. A Novel Redundant Binary Number to Natural Binary Number Converter. *Journal of Signal Processing Systems.* 2010; 59(3): 297-307.
[9] Chakraborty MS. Reverse Conversion Schemes for Signed-digit Numbers Systems: A Survey. *Journal of Institute of Engineers of India, Series B.* 2016.
[10] Chkraborty MS, Ghosh T, Mondal AC. Notes on A Novel Conversion Scheme from a Redundant Binary Number to Twos'-Complement Binary Number for Parallel Architecturea proposed by Choo et. al. *International Journal of Computer Applications*. 2014; 103(5): 5-7.
[11] Lv X, Xiao M. An Efficient Reverse Converter for the New High Dynamic Range 5-Moduli Set. *Indonesian Journal of Electrical Engineering.* 2013; 11(11): 6577-6583.
[12] Blair GM. The Equivalence of Two's-Complement Addition and the Conversion of Redundant Binary to Two's-Complement Numbers. *IEEE Transactions on Circuit and Systems I: Fundamental Theory and Applications.*1998*;* 45(6): 669-671.
[13] Rulling W. A Remark on Carry-Free Binary Multiplication. *IEEE Journal of Solid - State Circuits.* 2003; 38(1) 159-160.
[14] Stouraitis T, Chen C. Fast Digit-Parallel Conversion of Signed-Digit into Conventional Representations. *Electronics Letters*. 1991; 27(11): 964-965.
[15] Srikanthan T, Lam SK, Suman M. Area–Time Efficient Sign Detection Technique for Binary Signed–Digit Number System. *IEEE Transactions on Computers.* 2004; 53(1): 69-72.
[16] Chakraborty MS, Sao SK. Comments on Area-Time Efficient Sign Detection Technique for Binary Signed-Digit Number System proposed by Srikanthan, Lam and Suman. *International Journal of Computer Applications*. 2014; 88(15): 38-40.