

Residue Arithmetic Circuits Based on the Signed-Digit Multiple-Valued Arithmetic Circuits

Shugang Wei Kensuke Shimizu
Department of Computer Science, Gunma University
Tenjin-cho, 1-5-1, Kiryu 376, Japan
wei@ja4.cs.gunma-u.ac.jp

Abstract

Multiple-valued residue arithmetic circuits using integers 4^p and $4^p \pm 1$ as moduli of residue number system(RNS) are presented. Conventional residue arithmetic circuits have been designed using binary number arithmetic system, but the carry propagation arises which limits the speed of arithmetic operations in residue modules. In this paper, a radix-4 signed-digit(SD) number system is introduced, and the compact SD adder based on the multiple-valued current-mode circuits is applied for the implementation of high-speed and compact residue arithmetic circuits. The modulo m addition, $m = 4^p$ or $m = 4^p \pm 1$, can be performed by an SD adder or an end-around-carry SD adder with the multiple-valued circuits and the addition time is independent of the word length of operands. Modulo m multiplier can be compactly constructed using a binary modulo m SD adder tree based on the multiple-valued addition circuits, and the modulo m multiplication can be performed in a time proportional to $\log_2 p$.

1. Introduction

The residue number system(RNS) has the well-known property that the i th residue digit of sum, difference, and product is exclusively dependent on the i th digits of the operands[1]. This property determines that truly parallel operations can be performed on all residue digits. Various methods of applications of RNS in digital signal processing have been proposed[2].

Integers 2^r and $2^r \pm 1$ are commonly used as moduli for RNS, because the additions modulo 2^r or $2^r \pm 1$ can be implemented by r -bit binary adders[1, 3]. Some modulo $2^r \pm 1$ multipliers have been proposed[4, 5]. However, since these modulo $2^r \pm 1$ adders and multipliers are designed based on the ordinary binary arithmetic system, the carry propagation arises during additions and limits the speed of arithmetic operations in residue modules.

It is known that carry propagation is limited to one position during additions of signed-digit (SD) numbers[6]. A number of high-speed arithmetic circuits based on SD number systems have been presented [7, 8, 9]. For

residue arithmetic, radix-5 SD multiple-valued arithmetic circuits have been proposed based on the concept of the pseudo-primitive root[10]. We have also presented a modulo $2^p - 1$ arithmetic algorithm using radix-2 SD number representation[11]. In this paper, a radix-4 SD number representation is introduced to the residue number system, so that high-speed and compact residue arithmetic circuits can be implemented using multiple-valued current-mode circuits and a radix-4 SD number is easily converted to binary one.

First we give the definition and some properties on redundant modular representation for RNS, by which the efficient modular arithmetic algorithm with SD numbers can be constructed [12]. Then by using a set of integers 4^p and $4^p \pm 1$ as moduli m and applying the radix-4 SD number representation to the RNS, the modulo m addition may be implemented simply using a p -digit SD adder or an end-around-carry SD adder. The SD adder can be compactly implemented using the multiple-valued current-mode circuits. Thus the complexity of the modulo m adder is the same as that of a p -digit SD adder, and the modulo m addition time is independent of the word length of the operands. A modulo m multiplier can be implemented by a binary modulo m SD adder tree, whose layout for VLSI may be simpler than that of SD multiplier because all of modulo m adders in the multiplier have the same word length structure. In this case, the time required for the modulo m multiplication is proportional to $\log_2 p$.

2. Residue Number System Using Redundant Modular Representation

In order to simplify the residue arithmetic with the radix-4 SD number representation, we use a set of positive integers as the moduli, represented in the following forms: 4^p , $4^p - 1$ and $4^p + 1$, where p is a positive integer. Different values of p can be selected by satisfying the condition that the moduli are relatively prime in pairs. At least, a set of integers with same p value, for example $\{4^8 - 1, 4^8, 4^8 + 1\}$, can be used as the moduli in an RNS.

Conventionally, a nonnegative integer A is represented by the n -tuple (A_1, A_2, \dots, A_n) in an RNS with moduli

m_1, m_2, \dots, m_n , where A_i is in the range $[0, m_i)$ by

$$A_i = A \bmod m_i = |A|_{m_i}, \quad (i = 1, 2, \dots, n). \quad (1)$$

In this paper, to replace the binary number by the radix-4 SD number for high-speed residue arithmetic, we define the redundant modular representation by expanding the value range in each residue digit as follows.

Definition 1: Let X be an integer and m be a positive integer. Then $x = \langle X \rangle_m$ is defined as an integer in the range $[-m, m]$ as follows:

$$x = \langle X \rangle_m = \text{sign}(X) \times |abs(X)|_m, \quad (2)$$

or,

$$x = \langle X \rangle_m = \text{sign}(X) \times |abs(X)|_m - \text{sign}(X) \times m, \quad (3)$$

where

$$\text{sign}(X) = \begin{cases} -1 & X < 0 \\ 1 & X \geq 0 \end{cases}$$

and

$$abs(X) = \begin{cases} -X & X < 0 \\ X & X \geq 0 \end{cases}.$$

□

For example, $\langle 17 \rangle_7 = 3$ by Eq.(2) or -4 by Eq.(3). Obviously, for a given integer A satisfying $|abs(A)|_m = 0$, there are three possible values, that is, $-m$, 0 and m . Thus, the addition, subtraction and multiplication of n -tuples $A = (A_1, A_2, \dots, A_n)$ and $B = (B_1, B_2, \dots, B_n)$ in an RNS can be represented as follows:

$$A \pm B = (\langle A_1 \pm B_1 \rangle_{m_1}, \dots, \langle A_n \pm B_n \rangle_{m_n}), \quad (4)$$

$$A \times B = (\langle A_1 \times B_1 \rangle_{m_1}, \dots, \langle A_n \times B_n \rangle_{m_n}). \quad (5)$$

A modulus m can be represented by $m = m(p, h) = 4^p + h$, where $h \in \{-1, 0, 1\}$. Since $\langle 4^p \rangle_{m(p, h)} = -h$, we have the following property which is very useful for our modulo $m(p, h)$ arithmetic.

Property 1: Let k be a positive integer. Then

$$\langle 2^k \rangle_{2^p+h} = \begin{cases} 4^k & k < p \\ (-h)^{(k \text{ div } p)} \times 4^{|k|_p} & k \geq p \end{cases}, \quad (6)$$

where $h \in \{-1, 0, 1\}$ and $(k \text{ div } p)$ is the integer part of the division result.

□

For example, $\langle 4^{10} \rangle_{m(3,1)} = (-1)^{(10 \text{ div } 3)} \times 4^{|10|_3} = -4$.

Obviously, the following properties exist in the redundant modular representation.

Property 2: Let a and b be integers. Then

$$(a) \quad abs(\langle a \rangle_m) \leq m,$$

$$(b) \quad \langle a + b \rangle_m \equiv \langle \langle a \rangle_m + \langle b \rangle_m \rangle_m,$$

$$(c) \quad \langle a \times b \rangle_m \equiv \langle \langle a \rangle_m \times \langle b \rangle_m \rangle_m,$$

and

$$(d) \quad \langle -a \rangle_m \equiv -\langle a \rangle_m,$$

where \equiv indicates a binary congruent relation with modulo m .

□

3. Residue Arithmetic Based on the Radix-4 Signed-Digit Number Representation

Integer x is represented by the q -digit radix-4 SD number representation as follows:

$$\begin{aligned} x &= x_{q-1}4^{q-1} + x_{q-2}4^{q-2} + \dots + x_0, \\ x_i &\in \{-3, -2, -1, 0, 1, 2, 3\} \quad (i = 0, 1, \dots, q-1), \end{aligned} \quad (7)$$

which can be rewritten as $x = (x_{q-1}, x_{q-2}, \dots, x_0)_{SD}$. Obviously,

$$\begin{aligned} -x &= -(x_{q-1}, x_{q-2}, \dots, x_0)_{SD} \\ &= (-x_{q-1}, -x_{q-2}, \dots, -x_0)_{SD}. \end{aligned}$$

The SD number representation has redundancy; for example, 13 may be represented by $(0, 3, 1)_{SD}$, $(1, -1, 1)_{SD}$ or $(1, 0, -3)_{SD}$ for $q = 3$.

With the properties mentioned above, modulo $m(p, h)$ SD addition using the radix-4 SD number representation can be performed by the following algorithm.

[Algorithm 1 (Calculation of $\langle a + b \rangle_{m(p, h)}$)] Let a and b be two integers in the p -digit radix-4 SD number representation, and c_i, z_i and s_i be the intermediate carry, intermediate sum and sum at the i th SD digit ($i = 0, 1, \dots, p-1$), respectively. For each digit, the following three steps are performed.

1)

$$w_i = a_i + b_i,$$

where w_i is a linear sum of a_i and b_i and $-6 \leq w_i \leq 6$.

2)

$$z_i = \begin{cases} w_i + 4 & \text{if } w_i \leq -2 \\ w_i & \text{if } -1 \leq w_i \leq 1 \\ w_i - 4 & \text{if } w_i \geq 2 \end{cases}$$

and

$$c_i = \begin{cases} -1 & \text{if } w_i \leq -2 \\ 0 & \text{if } -1 \leq w_i \leq 1 \\ 1 & \text{if } w_i \geq 2 \end{cases},$$

where $z_i \in \{-2, -1, 0, 1, 2\}$ and $c_i \in \{-1, 0, 1\}$.

3) $s_i = z_i + c_{i-1}$ for $i \neq 0$ and $s_0 = z_0 + (-h) \times c_{p-1}$. Thus,

$$\langle a + b \rangle_{m(p, h)} = s = s_{p-1}4^{p-1} + s_{p-2}4^{p-2} + \dots + s_0.$$

□

The above algorithm is the same as the SD addition method by Kawahito et al.[8] except the step 3) for $i = 0$. It is always true that $4c_i + z_i = a_i + b_i$ and $s_i \in \{-3, -2, -1, 0, 1, 2, 3\}$. Thus the carry propagation is limited to one digit. The modulo $m(p, h)$ addition algorithm can be implemented efficiently using an SD adder for $h = 0$ or an end-around SD adder for $h \in \{-1, 1\}$.

$\begin{array}{r} i : 4 \quad 3 \quad 2 \quad 1 \quad 0 \\ \hline a : 1 \quad 0 \quad -2 \quad 3 \quad 1 \\ b : 2 \quad 1 \quad -1 \quad 0 \quad 3 \\ \hline 1) \quad w : 3 \quad 1 \quad -3 \quad 3 \quad 4 \\ \hline z : \quad -1 \quad 1 \quad 1 \quad -1 \quad 0 \\ c : 1 \quad 0 \quad -1 \quad 1 \quad 1 \quad \boxed{1} \\ \hline 3) \quad \underbrace{\hspace{10em}} \\ s : \quad -1 \quad 0 \quad 2 \quad 0 \quad 1 \\ \text{(a) } m = 1023 \end{array}$	$\begin{array}{r} i : 4 \quad 3 \quad 2 \quad 1 \quad 0 \\ \hline a : 1 \quad 0 \quad -2 \quad 3 \quad 1 \\ b : 2 \quad 1 \quad -1 \quad 0 \quad 3 \\ \hline 1) \quad w : 3 \quad 1 \quad -3 \quad 3 \quad 4 \\ \hline z : \quad -1 \quad 1 \quad 1 \quad -1 \quad 0 \\ c : 1 \quad 0 \quad -1 \quad 1 \quad 1 \quad \boxed{-1} \\ \hline 3) \quad \underbrace{\hspace{10em}} \\ s : \quad -1 \quad 0 \quad 2 \quad 0 \quad -1 \\ \text{(b) } m = 1025 \end{array}$
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure 1: Example of modulo m addition by algorithm 1

The calculation of $\langle a - b \rangle_{m(p,h)}$ can be realized by replacing b with $-b$ in the above algorithm.

Example 1 : Let $p = 5$, $a = (1, 0, -2, 3, 1)_{SD}$ and $b = (2, 1, -1, 0, 3)_{SD}$, so that $m(5, -1) = 1023$, $m(5, 0) = 1024$ and $m(5, 1) = 1025$, $a = 237$ and $b = 563$. Fig.1 illustrates the calculations of $\langle a + b \rangle_{m(p,h)}$ using the above algorithm. The results are $\langle 237 + 563 \rangle_{1023} = -479$, $\langle 237 + 563 \rangle_{1024} = -480$ and $\langle 237 + 563 \rangle_{1025} = -481$. \square

To calculate $\langle a \times b \rangle_{m(p,h)}$, where a and b are integers in the p -digit radix-4 SD number representation, $a \times b$ can be extended as follows:

$$\begin{aligned} a \times b &= (a_{p-1}4^{p-1} + a_{p-2}4^{p-2} + \dots + a_0) \\ &\quad \times (b_{p-1}4^{p-1} + b_{p-2}4^{p-2} + \dots + b_0) \\ &= \sum_{i=0}^{p-1} b_i 4^i \times (a_{p-1}4^{p-1} + a_{p-2}4^{p-2} \\ &\quad + \dots + a_0). \end{aligned}$$

Thus,

$$\begin{aligned} \langle a \times b \rangle_{m(p,h)} &= \left\langle \sum_{i=0}^{p-1} \langle b_i 4^i \times (a_{p-1}4^{p-1} + a_{p-2}4^{p-2} \right. \\ &\quad \left. + \dots + a_0) \rangle_{m(p,h)} \right\rangle_{m(p,h)} \\ &= \left\langle \sum_{i=0}^{p-1} pp_i \right\rangle_{m(p,h)}, \end{aligned}$$

where pp_i denotes as a partial product.

We express b_i with a sum of three items:

$$b_i = b'_i + b''_i + 4b'''_i, \quad (8)$$

where $b'_i, b''_i, b'''_i \in \{-1, 0, 1\}$ are determined as radix-2 signed digits by Table 1,

Table 1 Decomposition for radix-4 signed digit b_i

$abs(b_i)$	b'_i	b''_i	b'''_i
0	0	0	0
1	b_i	0	0
2	$b_i/2$	$b_i/2$	0
3	$-sign(b_i)$	0	$sign(b_i)$

Replacing b_i by b'_i, b''_i, b'''_i and using Property 1 shown in Eq.(6), we have

$$\begin{aligned} pp_i &= \langle b_i 4^i (a_{p-1}4^{p-1} + a_{p-2}4^{p-2} + \dots + a_0) \rangle_{m(p,h)} \\ &= \langle \langle b'_i 4^i (a_{p-1}4^{p-1} + a_{p-2}4^{p-2} + \dots + a_0) \rangle_{m(p,h)} \\ &\quad + \langle b''_i 4^i (a_{p-1}4^{p-1} + a_{p-2}4^{p-2} + \dots + a_0) \rangle_{m(p,h)} \\ &\quad + \langle b'''_i 4^{i+1} (a_{p-1}4^{p-1} + a_{p-2}4^{p-2} \\ &\quad + \dots + a_0 4) \rangle_{m(p,h)} \rangle_{m(p,h)} \\ &= \langle b'_i (a_{p-i-1}4^{p-1} + a_{p-i-2}4^{p-2} + \dots + a_0 4^i \\ &\quad - h(a_{p-1}4^{i-1} + \dots + a_{p-i+1}4 + a_{p-i})) \\ &\quad + b''_i (a_{p-i-1}4^{p-1} + a_{p-i-2}4^{p-2} + \dots + a_0 4^i \\ &\quad - h(a_{p-1}4^{i-1} + \dots + a_{p-i+1}4 + a_{p-i})) \\ &\quad + b'''_i (a_{p-i-2}4^{p-1} + a_{p-i-3}4^{p-2} + \dots + a_0 4^{i+1} \\ &\quad - h(a_{p-1}4^i + \dots + a_{p-i}4 + a_{p-i-1})) \rangle_{m(p,h)}. \end{aligned} \quad (9)$$

In the above equation, the sum of three radix-4 signed digits at each position has the value range $[-6, 6]$, because one of the three SD numbers must be zero. Therefore, $\langle a \times b \rangle_{m(p,h)}$ is represented by a modulo $m(p, h)$ sum of p partial products, and can be implemented by the following three steps.

[Algorithm 2 (Calculation of $\langle a \times b \rangle_{m(p,h)}$)] Let a and b be two p -digit radix-4 SD numbers.

1) Determine b'_i, b''_i, b'''_i from b_i using Table.1 for $i = 0, 1, \dots, p-1$.

2) Calculate partial products, pp_i ($i = 0, 1, \dots, p-1$), as follows: let

$$\begin{aligned} pp'_i &= b'_i (a_{p-i-1}, a_{p-i-2}, \dots, a_0, -ha_{p-1}, \\ &\quad \dots, -ha_{p-i+1}, -ha_{p-i}), \\ pp''_i &= b''_i (a_{p-i-1}, a_{p-i-2}, \dots, a_0, -ha_{p-1}, \\ &\quad \dots, -ha_{p-i+1}, -ha_{p-i}) \\ pp'''_i &= b'''_i (a_{p-i-2}, \dots, a_0, -ha_{p-1}, \\ &\quad \dots, -ha_{p-i+1}, -ha_{p-i}, -ha_{p-i-1}), \end{aligned}$$

then

$$pp_i = \langle pp'_i + pp''_i + pp'''_i \rangle_{m(p,h)}, \quad (10)$$

where subscripts are in the range $[0, p-1]$.

3) Calculate the modulo $m(p, h)$ sum of these partial products by performing the steps in Algorithm 1 repeatedly.

$$\langle a \times b \rangle_{m(p,h)} = \langle pp_0 + pp_1 + \dots + pp_{p-1} \rangle_{m(p,h)}$$

BASIC CIRCUIT	Current Source	Current Mirrors		Threshold Detector	Bidirectional Current Input
		N-ch.Type	P-ch.Type		
Symbol					
Function	$Y=0$ if $X="1"$; $Y=m$ if $X="0"$	$Y_i = -a_i X$ for $i=1..n$ a_i :Scale Factor		$Y=0$ if $X \leq T$; $Y=m$ if $X > T$.	$X^+=X, X^-=0$ if $X \geq 0$ $X^-=0, X^+=X$ if $X < 0$.

Figure 2: Symbols and functions of the current-mode circuits

Example 2 : Let $p = 4$, $a = (-3, 0, 2, -1)_{SD}$ and $b = (2, -3, 1, -1)_{SD}$. That is $m(4, 1) = 257$, $a = -185$ and $b = 83$. By Algorithm 2, 1) $b'_0 = -1, b''_0 = b'''_0 = 0$ for $b_0 = -1$; $b'_1 = 1, b''_1 = b'''_1 = 0$ for $b_1 = 1$; $b'_2 = 1, b''_2 = 0, b'''_2 = -1$ for $b_2 = -3$; $b'_3 = 1, b''_3 = 1, b'''_3 = 0$ for $b_3 = 2$. Then, 2) the partial products are calculated as follows:

$$\begin{aligned}
pp_0 &= \langle (-1)(-3, 0, 2, -1)_{SD} + 0(-3, 0, 2, -1) \\
&\quad + 0(0, 2, -1, -1)(-3) \rangle_{257} \\
&= (3, 0, -2, 1)_{SD} \\
pp_1 &= \langle 1(0, 2, -1, (-1)(-3))_{SD} \\
&\quad + 0(0, 2, -1, (-1)(-3)) \\
&\quad + 0(2, -1, (-1)(-3), (-1)(0)) \rangle_{257} \\
&= (0, 2, -1, 3)_{SD} \\
pp_2 &= \langle 1(2, -1, (-1)(-3), (-1)(0))_{SD} \\
&\quad + 0(2, -1, (-1)(-3), (-1)(0)) \\
&\quad + (-1)(-1, -1(-3), -1(0), (-1)2) \rangle_{257} \\
&= (-2, 1, 0, -3)_{SD} \\
pp_3 &= \langle 1(-1, (-1)(-3), (-1)0, (-1)2)_{SD} \\
&\quad + 1(-1, (-1)(-3), (-1)0, (-1)2)_{SD} \\
&\quad + 0((-1)(-3), (-1)(0), (-1)2, (-1)(-1)) \rangle_{257} \\
&= (3, 2, -1, 1)_{SD};
\end{aligned}$$

and 3) the modulo $m(p, h)$ sum of the partial products is calculated by applying algorithm 1 repeatedly.

$$\begin{aligned}
\langle a \times b \rangle_{257} &= \langle \langle pp_0 + pp_1 \rangle_{257} + \langle pp_2 + pp_3 \rangle_{257} \\
&= \langle (0, -3, 2, -1)_{SD} + (2, -1, -2, 2)_{SD} \rangle_{257} \\
&= (-3, 0, 0, 0)_{SD}.
\end{aligned}$$

The result is $\langle (-185) \times 83 \rangle_{257} = -192$. \square

4. Residue Arithmetic Circuits based on the Current-mode Circuits

The bidirectional current-mode MOS circuits are very suitable for the radix-4 SD number system, so that high-speed and compact SD adder and multiplier have been

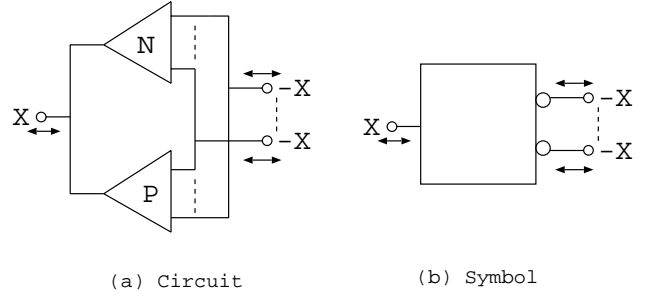


Figure 3: Sign inverter

presented based on the key idea that the addition can be easily implemented by wiring some signal lines[13, 8]. In this paper, we suppose that both input and output of the residue arithmetic circuits are expressed by current signals. In the case of voltage-mode input and output, the conversion circuits between current-mode and voltage-mode signals are required.

4.1 Basic current-mode circuits and modulo m SD adder

Fig.2 illustrates the basic current-mode circuits whose detailed explanation is found in Ref.[13]. Based on these basic circuits, compact SD full adder(SDFA) implementation has been discussed[8]. Fig.3 shows the circuit of a sign inverter which inverts the polarity of the input current, and can have multiple fanouts. Using one sign inverter and p SDFAs, modulo m SD adder(MSDA), where $m = m(p, 1)$, can be constructed to implement Algorithm 1 as shown in Fig.4. In the cases of $m = m(p, -1)$ and $m = m(p, 0)$, the sign inverter and the end-around carry are not required, respectively.

4.2 Modulo m SD multiplier

To construct the modulo m SD multiplier implementing Algorithm 2, we first consider the circuit of partial

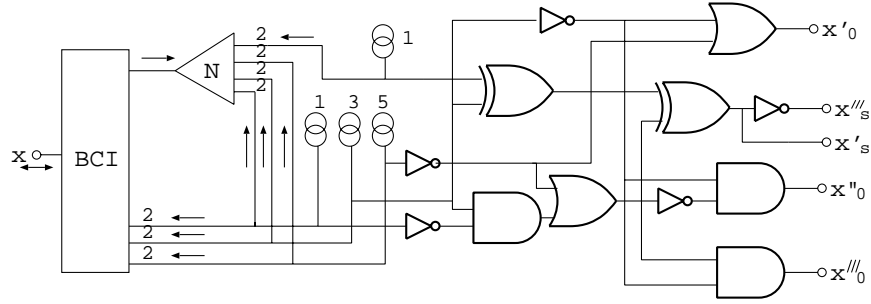


Figure 5: Decoder of a multiplier using I-V converter

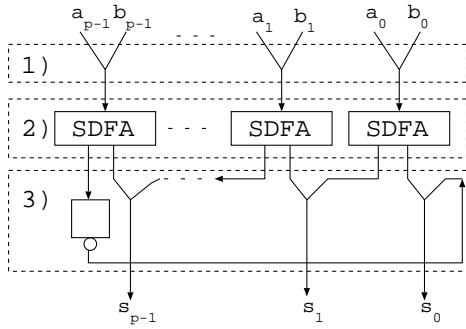


Figure 4: Block diagram of modulo m SD adder(MSDA) based on Algorithm 1

product generator. Suppose that multiplicand a and multiplier b are p -digit SD numbers expressed by the current-mode signals. We divide each digit b_i of b into three radix-2 signed-digits by Table.1. A radix-2 number g is encoded into two-bit binary code as shown in Table.2, which can be expressed by two-valued voltage signals.

Table.2 Binary representation of radix-2 signed digit

g	g_s (sign)	g_0 (magnitude)
-1	1	1
0	*	0
1	0	1

Thus a decoder circuit for a radix-4 signed-digit is implemented as shown in Fig.5 and the voltage-mode outputs of the decoder are used to switch pass MOS transistors for generating partial products. Fig.6 shows a multiply unit by a radix-2 signed digit(MU), which implements the following function:

$$Y = \begin{cases} X & (g_s = 0) \\ 0 & (g_0 = 0) \\ -X & (g_s = 1) \end{cases}. \quad (11)$$

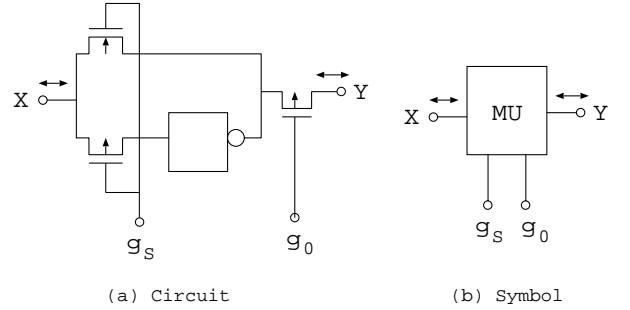


Figure 6: Multiply unit by radix-2 SD number(MU)

Thus, partial product generator(PPG) can be constructed using $3p$ MUs and one MSDA for $m = 4^p + 1$ as shown in Fig.7. For $m = 4^p - 1$ and $m = 4^p$, the inputs $-a_j$ ($j = p - i - 1, \dots, p - 1$) in Fig.8 are replaced by a_j and 0 respectively. That means, specially for $m = 4^p$, the corresponding circuits with zero-input are not required. The circuits shown in Fig.5 and Fig.7 implement steps 1) and 2) of Algorithm 2, respectively. Therefore, modulo m SD multiplier can be designed using a binary adder tree as shown in Fig.8. In the case of multiplier, the longest addition path is $\log_2 p + 1$. A compact design of the multiplier can be realized: for example, about 8000 transistors are required to construct the modulo $m(8, \pm 1)$ multiplier, in which the multiply time is estimated to be less than 33ns based on $2\mu\text{m}$ CMOS technology. For $m = 4^p$, moreover, half of SD-FAs can be saved.

For VLSI implementation of the modulo m multiplier, the chip layout methods proposed in Refs.[8] can be applied, because similar adder cells and binary adder tree may be constructed. Since the same MSDAs are used in the adder tree, the layout may be much simpler than that of an SD multiplier.

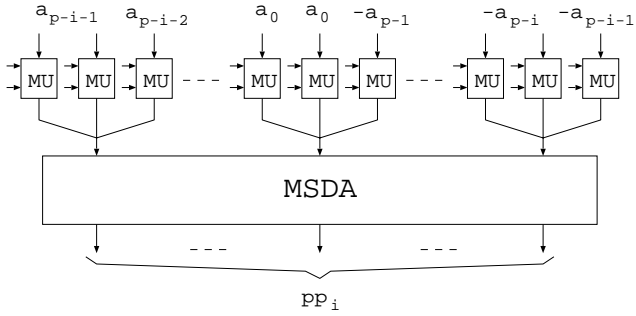


Figure 7: Partial product generator(PPG)

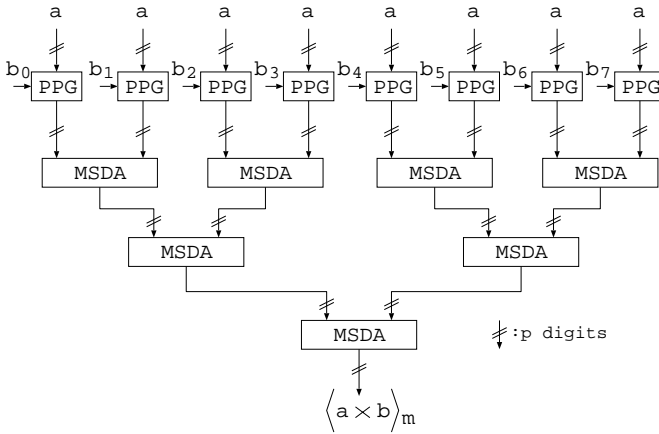


Figure 8: Block diagram of modulo m SD multiplier with a binary adder tree.

5. Conclusion

VLSI-oriented hardware algorithm for residue arithmetic using signed-digit number representation has been presented based on the multiple-valued current-mode circuits. Using integers $m \in \{4^p, 4^p \pm 1\}$ as moduli, the modulo m addition is implemented by an SD adder or an end-around-carry SD adder with the current-mode MOS SD full adders and the modulo m multiplier can be constructed with a binary SD adder tree. Thus the modulo m multiplication is performed in a time proportional to $\log_2 p$.

High-speed computations can be performed based on the assumption that input and output data of the residue arithmetic circuits are in the residue SD number form, because some computing system applications, such as digital filtering, require repeated calculations of sums of products before the final results are obtained. For integration with conventional binary systems, efficient circuits are required to convert into and out of the residue

SD systems. Our studies also focus on the application of the presented residue circuits to computation systems, such as digital signal processing and digital control systems.

References

- [1] N.S.Szabo and R.I.Tanaka, "Residue Arithmetic and Its Applications to Computer Technology", New York : McGraw-Hill, 1967.
- [2] M. A. Sonderstrand, W. K. Jendins, G. A. Junllien, and F. J. Taylor, "Residue Number System Arithmetic: Modern Applications in Digital Signal Processing," IEEE Press, New York, 1986.
- [3] D.P. Agrawal and T.R.N.Rao, "Modulo $(2^n + 1)$ arithmetic logic," IEE J. Electronic Circuits and Systems, Vol.2, pp. 186-188, Nov. 1978.
- [4] F.J.Taylor, "A VLSI residue arithmetic multiplier," IEEE Trans. Comput., Vol.C-31, pp.540-546, June 1982.
- [5] A.Hiasat, "New memoryless, mod $(2^n \pm 1)$ residue multiplier," Electron. Lett., Vol.28, No.3, pp.314-315, Jan. 1992.
- [6] A.Avizienis, "Signed-digit number representations for fast parallel arithmetic," IRE Trans. Elect. Comput., EC-10, pp.389-400, Sept. 1961.
- [7] N.Takagi, H.Yasuura and S.Yajima, "High-Speed VLSI multiplication algorithm with a redundant binary addition tree," IEEE Trans. Comput., Vol. C-34, pp.789-796, Sept. 1985.
- [8] S.Kawahito, M.Kameyama, T.Higuchi and H.Yamada, "A 32×32 -bit multiplier using multiple-valued MOS current circuits," IEEE J. Solid-State Circuits, SC-23, No.1, pp.124-132, Feb. 1988.
- [9] S. Wei, M. Kameyama and T. Higuchi, "Performance Evaluation of a Multiple-Valued RSA Encryption VLSI," Trans. IEICE, Vol. J73-D-I, No. 5, pp.484-491, May 1990.
- [10] M.Kameyama, T. Sekibe and T.Higuchi, "Highly-parallel residue arithmetic chip based on multiple-valued bidirectional current-mode circuits," IEEE J. Solid-State Circuits, SC-24, No.5, pp.1404-1411, May 1989.
- [11] S.Wei and K.Shimizu, "Modulo $2^p - 1$ arithmetic hardware algorithm using signed-digit number representation," Trans. IEICE, Vol.E79-D, No.3, pp.242-246, March 1996.
- [12] M.Kameyama, S.Wei and T.Higuchi, "Design of an RSA encryption processor based on the signed-digit multiple-valued arithmetic circuits," Trans. IEICE, Vol. J71-D, No.12, pp. 2659-2668, Dec.1988.
- [13] S. Kawahito, M. Kameyama and T. Higuchi, "VLSI-Oriented Bi-Directional Current-Mode Arithmetic Circuits Based on the Radix-4 Signed-Digit Number System", Proc. 1986 Int. Symp. Multiple-Valued Logic, pp.70-77, May 1986.