On the Use of Diminished-1 Adders for Weighted Modulo $2^n + 1$ Arithmetic Components

H. T. Vergos
Computer Engineering & Informatics Dept.
University of Patras, 26500 Patras, Greece
vergos@ceid.upatras.gr

D. Bakalis
Department of Physics
University of Patras, 26500 Patras, Greece
bakalis@physics.upatras.gr

Abstract

The contribution of this paper is twofold. We firstly show that an augmented diminished-1 adder can be used for the modulo $2^n + 1$ addition of two n-bit operands in the weighted representation, if it is driven by operands whose sum has been decreased by 1. This scheme outperforms solutions that are based on the use of binary adders and / or weighted modulo $2^n + 1$ adders in both area and delay terms. We then apply this scheme in the design of residue generators (RGs) and multi-operand modulo adders (MOMAs). The resulting arithmetic components remove at least a whole parallel adder out of the critical path of the currently most efficient proposals. Experimental results indicate savings of more than 30% in execution time and of approximately 19% in implementation area when the proposed architectures are used.

1. Introduction

Modulo $2^n + 1$ arithmetic has been used in a variety of applications, ranging from pseudorandom number generation and cryptography [14] up to convolution computations without round-off errors [13]. A channel performing arithmetic operations modulo $2^n + 1$ is most commonly an integral part of almost every residue number system (RNS) [19]. The RNS is an arithmetic system well-suited to applications in which the operations are limited to addition, subtraction and multiplication. The adoption of an RNS can offer significant speedup over the binary system and has therefore been reported in the design of digital signal processors [17, 19], FIR filters [4] and communication components [16].

Two different representations have been studied for the modulo $2^n + 1$ operands; the common weighted one and the diminished-1. An arithmetic component that performs a modulo $2^n + 1$ operation will hereafter be denoted as weighted or diminished-1 component depending on the representation used for its inputs and outputs. The weighted representation has the disadvantage that it requires $n+1$ bits for representing each operand while it utilizes only the $2^n + 1$ combinations. In the most common three-moduli RNS that uses channels of the $\{2^n - 1, 2^n, 2^n + 1\}$ form, the execution delay is therefore dictated by the modulo 2^n+1 channel. On the other hand, the diminished-1 representation [12] dictates that each operand is represented decreased by one compared to its weighted representation. Zero operands are not used in the computation channel; the results are derived alternatively when any operand or the result is zero. Therefore only n-bit operands are used in a diminished-1 channel, leading to smaller and faster components. However, the use of the diminished-1 representation involves the overhead of translators from and to the weighted system. Intermediate results do not have to be translated immediately back to the weighted system; therefore, if a significant number of computations takes place before a new translation is required, the use of the diminished-1 system may be profitable. Several architectures have been proposed for modulo $2^n + 1$ arithmetic components for each of the two representations, including parallel-adders [2, 5, 7, 9, 22, 23], multi-operand adders [1, 3, 15] and residue generators [1, 15].

In this paper we show that a diminished-1 adder with minor modifications can be also used for the modulo 2^n+1 addition of n-bit weighted operands, provided that it is driven by operands whose sum has been decreased by 1. The required modifications do not increase the execution delay of the adder and can be implemented in low area. Since currently, the most efficient architectures for a diminished-1 adder outperform those for weighted operands in both area and delay terms, the augmented diminished-1 adder can be used very efficiently if the decreased sum of the input operands can be easily derived. We show that this is the case in both the residue generators and the weighted multi-operand modulo adders cases.

2 The Augmented Diminished-1 Adder

2.1 Weighted Modulo $2^n + 1$ Addition Background

A number of different architectures can be followed for the design of a modulo 2^n+1 weighted adder; some of them stem from the general residue adder case, whereas others are dedicated architectures for this particular modulus.

For the modulo $2^n + 1$ addition of A and B, hereafter denoted by $|A+B|_{2^n+1}$, where $A = a_n a_{n-1} a_{n-2} \cdots a_1 a_0$ and $B = b_n b_{n-1} b_{n-2} \cdots b_1 b_0$ are two $(n + 1)$-bit binary numbers in the range $[0, 2^n]$, we have that :

$$|A+B|_{2^n+1} = \begin{cases} A+B-(2^n+1), & \text{if } A+B \geq 2^n+1 \\ A+B, & \text{otherwise.} \end{cases} \quad (1)$$

Following the general residue adder architecture presented in [2], we can implement (1) by using two binary adders connected in series and a multiplexer. A $(n + 1)$-bit adder is used to compute $A + B$, while a $-(2^n + 1)$ correction is added to its output by the second $(n + 2)$-bit adder. The multiplexer is then used to select between the two adders' results depending on the value of the carry output of the second adder. In [5], Dugdale has reduced the width of the second adder to $(n + 1)$ bits and has shown that the multiplexers can be controlled by the logical OR of the two adders' carry outputs. She has also presented an area efficient architecture that performs the modular addition using just one adder in two addition cycles. Both these architectures are very slow. An obvious solution for decreasing the delay of the above architectures is to have both cases of (1) computed in parallel [21]. This solution however, apart from the two adders requires the addition of a carry-save adder (CSA) stage. A more area effective solution was proposed in [9], by observing that most carry propagate and generate signals for both cases of (1) are common and therefore an augmented single carry look-ahead (CLA) unit is sufficient. Finally, in [7] parallel-prefix weighted adders have been presented. These have been shown to be the fastest available and more efficient than the proposal of [9].

2.2 Diminished-1 Modulo $2^n + 1$ Addition Background

An even larger number of architectures is available for the design of a diminished-1 adder since its operation has been shown to be equivalent to an inverted end-around carry (EAC) binary adder [22, 23]. [22] proposed single and multiple level CLA architectures. Parallel-prefix architectures have been proposed in [22, 23]. The parallel-prefix diminished-1 adders of [22] in particular, offer the minimum number of prefix levels reported in the open literature, which is equal to that required by the fastest binary adders as well. As a consequence, a diminished-1 adder can be designed to operate as fast as a parallel-prefix binary adder.

Table 1 summarizes the area and delay requirements in equivalent gates of some weighted and diminished-1 architectures (ignore the last part at this point). These estimates are derived using the unit-gate model [20]. We assume that both addition operands are in the $[0, 2^n - 1]$ range (the 2^n operand value is not considered at this point) and that all binary adders follow the Kogge-Stone [11] parallel-prefix carry computation architecture. The area and delay estimates derived for n=4, 8, 16 or 32 are given in Table 2.

From the estimates it becomes obvious that, considering the current state of the art, a diminished-1 adder is both a smaller and faster circuit than a weighted modulo $2^n + 1$ adder. Therefore, if we could use a diminished-1 adder with minor modifications to also perform weighted modulo $2^n + 1$ addition, we would reduce both the area and the delay of the resulting components. In the next subsection we show that this is possible.

2.3 Using a Diminished-1 Adder for Weighted Addition

Let A and B represent two n-bit operands in the $[0, 2^n - 1]$ range. Let $A^\star$ and $B^\star$ denote two n-bit vectors such $A^\star + B^\star = A + B - 1$. According to (1), we then have that :

$$|A+B|_{2^n+1} = \begin{cases} A+B-(2^n+1), & \text{if } A+B \geq 2^n+1 \\ A+B, & \text{otherwise} \end{cases}$$

or equivalently that :

$$|A+B|_{2^n+1} = \begin{cases} (A+B-1)-2^n, & \text{if } A+B-1 \geq 2^n \\ (A+B-1)+1, & \text{otherwise.} \end{cases} \quad (2)$$

Taking the modulo 2^n of (2) and using $A^\star$ and $B^\star$ we then get :

$$\left||A+B|_{2^n+1}\right|_{2^n} = \begin{cases} (A^\star+B^\star), & \text{if } (A^\star+B^\star) \geq 2^n \\ (A^\star+B^\star)+1, & \text{otherwise.} \end{cases} \quad (3)$$

Let c_{out} denote the carry output of the $(A^\star + B^\star)$ n-bit integer addition. Using it in (3), we can unify the two cases as :

$$\left||A+B|_{2^n+1}\right|_{2^n} = |A^\star+B^\star|_{2^n} + \overline{c_{out}} \quad (4)$$

Relation (4) indicates that we can derive the n least significant bits of the weighted modulo 2^n+1 addition of A and B by using an inverted end-around carry adder (equivalently, a diminished-1 adder) provided that the sum of its inputs is decreased by 1, that is, if we use as inputs the $A^\star$ and $B^\star$ vectors.

The most significant bit of the weighted addition of A and B should be 1, only when $|A+B|_{2^n+1} = 2^n$, which

Table 1. Area and delay estimations provided by the unit-gate model

Weighted modulo 2^n+1 adders		
Architecture	*Delay*	*Area*
[2]	$2\log n + 2\log(n+1) + 8$	$\frac{3}{2}n\log n + \frac{3}{2}(n+1)\log(n+1) + 13n + 8$
[21]	$2\log n + 7$	$\frac{3}{2}n\log n + \frac{3}{2}(n+1)\log(n+1) + 16n + 8$
[7]	$2\log n + 7$	$\frac{9}{2}n\log n + 4n + 11$
Diminished-1 modulo 2^n+1 adders		
Architecture	*Delay*	*Area*
[22]	$2\log n + 3$	$\frac{9}{2}n\log n + \frac{1}{2}n + 5$
Proposed augmented diminished-1 adder for n-bit weighted modulo 2^n+1 addition		
Architecture	*Delay*	*Area*
Proposed	$2\log n + 3$	$\frac{9}{2}n\log n + \frac{3}{2}n + 4$

Table 2. Derived area and delay estimates

Weighted modulo 2^n+1 adders								
	$n=4$		$n=8$		$n=16$		$n=32$	
Architecture	*Delay*	*Area*	*Delay*	*Area*	*Delay*	*Area*	*Delay*	*Area*
[2]	18	95	22	202	26	440	30	961
[21]	11	107	13	226	15	488	17	1057
[7]	11	63	13	151	15	363	17	859
Diminished-1 modulo 2^n+1 adders								
	$n=4$		$n=8$		$n=16$		$n=32$	
Architecture	*Delay*	*Area*	*Delay*	*Area*	*Delay*	*Area*	*Delay*	*Area*
[22]	7	44	9	118	11	302	13	742
Proposed augmented diminished-1 adder for weighted modulo 2^n+1 addition								
	$n=4$		$n=8$		$n=16$		$n=32$	
Architecture	*Delay*	*Area*	*Delay*	*Area*	*Delay*	*Area*	*Delay*	*Area*
Proposed	7	47	9	125	11	317	13	773

since $0 \leq A,\ B \leq 2^n-1$ reduces to $A^\star + B^\star = 2^n - 1$, that is, when $A^\star$ and $B^\star$ are bit-wise complementary. This condition can be easily detected as the logical AND of the XOR of the bits of $A^\star$ and $B^\star$ with the same weight. Since in every fast adder architecture there is a preprocessing stage that computes the half-sum terms, that is, the XOR of the corresponding input operands bits, the extra hardware required for deriving the most significant bit of the weighted addition is small. Since this operation, according to the unit-gate model, can be completed by a tree of two-input gates in $\log n + 2$ time units, while the diminished-1 adder computes the rest bits in $2\log n + 3$ time units, it does not add any delay on the critical path of the diminished-1 adder. In some adder cases (known as XOR adders) the half sum term is also used as the carry propagate term. The group propagate terms in these adders are the logical AND of the half-sum terms and therefore no extra hardware is required for the derivation of the most significant bit.

Figure 1 presents the architecture that results from the previous analysis. A translator circuit accepts the n-bit vectors A and B and provides the vectors $A^\star$ and $B^\star$. These are driven to an augmented diminished-1 adder that is capable of providing the $(n+1)$-bit sum of the weighted modulo 2^n+1 addition of A and B. The last parts of Tables 1 and 2 provide area and delay estimates for the augmented diminished-1 adder of Figure 1, assuming that the diminished-1 adder is designed following the architecture of [22]. The estimates of Table 2 indicate that the augmented diminished-1 adder also offers implementation area and execution delay savings over every architecture proposed for weighted addition. We therefore conclude that its use is very attractive if the translator circuit of Figure 1 can be designed efficiently. In the following section, we show that for two weighted arithmetic components, namely the residue generator and the multi-operand weighted modulo adder, the translator required is either extremely small or

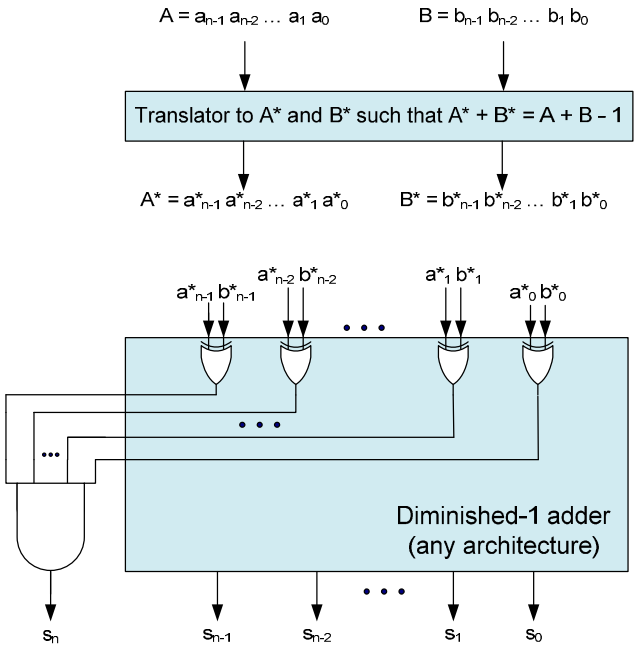

Figure 1. Using an augmented diminished-1 adder for n-bit weighted addition.

nothing at all. During the multi-operand weighted modulo adder case, we further examine how we can handle $(n+1)$-bit operands as well.

3 Applications

In this section we show that the adder proposed in Figure 1 can be easily incorporated in the design of weighted residue generators (RGs) and multi-operand modulo adders (MOMAs). The notations RG$(k, 2^n + 1)$ and MOMA$(k, 2^n + 1)$ are used hereafter to indicate a circuit that produces the residue of a k-bit weighted number A, in modulo $2^n + 1$ arithmetic and a weighted multi-operand modulo $2^n + 1$ adder for k operands respectively. The notation n-bit MOMA is used to denote a MOMA$(k, 2^n + 1)$, with n-bit wide input operands.

3.1 Novel Residue Generators

In a modulo $2^n + 1$ arithmetic component, irrespectively of the representation used, every operand has to be expressed in modulo $2^n + 1$ arithmetic before it can be used. This is done by a residue generator circuit. Although a divider can be employed for attaining the residue of A taken modulo 2^n+1, solutions that require a divider are very slow and therefore inefficient. Faster and smaller residue generators can be devised following the proposals of [1, 15].

In [1] an RG$(k, 2^n + 1)$ is proposed, in which all the residues of the powers 2^i taken modulo 2^n+1, with $0 \leq i \leq (k-1)$, are pre-computed. Each such residue is logically ANDed with the corresponding bit i of the input operand. The resulting logical products are then driven to a weighted MOMA$(k, 2^n + 1)$ that provides the residue.

The area and time complexity for the RG circuit can be further reduced by taking advantage of the periodic properties of the powers of 2 taken modulo $2^n + 1$ [15]. The residue $|A|_{2^n+1}$ was proposed in [15] to be computed by the following steps :

1. partition the k bits in groups suppose $g_0, g_1, \ldots g_{\lceil \frac{k}{n} \rceil}$, of n bits each, starting from the least significant bits. If the last group contains less than n bits, then 0s are assumed at the most significant bit positions.

2. invert the bits of the odd numbered groups and account a correction factor of 2 for every inverted group. If the last group is an odd numbered one and incomplete at bit positions $d, d+1, \ldots, n-1$ also account a correction factor of $\sum_{i=d}^{n-1} 2^i$.

3. use an inverted EAC CSA tree to add the numbers of the even and the inverted odd groups along with a required total correction factor, until two n-bit final operands are derived. The use of an inverted EAC CSA tree for reducing the operands in two final summands is justified by observing that a carry output at the most significant bit position, suppose c_n, that has a weight of 2^n, can be complemented and added at the least significant bit position in the next stage, provided that a correction equal to -1 is taken into account, since it holds that :

   $$|c_n 2^n|_{2^n+1} = |-c_n|_{2^n+1} = |2^n + \overline{c_n}|_{2^n+1} = |\overline{c_n} - 1|_{2^n+1} .$$

   The total correction factor is equal to E, with $0 \leq E \leq 2^n$. E does not only include the correction required due to step 2 above, but also incorporates the correction due to the inverted EAC CSA tree itself.

4. use a final weighted adder, consisting of two binary adders connected in series and a multiplexer [2], to derive the residue.

The augmented diminished-1 adder of Figure 1 can obviously replace the final weighted adder of step 4, just by using $|E-1|_{2^n+1}$ instead of E as the total correction factor. When E is 0, no correction factor is at all required in the proposed architecture. This is justified by observing that in modulo 2^n+1 arithmetic an inverted EAC CSA addition of A, B with 0 results into $A + B + 1$, that is, it increases the sum of the operands by 1 modulo $2^n + 1$. Therefore, to derive the two final summands decreased by 1, we can just not add any correction factor. Note that in this case apart

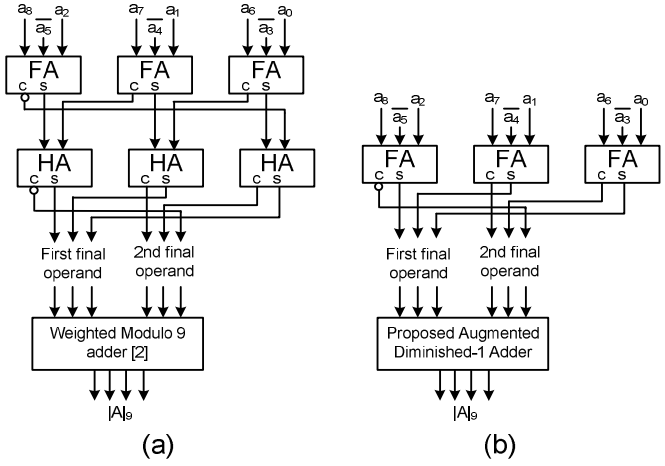

Figure 2. An RG$(9, 2^3+1)$ designed (a) according to [15] and (b) according to the proposed method.

from the area and time savings because of the replacement of the final weighted adder with the augmented diminished-1 adder of Figure 1, there are also area savings in the inverted EAC CSA tree, since it has one input operand less and maybe further time savings, if the CSA tree depth is reduced due to the reduction of the operands.

Example 1. Consider the design of an RG$(9, 2^3+1)$ for $A = a_8a_7 \dots a_0$. According to [15] the input bits are partitioned in three groups : $g_0 = \{a_2, a_1, a_0\}$, $g_1 = \{a_5, a_4, a_3\}$ and $g_2 = \{a_8, a_7, a_6\}$. The bits of g_1 are then inverted and along with the bits of the other groups and a total correction factor are the inputs of an inverted EAC CSA tree. The total correction factor that the architecture of [15] requires is 0, but its addition can not be omitted since this would alter the number of inverted EACs. The adder tree provides two final summands that are the inputs of a weighted modulo $2^3 + 1$ adder designed according to [2]. The design of the RG according to [15] is shown in Figure 2.a. Blocks labeled as HA and FA indicate half and full adders, respectively. In the proposed architecture we do not need to add any correction and therefore the number of the input operands of the adder tree is reduced. The derived RG is shown in Figure 2.b. Apart from the savings in both area and time because of the use of the proposed augmented diminished-1 adder instead of the proposal of [2], savings in both area and time result from the simplified adder tree required. Considering the proposed architecture and as an example the value $A = 143_{10} = 010001111_2$, we have that the final operands at the inputs of the proposed augmented diminished-1 adder are 011 and 100. Since these are complimentary, the most significant bit of the residue is 1. The rest bits are computed by the diminished-1 addition of the final operands, which provides 000 at the least significant bit positions. We therefore get $|143|_{2^3+1} = 1000_2 = 8_{10}$. □

3.2 Novel Weighted Multi-operand modulo $2^n + 1$ Adders

One of the arithmetic components that has been heavily researched in residue arithmetic is the multi-operand modulo adder (MOMA). Hardware support for multi-operand modulo addition is highly appreciated in several multiply-and-add intensive computations, such as digital filtering, convolution estimation and FFT transforms. The first effort for a weighted MOMA appeared in [18], but required several parallel-adders connected in series. The problem of designing MOMAs for generalized moduli was attacked in [1, 8, 10, 15]. The architecture of [15] is currently considered the most efficient for modulo $2^n + 1$. It uses an inverted EAC CSA tree for reducing the k summands along with a total correction factor into two final operands. Unfortunately, for the addition of the latter it requires two parallel adders connected in series since it uses the architecture of [2] as the final weighted adder. One can achieve significant savings in both area and delay by replacing the final adder of [15] with the augmented diminished-1 adder of Figure 1. The modifications required for this replacement are analyzed in detail below and are shown to be small enough.

We first consider that the input operands are n bits wide. According to the analysis of section 2.3, if two n-bit weighted operands are driven to a diminished-1 adder, this will output their modulo $2^n + 1$ sum increased by 1. Suppose now the addition of k n-bit weighted operands. This can obviously be achieved by diminished-1 additions, that will provide the modulo $2^n + 1$ sum increased by $(k - 1)$ (irrespectively if the additions are carried out in parallel or in series). For deriving the correct sum, we can use one further addition (in fact a subtraction). Since however this will also increase by 1 the expected sum, in this last addition we have to add a correction factor of $|-k|_{2^n+1}$. Obviously, if $|-k|_{2^n+1} = |-1|_{2^n+1} = 2^n$ this last addition is not required. Finally, a $(n + 1)$-bit weighted result can be derived by using an augmented diminished-1 adder for the last addition.

Example 2. Consider the multi-operand modulo $2^3 + 1$ weighted addition of the 3-bit operands $A = 7_{10} = 111_2$, $B = 4_{10} = 100_2$, $C = 5_{10} = 101_2$ and $D = 0_{10} = 000_2$. Adding A with B and C with D in diminished-1 adders, will provide the sums $S_1 = 3_{10} = 011_2$ and $S_2 = 6_{10} = 110_2$, respectively. Then, adding these together by a diminished-1 adder will provide $R = 1_{10} = 001_2$. Finally, a correction factor of $E = |-k|_9$ is added. Since $k = 4$, we have that $E = |-k|_9 = 5_{10} = 101_2$. The addition of R with E, in an augmented diminished-1 adder

yields the expected result $0111_2 = 7_{10}$. □

Example 3. Consider the multi-operand modulo $2^2 + 1$ weighted addition of the 2-bit operands A, B, C, D, E and F, with $A = B = D = F = 1_{10} = 01_2$, $C = 3_{10} = 11_2$ and $E = 0_{10} = 00_2$. Adding A with B, C with D and E with F in diminished-1 adders, will provide the sums $S_1 = 3_{10} = 11_2$, $S_2 = 0_{10} = 00_2$ and $S_3 = 2_{10} = 10_2$ respectively. The diminished-1 addition of S_2 and S_3 will result in $S_4 = 3_{10} = 11_2$. Since there are 6 input operands and $|-6|_{2^2+1} = |-1|_{2^2+1} = 4$, no correction factor is required. To obtain the result we need to add S_1 and S_4 in an augmented diminished-1 adder that will provide the expected result $010_2 = 2_{10}$. □

Instead of using in series or parallel diminished-1 adders, the multi-operand diminished-1 addition described above can be achieved by a diminished-1 MOMA. In [6] it was shown that a diminished-1 MOMA can be implemented by an inverted EAC CSA tree and a final diminished-1 adder. From the above analysis, we conclude that an n-bit weighted MOMA$(k, 2^n + 1)$, can be derived by using a diminished-1 MOMA$(k + 1, 2^n + 1)$ using :

- the correction factor E which is equal to $|-k|_{2^n+1}$, as the $(k + 1)$-th summand and
- the augmented diminished-1 adder introduced earlier as the final adder.

As explained earlier, a diminished-1 MOMA$(k, 2^n + 1)$ is required when $|-k|_{2^n+1} = |-1|_{2^n+1} = 2^n$.

In the following we extend the above scheme to also account for $(n+1)$-bit operands. Since in modulo 2^n+1 arithmetic the most significant bit of an operand is 1 only when the value of the operand is 2^n and given that $|2^n|_{2^n+1} = -1$, we can still use in the previously described MOMA the n least significant bits of these operands and decrease the correction factor E according to the number of operands that have their most significant bit equal to 1. That is, if only one operand is equal to 2^n, the correction factor should be $|-k-1|_{2^n+1}$, if two operands are equal to 2^n, the correction factor should be $|-k-2|_{2^n+1}$ and so on. Therefore, a combinational circuit must be used which accepts the most significant bits of the k operands and outputs the correction factor that should be used.

Example 4. Consider the multi-operand modulo $2^3 + 1$ addition of the 4-bit operands $A = 8_{10} = 1000_2$, $B = 4_{10} = 0100_2$, $C = 6_{10} = 0110_2$ and $D = 3_{10} = 0011_2$. Let a_3, b_3, c_3 and d_3 denote the most significant bits of the input operands respectively and let $a_2a_1a_0$, $b_2b_1b_0$, $c_2c_1c_0$, $d_2d_1d_0$ denote their rest bits. Figure 3 presents the proposed implementation for a weighted MOMA$(4, 2^3 + 1)$. On the left of the figure, a small circuit based on half adder blocks (labeled HA in Figure 3) and simple logic gates is used to derive the correction factor that should be used. The circuit is actually a modified 1s counter. The inverted EAC CSA tree is shown on the right. It reduces the summands in two final operands that are added in an augmented diminished-1 adder. For the values of our example we derive that the correction factor $E = e_2e_1e_0$ computed by the circuit is $4 = |-4-1|_{2^3+1}$. When used in the adder tree, the two final operands that result are 000_2 and 010_2. The augmented diminished-1 adder then produces the expected sum $0011_2 = 3_{10}$. □

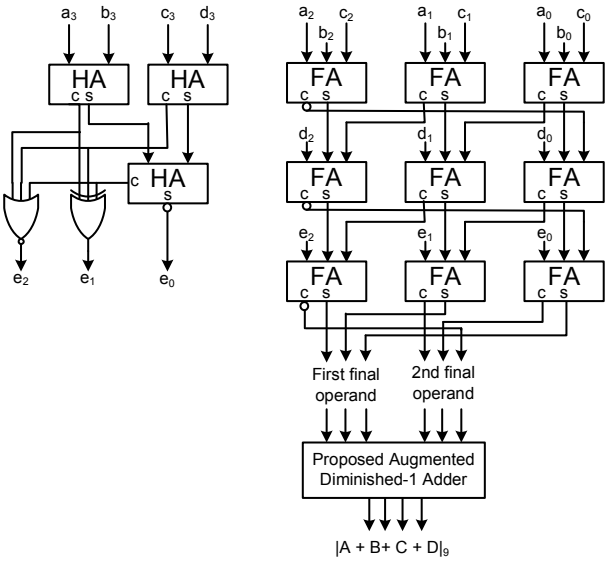

Figure 3. Proposed weighted MOMA$(4, 2^3+1)$.

Finally, it should be noted that :

- since the correction factor is derived later than the rest of the summands it should be driven to the last stage of the inverted EAC CSA tree.
- since the correction factor can always be derived by an 1s counter with a translator at its outputs composed of single gates, it can always be derived earlier than needed in the CSA tree and therefore does not add any delay on the critical path.
- since the correction factor can receive all values in the $[|-2k|_{2^n+1},\ |-k|_{2^n+1}]$ range, in few MOMA cases a value of 2^n is possible. In these cases the combinational circuit must be designed so as to provide a $(n + 1)$-bit correction factor. The most significant bit of this correction factor, should be used to control a multiplexer that drives the inputs of the augmented diminished-1 adder. If this bit is 0, then the inputs of the augmented diminished-1 adder are driven by the last stage of the CSA tree, that is, the stage used to add the remaining bits of the correction factor. If this bit is 1, then the inputs of the augmented diminished-1 adder are driven directly by the inputs of the last stage

Table 3. RG experimental results

RG		[15]		Proposed		Savings	
k	n	Delay (ns)	Area (μm^2)	Delay (ns)	Area (μm^2)	Delay (%)	Area (%)
8	4	2.04	5228	1.05	3590	48.5	31.3
16	4	2.77	8708	1.83	6928	33.9	20.4
32	4	3.16	15774	2.17	14250	31.3	9.7
16	8	2.43	11151	1.27	7940	47.7	28.8
32	8	3.17	18030	2.05	14506	35.3	19.5
64	8	3.54	32301	2.40	28878	32.2	10.6
32	16	2.87	23690	1.49	17250	48.1	27.2
64	16	3.64	37037	2.26	30415	37.9	17.9
128	16	4.00	66220	2.62	59361	34.5	10.4

Table 4. MOMA experimental results

MOMA		[15]		Proposed		Savings	
k	n	Delay (ns)	Area (μm^2)	Delay (ns)	Area (μm^2)	Delay (%)	Area (%)
4	4	2.93	10155	1.90	7655	35.2	24.6
8	4	3.58	18856	2.59	15550	27.7	17.5
12	4	4.01	26373	3.13	23528	21.9	10.8
4	8	3.30	19567	2.07	15010	37.3	23.3
8	8	3.94	35647	2.77	29386	29.7	17.6
12	8	4.34	50230	3.28	42676	24.4	15.0
4	16	3.68	39379	2.27	30252	38.3	23.2
8	16	4.36	70444	2.99	57711	31.4	18.1
12	16	4.79	97505	3.44	83942	28.2	13.9

of the CSA tree, that is, without the addition of any correction factor.

4 Comparisons

In this section, we compare the architectures derived in Section 3 for residue generators and multi-operand modulo adders against those proposed in [15].

The proposed architectures as well as the architectures of [15] were described in HDL for 9 different pairs (k, n) of the number of bits k (in case of RGs) or of the number of operands k (in case of $(n+1)$-bit MOMAs) and the word length n. All binary adders used in these descriptions follow the Kogge-Stone [11] parallel-prefix carry computation architecture whereas all diminished-1 adders used follow the parallel-prefix carry computation of [22]. After simulating the resulting descriptions, the designs were mapped to a CMOS standard cell library (180nm, 6-metal layer, 1.8 V) assuming typical case process parameters. A bottom-up approach was followed during mapping. Once a hierarchy level was mapped and optimized for delay and area, "don't touch" primitives were applied to it, for preserving the description of each architecture as much as possible. The derived results are given in Tables 3 and 4. All delay results are expressed in ns whereas all area results are expressed in μm^2.

The results show that in every examined case the proposed designs outperform those of [15] in both delay and area. The proposed RG circuits are on the average 39% faster and 20% smaller than those of [15], while the proposed MOMA circuits are 31% faster and 18% smaller.

5 Conclusions

This paper has shown that we can replace any weighted modulo $2^n + 1$ adder for n-bit operands with an augmented diminished-1 adder that will output the correct result provided that the sum of its input operands has been decreased by 1. Given the current state of the art in the architectures proposed for weighted and diminished-1 adders, this replacement leads to smaller and faster designs provided that the decrement of the input operands can be performed with small or no cost.

We examined two applications of this replacement,

namely residue generators and multi-operand weighted modulo adders. In the first case, no circuitry is required at all for decreasing the input operands' sum; moreover, in some cases it is more efficient in both area and delay terms to derive the decreased input operands' sum than the weighted operands' sum. In the second case, a small combinational circuit is required that does not add any delay on the critical path of the multi-operand modulo adder.

The experimental results, derived by implementing the proposed residue generators and multi-operand weighted modulo adders in static CMOS, indicate that on the average, the proposed approach for residue generators offers approximately 39% and 20% savings in the delay and the area compared to the currently most efficient proposal, respectively. The corresponding savings for multi-operand modulo adders are 31% and 18% on the average.

The application of the augmented diminished-1 adder in other weighted modulo components is currently under investigation.

References

[1] G. Alia and E. Martineli. Designing Multioperand Modular Adders. *Electronics Letters*, 32:22–23, 1996.

[2] M. Bayoumi and G. Jullien. A VLSI Implementation of Residue Adders. *IEEE Trans. Circuits Syst.*, 34:284–288, 1987.

[3] B. Cao, C.-H. Chang, and T. Srikanthan. A New Formulation of Fast Diminished-1 Multioperand Modulo $2^n + 1$ Adder. In *Proc. of the IEEE International Symposium on Circuits and Systems*, pages 656–659, 2005.

[4] G. C. Cardarilli, A. Nannarelli, and M. Re. Reducing Power Dissipation in FIR Filters using the Residue Number System. In *Proc. of the IEEE 43rd IEEE Midwest Symposium on Circuits and Systems*, pages 320–323, 2000.

[5] M. Dugdale. VLSI Implementation of Residue Adders Based on Binary Adders. *IEEE Trans. Circuits Syst. II*, 39:325–329, 1992.

[6] C. Efstathiou, H. T. Vergos, G. Dimitrakopoulos, and D. Nikolos. Efficient Diminished-1 Modulo $2^n + 1$ Multipliers. *IEEE Trans. Comput.*, 54(4):491–496, 2005.

[7] C. Efstathiou, H. T. Vergos, and D. Nikolos. Fast Parallel-Prefix Modulo $2^n + 1$ Adders. *IEEE Trans. Comput.*, 53(9):1211–1216, 2004.

[8] K. E. Elleithy, M. A. Bayoumi, and K. P. Lee. $\theta(\log n)$ Architectures for rns Arithmetic Decoding. In *Proc. of the 9^{th} IEEE Symposium on Computer Arithmetic*, pages 202–209, 1989.

[9] A. A. Hiasat. High-Speed and Reduced Area Modular Adder Structures for RNS. *IEEE Trans. Comput.*, pages 84–89, 2002.

[10] C. K. Koc and C. Y. Hung. Multi-operand Modulo Addition Using Carry-Save Adders. *Electronics Letters*, 26:361–363, 1990.

[11] P. M. Kogge and H. S. Stone. A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations. *IEEE Trans. Comput.*, C-22:786–792, 1973.

[12] L. M. Leibowitz. A Simplified Binary Arithmetic for the Fermat Number Transform. *IEEE Trans. Acoust., Speech, Signal Processing*, 24:356–359, 1976.

[13] Y. Ma. A Simplified Architecture for Modulo $(2^n + 1)$ Multiplication. *IEEE Trans. Comput.*, 47(3):333–337, 1998.

[14] H. Nozaki et al. Implementation of RSA Algorithm based on RNS Montgomery Multiplication. In *Proc. of the 3^{rd} International Workshop on Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science Vol. 2162, Springer-Verlag*, pages 364–376, 2001.

[15] S. J. Piestrak. Design of Residue Generators and Multioperand Modular Adders using Carry-Save Adders. *IEEE Trans. Comput.*, 43:68–77, 1994.

[16] J. Ramirez et al. Fast RNS FPL-based Communications Receiver Design and Implementation. In *Proc. of the 12^{th} International Conference on Field Programmable Logic, Lecture Notes in Computer Science Vol. 2438, Springer-Verlag*, pages 472–481, 2002.

[17] J. Ramirez et al. RNS-enabled Digital Signal Processor Design. *Electronics Letters*, 38(6):266–268, 2002.

[18] L. Skavantzos. Design of Multi-operand Carry-Save Adders for Arithmetic Modulo $(2^n + 1)$. *Electronics Letters*, pages 1152–1153, 1989.

[19] M. A. Soderstrand et al. *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. IEEE Press, 1986.

[20] A. Tyagi. A Reduced-Area Scheme for Carry-Select Adders. *IEEE Trans. Comput.*, 42(10):1163–1170, 1993.

[21] H. T. Vergos and C. Efstathiou. On the Design of Efficient Modular Adders. *Journal of Circuits, Systems and Computers*, 14(5):965–972, 2005.

[22] H. T. Vergos, C. Efstathiou, and D. Nikolos. Diminished-One Modulo $2^n + 1$ Adder Design. *IEEE Trans. Comput.*, 51:1389–1399, 2002.

[23] R. Zimmerman. Efficient VLSI Implementation of Modulo $(2^n \pm 1)$ Addition and Multiplication. In *Proc. of the 14^{th} IEEE Symposium on Computer Arithmetic*, pages 158–167, April 1999.