International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:2, No:3, 2008

Improved Modulo 2^n +1 Adder Design

Somayeh Timarchi, Keivan Navi

***Abstract*—**Efficient modulo 2^n+1 adders are important for several applications including residue number system, digital signal processors and cryptography algorithms. In this paper we present a novel modulo 2^n+1 addition algorithm for a recently represented number system. The proposed approach is introduced for the reduction of the power dissipated. In a conventional modulo 2^n+1 adder, all operands have (n+1)-bit length. To avoid using (n+1)-bit circuits, the diminished-1 and carry save diminished-1 number systems can be effectively used in applications. In the paper, we also derive two new architectures for designing modulo 2^n+1 adder, based on n-bit ripple-carry adder. The first architecture is a faster design whereas the second one uses less hardware. In the proposed method, the special treatment required for zero operands in Diminished-1 number system is removed. In the fastest modulo 2^n+1 adders in normal binary system, there are 3-operand adders. This problem is also resolved in this paper. The proposed architectures are compared with some efficient adders based on ripple-carry adder and high-speed adder. It is shown that the hardware overhead and power consumption will be reduced. As well as power reduction, in some cases, power-delay product will be also reduced.

***Keywords*—**Modulo 2^n+1 arithmetic, residue number system, low power, ripple-carry adders.

I. INTRODUCTION

THE modular characteristic of the Residue Number System (RNS) offers the potential for high-speed and parallel arithmetic. In RNS logic, each operand is represented by its residues with respect to a set of numbers comprising the base. Addition, subtraction and multiplication are performed in parallel on the residues in distinct design units (often called channels) avoiding carry propagation among residues [1], [2]. So, arithmetic operations, e.g. addition, subtraction and multiplication can be carried out more efficiently in RNS than in conventional two's complement systems. That makes RNS a good candidate for implementing a lot of application fields [2]. Typical applications of the RNS can be found in Digital Signal Processing (DSP) for filtering, convolutions, correlations, FFT computation [3], [4], fault-tolerant computer systems [5], communication [6], cryptography [7], [8].

The choice of moduli set is very important and necessary for nearly equal delay of the channels. Special moduli sets have been used extensively to reduce the hardware complexity in the implementation of converters and arithmetic operations [9]-[13]. Among which the triple moduli set $\{2^n-1, 2^n, 2^n+1\}$ has some benefits [14]. Because of operand lengths of these moduli, the operation delay of this system is determined by the modulo 2^n+1 channel. The latter means that, if we cut down the time required for modulo 2^n+1 addition, we also cut down the RNS addition time [15].

In order to speed up the modulo 2^n+1 arithmetic operations the diminished-1 representation of binary numbers has been introduced in [16]. In the Diminished-1 number system, each number X is represented by $X^*=X$-1, while zero is handled separately. In this system, efficient adders have been reported in [17], [18]. But in these circuits, it is necessary to use special treatment for zero operands. To overcome mentioned problem, a number representation so-called "Carry Save Diminished-1" has been proposed in [19] and [20]. In this paper, an addition algorithm in the carry save diminished-1 system is proposed. In the proposed addition algorithm, the special treatment for zero operands is not required.

Modulo 2^n+1 adders can also be designed as a special case of general modulo m adders. The most efficient circuits for generalized modulo adders have been reported in [15], [21], and [22]. It has been shown in [15] that its proposed architecture is the most efficient adder in comparison to [21] and [22] structures. However, the problem of [15] is the existence of a 3-operand adder which is eliminated in our method.

In the following, it is shown that the novel architecture removes some significant problems of previous structures and reduces both area and power dissipation. In the paper, we derive new methodology for modulo 2^n+1 adder that leads to a ripple-carry adder architecture. Although ripple-carry adder has more delay than carry-accelerate adder, it is useful for low power and low area applications. Using implementation in a CMOS technology, we show that the proposed ripple-carry design methodology leads to considerably less area and power consumption than those reported in the related papers and in some cases, power-delay product is also reduced.

The rest of the paper is organized as follows. The conventional methods for modulo 2^n+1 adder including general modulo adders, diminished-1 and carry save diminished-1 modulo adders implemented by ripple-carry and parallel-prefix addition are reviewed in the next section. In section 3, the modulo 2^n+1 addition algorithm is proposed and implemented using ripple-carry adder. In section 4 the resulted structure is compared to the most efficient conventional adders. Conclusions are given in the last section.

S. Timarchi is with the Faculty of Electrical & Computer Engineering, Shahid Beheshti University, Tehran, Iran (e-mail: s_timarchir@sbu.ac.ir).
K. Navi, is with the Faculty of Electrical & Computer Engineering, Shahid Beheshti University, Tehran, Iran (corresponding author, phone: +982129902282; fax: +982122431804; e-mail: navi@sbu.ac.ir).

II. Review on Modulo 2^N+1 Adders

Addition delay of moduli set of the form $\{2^n-1, 2^n, 2^n+1\}$ is determined by the 2^n+1 channel; because this module has (n+1)-bit wide operands and the two first moduli have n-bit wide operands. Therefore, the design of efficient modulo 2^n+1 adders is very important.

Modulo 2^n+1 adders can be designed as a special case of general modulo m adders. The most efficient circuits are reported in [21], [22]. To remove the problem of (n+1)-bit wide circuits for the modulo 2^n+1 channel, the diminished-1 and carry save diminished-1 number systems [16] have been proposed.

A. Minimum Hardware Method

Consider a modulus m satisfying $2^{h-1} < m \le 2^h$. All binary residues in this modulus are h-bit unsigned numbers in the range [0, m).

One way of implementing a residue adder for modulo m is a structure composed of one h-bit adder. This adder adds two h-bit numbers, X and Y in the first step. In the second step, the result is added with the two's complement of modulo m. The final result is selected between the two outputs according to the two output carries. Since the output depends on the input, a combinational loop is created that may lead to an unwanted race condition. Two solutions for this problem are as bellow [18]:

a) In some cases, an additional logical operation on the feedback carry can eliminate the race condition.

b) The addition is done in two cycles. The output carry of the first cycle is added in the second cycle.

Other solutions for mentioned problem are introduced in the following sections.

B. Series Method

One solution for eliminating race condition is to apply two adders to compute the results of "$X+Y$" and "$X+Y-m$" in series. The correct sum is selected afterwards according to the output carries.

Modulo 2^n+1 adder in series method is shown in Fig. 1. The delay of modulo 2^n+1 structure is equal to the delay of two (n+1)-bit adders as well as the delay of one (n+1)-bit 2×1 multiplexer.

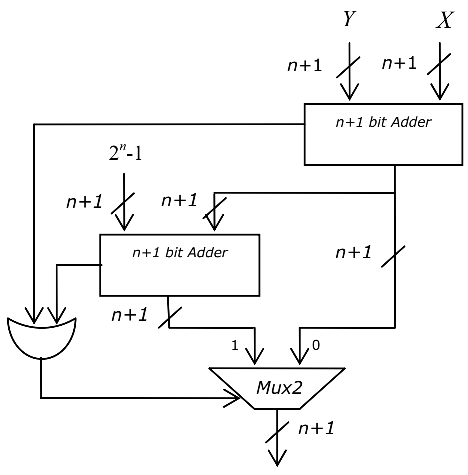

Fig. 1 Modulo 2^n+1 adder with series method

C. Parallel Method

For increasing the speed of Fig. 1 structure, series adders in this figure can be operated in parallel [23], [24]. One adder adds two h-bit residues, X and Y to form their sum $S_1 + 2^h C_{out1}$. Another one is a 3-operand adder that computes "$X+Y-m$". Note that if m=2^n+1, we have h=n+1. It has been reported in [25] that if either Cout1 or Cout2 of this addition is '1' then the output is $X+Y-m$ instead of $X+Y$. However, in the following we illustrate that only if the carry of "$X+Y-m$" is '1', it is sufficient to select it as the final output. The related structure is illustrated in Fig. 2.

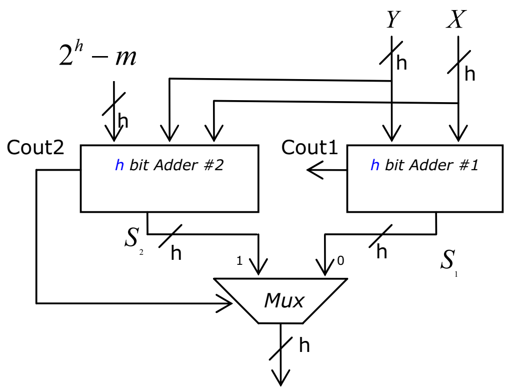

Fig. 2 Modulo m adder with parallel method

Let us assume $m \in (2^{h-1}, 2^h]$ and $X, Y \in [0, m)$ then,

$$X + Y + 2^h - m \le 2^h + m - 2 \Rightarrow X + Y + 2^h - m \le 2^{h+1} - 2$$

Since $S_2 + 2^h C_{out2} = X + Y + 2^h - m$ as shown in Fig. 2, $S_2 + 2^h C_{out2} \le 2^{h+1} - 2$. Therefore $C_{out2} \in \{0,1\}$, i.e., the 3-operand adder in Fig. 2 has one output carry. Besides, $2^h - m \in [0, 2^{h-1})$ then

$$(X+Y)<(X+Y+2^h-m)\Rightarrow(S_1+2^hC_{out1})<(S_2+2^hC_{out2})$$

It is obvious from above non-equality that if C_{out1} = 1 then C_{out2} = 1. It means that C_{out1} + C_{out2} = C_{out2}. So, we cay say that if C_{out2} = 1 then the output is $S = X+Y -m$ and if C_{out2} = 0 then the correct output is $X+Y$. One way of implementing the 3-operand adder is Carry Save Adder (CSA) that is shown in Fig. 3.

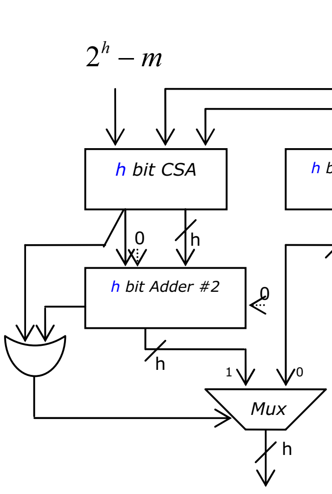

Fig. 3 Parallel modulo m adder with CSA

■

For modulo 2^n+1, this design is shown in Fig. 4. The critical path is visualized by an arrow in Fig. 4. Therefore, the delay is reduced to the delay of one CSA as well as one n-bit adder and multiplexer.

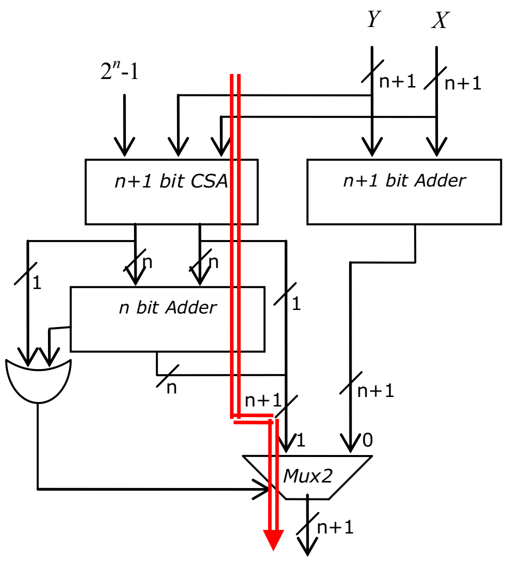

Fig. 4 Modulo 2^n+1 adder using CSA

D. Incrementer Method

In this method, one adder followed by an incrementer is used. One implementation of the design has been proposed in [15]. This design is based on theorem 1 of [15].

Theorem 1 of [15]: if X and Y are (n+1)-bit wide numbers in the range [0,2^n+1) then,

$$|X+Y|_{2^n+1}=\begin{cases}\left|\,|X+Y+2^n-1|_{2^{n+1}}+2^n+1\right|_{2^{n+1}} & \text{if } X+Y+2^n-1<2^{n+1}\\ |X+Y+2^n-1|_{2^{n+1}} & \text{otherwise}\end{cases}\qquad(1)$$

Equation (1) reveals that a two-stage combinational circuit can be utilized for the modulus addition. The first stage computes an intermediate sum “M”. If the most significant bit of M is ‘0’, the term 2^n+1 is added to the n least significant bits of M in the second stage. For computing M, a CSA which calculates a carry vector C, and a sum vector S, followed by an (n+1)-bit parallel adder, which calculates $2\times C+S$, can be utilized. Since 2^n-1 in its (n+1)-bit binary representation has a ‘0’ at its leftmost position and ‘1’s at all other positions, the CSA is composed of one half-adder (HA) at its leftmost position and n semi half-adders (HA*) in the other ones. An HA* is a full-adder which one of its inputs is ‘1’ [15]. This structure is shown in Fig. 5.

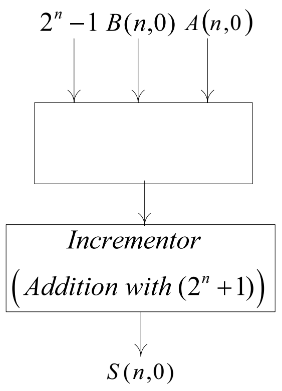

Fig. 5 The incrementer method for modulus addition

E. Diminished-1 Method

In the Diminished-1 number system, the number A is represented by $A'=A-1$ and the value zero is treated separately, i.e., it requires an additional zero indication bit [16]-[18].

S' = (S-1) = (X+Y-1) mod (2^n+1)
= [(X'+1) + (Y'+1) - 1)] mod (2^n+1)
= (X' + Y' + 1) mod (2^n+1)

The above equation can be depicted by (2).

$$(X'+Y'+1) \bmod (2^n+1) = \begin{cases} X'+Y'+1-(2^n+1) = (X'+Y') \bmod 2^n & \textit{if } X'+Y'+1 \geq 2^n \\ X'+Y'+1 & \textit{otherwise} \end{cases} \tag{2}$$

Algorithm 1 (Modulo 2^n +1 addition in diminished-1 number system): A number in Diminished-1 is represented by n+1 bits. The (n+1)th bit is used to indicate '0'. In [16], the modulo 2^n +1 addition algorithm has been presented for zero and non zero operands:

1) If the most significant bit of one addend is '1', inhibit the addition and the other addend is the output.

2) If the msb of both addends are '0', ignore the msb, add the n lsb's, complement the carry and add it to the n lsb's of the sum.

One structure of diminished-1 addition algorithm is depicted in Fig. 6. References [17], [18] and [26] have proposed implementations of diminished-1 method.

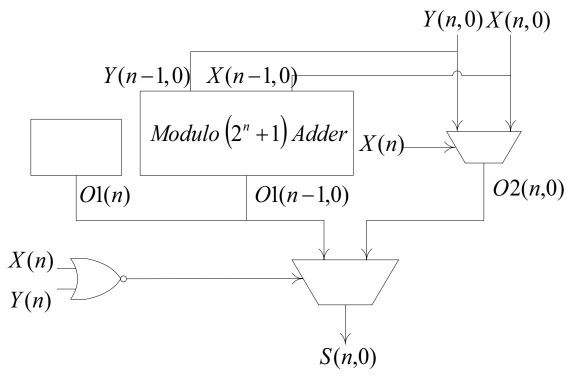

Fig. 6 The general design of Diminished-1 modulo adder

■

A. Carry Save Diminished-1 (CSD-1) Number System

Definition (Carry Save Diminished-1 modulo 2^n +1 encoding): The digit set [0,2^n] is composed of n positions with two bits in the least significant position and n-1 bits in the other ones.

We have proposed the encoding for modulo 2^n+1 in [19], [20]. Table I shows the representation of numbers in this module.

TABLE I
THE CSD-1 CODING FOR MODULO 2^N+1

Range	*Bit Representation*
[0 , 2^n]	$\overbrace{\bullet\cdots\bullet\bullet}^{n}$ $\bullet$

Real Zero Detection

A number X is represented as below:

$$X = x_{n-1}\ \ldots\ x_2\, x_1\, x'_0 \qquad\qquad x''_0 \tag{3}$$

where x'_0 and x''_0 are two bits in the first position of the binary positional number system. We called this system Carry Save Diminished-1 or abbreviatly "CSD-1". If $X \neq 0\ (X = 0)$ then $x'_0 = 1\ (x'_0 = 0)$. When we eliminate x'_0 from X representation in (3), the remaining bits are equal to diminished-1 representation of X.

The difference and benefit of the representation have been shown in [20]. As shown there, carry save diminished-1 has a unique circuit for zero and non zero operands. Also, carry save diminished-1 is extendable to any other modulo whereas diminished-1 is only defined for modulo 2^n +1 [20]. An efficient low-power CSD-1 parallel-prefix adder has been proposed in [19]. In this paper we try to improve the performance (power optimization) of modulo 2^n +1 arithmetic adders using carry save diminished-1 coding.

Carry save diminished-1 requires converters from/to the normal binary to/from diminished-1 representation. The converters are similar to diminished-1 converters. Let us assume the normal binary input A. The lsb a'_0 of CSD-1 representation is achieved by the logical XOR of all bits of A. The other bits are the binary representation of A-1.

CSD-1 multiplication can be done by small changes in normal multiplier structure. So, we use a basic modulo 2^n +1 multiplier [18] to show that CSD-1 number system is suitable for all RNS operations and intermediate results don't have to be translated into the new representation before using in another operation. Adder trees can be applied very easily to speed up carry-save addition of CSD-1 partial products in CSD-1 multiplication. Therefore, other arithmetic operations can handle the new representation. For example, a multiplication following an addition requires no translation step.

III. THE PROPOSED CARRY SAVE DIMINISHED-1 RIPPLE-CARRY ADDER

Algorithm 2 (Modulo 2^n+1 addition in the CSD-1 number system) Let X and Y are two carry save diminished-1 numbers and their addition is shown by 'S'. Then we have,

$$\begin{array}{rr} & x_{n-1}\ \ldots\ x_2\, x_1\, x'_0 \\ & x''_0 \\ + & y_{n-1}\ \ldots\ y_2\, y_1\, y'_0 \\ & y''_0 \\ \hline & s_{n-1}\ \ldots\ s_2\, s_1\, s'_0 \\ & s''_0 \end{array}$$

0

X

Two cases can occur depends on the value of X+Y value:

1 0

MUX

Case 1) If $X+Y<2^n+1$ then *S*=*X*+*Y*. One adder called "Madder#1" is used to calculate this value in CSD-1. In Madder#1, the addition range of four bits of the first digits (least significant bits: x'_0, x''_0, y'_0 and y''_0) is in the range [0, 4] which can be represented by two bits in the same position and one carry bit.

Case 2) If $X+Y\geq 2^n+1$ then *X*+*Y* should be reduced from 2^n+1. Another adder called "Madder#2" is used to calculate this value in CSD-1. As mentioned earlier, in CSD-1, each number is represented by *n* digits. Since *X*, $Y\in[0, 2^n]$, the second case leads to the following inequalities:

$$X+Y-1-2^n\leq 2^n+2^n-1-2^n<2^n$$

So,

$$X+Y-1-2^n=\left|X+Y-1-2^n\right|_{2^n}=\left|X+Y-1\right|_{2^n}$$

Therefore, it is sufficient that (*X*+*Y*) be decremented. However, this condition occurs when both numbers, *X* and *Y* are non zero. Considering the fact that the least significant position (composed of two bits) in each non zero number in carry save diminished-1 is 1 or 2, a decrementation can be done in the first digit without any carry propagation to the other positions.

■

Therefore, two special adder cells are required for the addition of the first positions in Madder#1 and Madder#2. In the other positions *i*, where 0 < *i* < *n*, the addition of x_i, y_i and c_{i-1} (the output carry of previous position) can be implemented by a conventional FA.

A. Addition Cell Design of First Digit of Madder#1 (MFA$_1$)

The MFA$_1$ cell adds the bits of first digits. The inputs are 4 bits (x'_0, x''_0, y'_0 and y''_0). The outputs of the MFA$_1$ are two sum bits $(s'_0)_{\text{MFA1}}$ and $(s''_0)_{\text{MFA1}}$ and one output carry $(c_0)_{\text{MFA1}}$. The addition of four bits is in the range [0,4]. Thus five cases can occur as shown in Table II.

TABLE II
MFA$_1$ Truth Table

$x'_0+x''_0+y'_0+y''_0$	$(s''_0)_{\text{MFA1}}$	$(s'_0)_{\text{MFA1}}$	$(c_0)_{\text{MFA1}}$
0	0	0	0
1	0	1	0
2	1	1	0
3	0	1	1
4	1	1	1

Note that for a number '*A'* in carry save diminished-1, the case $a'_0=0$, $a''_0=1$ never occurs. Therefore, the equations of the outputs are as below,

$$(s'_0)_{MFA1}=x'_0+y'_0$$
$$(s''_0)_{MFA1}=(x'_0+y''_0)\cdot(x''_0+y'_0)\cdot\left(\overline{x''_0}+\overline{y'_0}+y''_0\right)\cdot\left(\overline{x'_0}+x''_0+\overline{y''_0}\right)$$
$$(c_0)_{MFA1}=x'_0\;y''_0+x''_0\;y'_0 \tag{4}$$

B. Addition Cell Design of First Digit of Madder#2 (MFA$_2$)

The operation of MFA$_2$ is to calculate ($x'_0+x''_0+y'_0+y''_0-1$). It produces two sum bits $(s'_0)_{\text{MFA2}}$ and $(s''_0)_{\text{MFA2}}$ and one output carry $(c_0)_{\text{MFA2}}$.

The range of the output is [-1, 3]. So, the MFA$_2$ output ranges from 0 to 4. The output '-1' occurs when $x'_0+x''_0+y'_0+y''_0=0$, that is, both of inputs are zero. In this case, the output of Madder#1 will be selected and the output of Madder#2 will be "don't care" shown by 'd' in Table III. Also the output '0' occurs when $x'_0+x''_0+y'_0+y''_0=1$, that is, one input is zero. Similarly, the output of Madder#1 will be selected and the output of Madder#2 will be "don't care". Different cases of this addition are represented in Table III.

TABLE III
MFA$_2$ Truth Table

$x'_0+x''_0+y'_0+y''_0$	$(s''_0)_{\text{MFA2}}$	$(s'_0)_{\text{MFA2}}$	$(c_0)_{\text{MFA2}}$
0	d	d	0
1	d	d	0
2	0	1	0
3	1	1	0
4	*0*	*1*	*1*

After simplifying the above table, the equations of the outputs are as below:

$$(s'_0)_{MFA2}='1'$$
$$(s''_0)_{MFA2}=(s''_0)_{MFA1}$$
$$(c_0)_{MFA2}=x''_0y''_0 \tag{5}$$

In other words, instead of applying a 3-operand adder, we design a 2-operand adder for Madder#2 which its first cell is MFA$_2$.

If *X*+*Y*= 2^n+1, then the output should be zero. In this case, *X* and *Y* are non zero. As shown earlier, if two inputs are non-zero, the outputs of MFA1 and MFA2 will be 1 and 2, respectively. Therefore, the output is not zero and it is necessary to remove this problem. Example 1 presents the problem.

Example. Let $m=2^4+1$, *X* = 5 and *Y* = 12 are two carry save diminished-1 numbers in this module. Then we have,

5 → 0 1 0 1
12 → 1 0 1 2

The outputs are shown in Fig. 7.

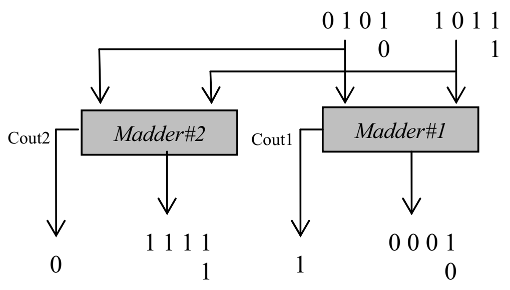

Fig. 7 The outputs of madder#1 and madder#2 for the example

As shown earlier, when two inputs are complement ($X+Y=2^n+1$), none of adders produce the correct output. In this case, the output of Madder#1 is always "0001" and the output of Madder#2 is "1112".

■

With a few modifications in the proposed method, the desired zero result can be produced. Two methods are proposed to solve this problem which they are introduced in the next sections. One architecture has less hardware overhead (CSD-RC1) and another one has less propagation delay (CSD-RC2).

C. The Proposed Architecture with Lower Area

As mentioned in the above example, if $X+Y=2^n+1$ then $C_{out1}=1$ and $C_{out2}=0$. One way to correct the output is to invert the s_0' of Madder#1. The solution is shown in Fig. 8.

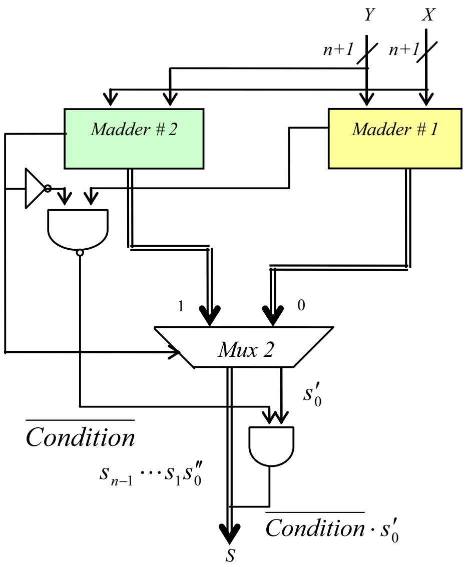

Fig. 8 The structure of CSD-RC1

Therefore, the final output can be corrected using the following equations:

$$Condition = C_{out1}\overline{C_{out2}}$$

$$s_0' = \overline{Condition}\cdot s_0' + Condition\cdot 0 = \overline{Condition}\cdot s_0' \qquad (6)$$

D. The Proposed Adder with Faster Architecture

The desired zero can be produced by the third input of one multiplexer. In other words, instead of using extra gates in Fig. 8, one 3×1 multiplexer is applied, where C_{out1} and C_{out2} are its select lines. The operation of the 3×1 multiplexer is described in Table IV.

TABLE IV
THE OPERATION OF 3 × 1 MULTIPLEXER

C_{out2}	C_{out1}	S
0	0	I_1
0	1	*"00…0"*
1	0	*don't care*
1	1	I_2

This structure is shown in Fig. 9.

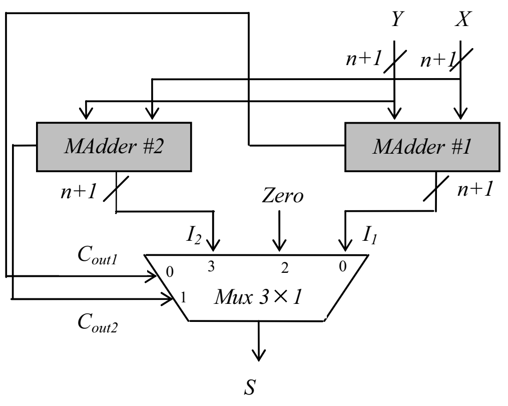

Fig. 9 The structure of CSD-RC2

IV. Comparisons

Two and three-operand adders of modulo 2^n+1 adders discussed in section 2 can be designed in different ways. They can be implemented using ripple-carry addition [21], [22], parallel-prefix addition or etc. Efficient parallel-prefix adders have been proposed in [15], [17]-[19], [21], [22]. If we want to design modulo 2^n+1 adder with ripple-carry adder, to achieve lower power and lower area, it is obvious that series and incrementer methods realize two carry propagations in series and thus they are slow, while the delay of parallel method is equal to one carry propagation.

TABLE V
REAL COMPARISON RESULTS FOR N=8

Adder Architecture	Transistor Count	Average Power Dissipation (µW)	Power–Delay Product (fJ)
PPREF [17]	1036	293.56	127.54
TPP [15]	844	278.64	156.80
CSD-PP [19]	838	214.51	50.53
CSA-RC	794	152.15	119.76
CSD-RC1	574	146.89	110.68
CSD-RC2	612	167.79	119.89

TABLE VI
IMPROVEMENTS OF CSD-RC1 AND CSD-RC2 VS. OTHER COMPARED ADDERS

	Hardware Improvement	Power Improvement	PDP Improvement
CSD-RC1 *vs.* PREF	≈ 45%	≈ 50%	13.22%
CSD-RC1 *vs.* TPP	≈ 32%	≈ 47%	29%
CSD-RC1 *vs.* CSD-PP	≈ 31.5%	≈ 32%	-119%
CSD-RC1 *vs.* CSA-RC	≈ 27.7%	≈ 3.46%	7.6%
CSD-RC2 *vs.* PREF	≈ 41%	≈ 43%	6%
CSD-RC2 *vs.* TPP	≈ 27.5 %	≈ 40%	24%
CSD-RC2 *vs.* CSD-PP	≈ 27 %	≈ 22%	-137%
CSD-RC2 *vs.* CSA-RC	≈ 23%	-10.3%	-0.1%

In diminished-1, there is a critical path composed of one modulo $2^n + 1$ adder as well as one multiplexer. Therefore, the parallel method is the fastest method with ripple-carry adder architecture. We call this structure as CSA-RC adder which has been shown in Fig. 4.

For hardware overhead evaluation, the transistor numbers of proposed CSD-RC1 and CSD-RC2 architectures and the CSA-RC adder have been considered. The CSA-RC adder has ($92\ n + 58$) transistors. The area of CSD-RC1 adder is equal to ($70\ n + 14$) transistors and the proposed CSD-RC2 adder offers hardware area equal to ($76\ n + 4$) transistors. So, CSD-RC1 has the least area among the modular adders.

Therefore, in this paper, CSA-RC adder based on ripple-carry adder and 3 efficient modulo $2^n + 1$ adders based on parallel-prefix adder [15], [17], [19] with conventional binary, diminished-1 and CSD-1 number systems are considered for area, power and power-delay-product (PDP) comparisons. We compare the proposed adders with CSA-RC, PPREF [17], TPP [15] and CSD-PP [19]. For the quantitative comparison, HSPICE software is used and all architectures are mapped to the 0.18 implementation technology (0.18 µm, Vdd=1.8 v).

We optimize our designs to achieve lower optimized PDP and transistor sizing is done (in a 1-bit full-adder cell and other components of whole architecture) to obtain more efficient PDP. The different designs consumptions measured at the same operating frequency.

As shown in table V, CSD-RC1 has the smallest implementation area is the lowest power consumption among the compared adders. The power reported by HSPICE includes static and dynamic power. Beside, it reduces power-delay-product parameter of CSA-RC, PPREF and TPP. But they don't improve power-delay product of CSD-PP.

Table VI clearly describes the comparisons. The proposed architectures lead to smaller hardware overhead than the CSA-RC adder. They both reduce average power consumption of all modulo 2^n+1 adders based on parallel-prefix addition. The proposed CSD-RC1 improves CSA-RC, PPREF, TPP and CSD-PP power consumption about 3.46%, 50%, 47% and 32% respectively. They decrease hardware overhead of the compared adders and they also reduce PDP parameter of CSA-RC, PPREF and TPP.

I. CONCLUSION

A novel ripple-carry addition algorithm for carry save diminished-1 number system has been proposed. This allows for a power and area optimized solution in application that lower speed of ripple-carry addition may be tolerated. Two new architectures have been presented for designing modulo 2^n +1 addition based on ripple-carry adder. The CSD-RC1 architecture is a fast architecture whereas the CSD-RC2 architecture applies less hardware than the first one. For the quantitative comparison, HSPICE software has been used. VLSI implementations of these architectures have revealed that the proposed adders have better efficiency than previous solutions based on ripple-carry and parallel-prefix addition in both implementation area requirements and power consumptions. The proposed architectures lead to faster design and smaller hardware overhead than the CSA-RC adder. They both reduce average power consumption of all modulo 2^n+1 adders based on parallel-prefix addition. The proposed CSD-RC1 improves CSA-RC, PPREF, TPP and CSD-PP power consumption about 3.46%, 50%, 47% and 32% respectively. They also decrease hardware overhead of the compared adders and reduce PDP parameter of CSA-RC, PPREF and TPP.

References

[1] H. Garner, "The residue number system," IRE Trans. Electronic Computer, vol. EC-8, pp.140-147, Jun.1959.

[2] N. Szabo and R. Tanaka, "Residue arithmetic and its applications to computer technology," New York, LCCCN: 66-15186, McGraw-Hill Book Company, 1967.

[3] R. Conway and J. Nelson, "Improved RNS FIR Filter Architectures", IEEE Trans. On Circuits and Systems-II: Express Briefs, vol. 51, no. 1, Jan. 2004.

[4] P. G. Fernandez, et al., "A RNS-Based Matrix-Vector-Multiply FCT Architecture for DCT Computation," Proc. 43th IEEE Midwest Symposium on Circuits and Systems, pp. 350-353, 2000.

[5] L. Yang and L. Hanzo, "Redundant Residue Number System Based ERROR Correction Codes", IEEE VTS 54th on Vehicular Technology Conference, vol. 3, pp. 1472 – 1476, 7-11 Oct. 2001.

[6] J. Ramirez, et al., "Fast RNS FPL-Based Communications Receiver Design and Implementation," Proc. 12th Int'l Conf. Field Programmable Logic, pp. 472-481, 2002.

[7] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems," Comm. ACM, vol. 21, no. 2, pp. 120-126, Feb. 1978.

[8] J. Bajard, and L. Imbert, "A Full RNS Implementation of RSA," IEEE Transactions on Computers, vol. 53, no. 6, pp. 769-774, Jun 2004.

[9] A. S. Molahosseini, K. Navi, O. Hashemipour, A. Jalali, "An efficient architecture for designing reverse converters based on a general three-moduli set", Journal of Systems Architecture, no. 54, pp. 929–934, 2008.

[10] M. Hosseinzade, S. Timarchi, and K. Navi, "Multi Level Residue Number System with Moduli Set of $(2^n, 2^n-1, 2^{n-1}-1)$", 12th International CSI Computer Conference, 20-22 Feb. 2007.

[11] A. Hariri, K. Navi, and R. Rastegar, "A New High Dynamic Range Moduli Set with Efficient Reverse Converter", Elsevier Journal of Computers & Mathematics with Applications, vol. 55, Issue 4, pp. 660-668, Feb. 2008.

[12] M. Hosseinzadeh, K. Navi and S. Timarchi, "Design of Current Mode Circuits of Residue Number Systems", 14th Iranian Conference of Electrical Engineering (ICEE'2006), 16-18 May 2006.

[13] M. Hosseinzadeh, A. Sabbagh, K. Navi, "A Fully Parallel Reverse Converter", International Journal of Electrical, Computer, and Systems Engineering, vol. 1, no. 3, 2007.

[14] A. S. Molahosseini, and K. Navi, "New arithmetic Residue to Binary converters", IJCSES International Journal of Computer Sciences and Engineering Systems, vol. 1, no.4, pp. 295-299, October 2007.

[15] C. Efstathiou, H. T. Vergos and D. Nikolos, "Fast Parallel-Prefix 2^n+1 Adder", IEEE Trans. On Computers, vol. 53, no. 9, Sep. 2004.

[16] L. M. Leibowitz, "A Simplified Binary Arithmetic for the Fermat Number Transform," IEEE Trans. Acoustics, Speech, Signal Processing, vol. 24, pp. 356-359, 1976.

[17] H.T. Vergos, et al., "Diminished-1 Modulo 2^n+1 Adder Design," IEEE Trans. Computers, vol. 51, pp. 1389-1399, 2002.

[18] R. Zimmermann, "Efficient VLSI Implementation of Modulo (2^n±1) Addition and Multiplication," Proc. 14th IEEE Symp. Computer Arithmetic, pp. 158-167, Apr. 1999.

[19] S. Timarchi, O. Kavehei, and K. Navi, "Low Power Modulo 2^n+1 Adder Based on Carry Save Diminished-1 Number System," American Journal of Applied Sciences 5 (4), pp. 312-319, 2008.

[20] S. Timarchi and K. Navi, "A Novel Modulo 2^n+1 Adder Scheme", 12th International CSI Computer Conference, 20-22 Feb. 2007.

[21] M. Bayoumi and G. Jullien, "A VLSI Implementation of Residue Adders," IEEE Trans. Circuits Systems, vol. 34, pp. 284-288, 1987.

[22] A. A. Hiasat, "High-Speed and Reduced Area Modular Adder Structures for RNS," IEEE Trans. Computers, pp. 84-89, 2002.

[23] M. Hosseinzadeh, K. Navi, and S. Timarchi, "New Design of RNS High Speed Multi Operand Adder," 14th Iranian Conference of Electrical Engineering, 16-18 May 2006.

[24] S. Timarchi, K. Navi, and M. Hosseinzade, "New Design of RNS Subtractor for modulo 2^n+1," 2nd IEEE International Conference on Information & Communication Technologies: From Theory to Application, 24-28 Apr. 2006.

[25] B. Parhami, "RNS Representation with Redundant Residues", Proc. of the 35th Asilomar Conf. on Signals, Systems, and Computers, Pacific Grove, CA, pp. 1651-1655, 4-7 Nov. 2001.

[26] C. Efstathiou, H. T. Vergos, and D. Nikolos, "Handling Zero in Diminished-1 Modulo 2^n+1 Adders," International Journal of Electronics, vol. 90, no. 2, pp. 133-144, Feb. 2003.

Somayeh Timarchi received her B.Sc. and M.Sc. in Computer Hardware Engineering from Shahid Beheshti University in 2002 and Sharif University of Technology in 2004, respectively. Currently, she is a Ph.D. student in faculty of electrical and computer engineering of Shahid Beheshti University under supervision of Dr. Keivan Navi. She is currently part time instructor in the Department of Electrical and Computer Engineering at Shahid Beheshti University,GC. Her research interests include Computer Arithmetic, Residue and Redundant Number System, VLSI Design, and Computer Architecture.

Keivan Navi received his B.Sc and M.Sc in Computer Hardware engineering from Beheshti University in 1987 and Sharif University of Technology in 1990, respectively. He also received his Ph.D. of Computer Architecture from Paris XI university in 1995. He is currently associate professor in faculty of electrical and computer engineering of Beheshti University,GC. His research interests include Residue Number System, VLSI design, Single Electron Transistors (SET), Carbon Nano Tube, Interconnection Network and Quantum Cryptography.