International Journal of Electronics, Electrical and Computational System
IJEECS
ISSN 2348-117X
Volume 6, Issue 8
August 2017

Academic Science

Performance Comparison of Weighted Modulo (2^n +1) Adder using Different Prefix Structures

Aruna Kokkula
Electronics and Communication Engineering
Matrusri Engineering College, Saidabad
Hyderabad, India

A.S. Keerthi Nayani
Electronics and Communication Engineering
Matrusri Engineering College, Saidabad
Hyderabad, India

Abstract -- Arithmetic modulo 2^n+1 computation is one of the most common Residue Number System (RNS) operations that are used in Pseudo random number generation and Cryptography. The RNS has been employed for efficient parallel carry-free arithmetic computations, such as addition, subtraction, and multiplication, in DSP applications. As the computations for each residue channel can independently be done without carry propagation. Since RNS based computations can achieve significant speedup over the binary-system-based computation, they are widely used in DSP processors, FIR filters, and communication components. Modulo 2^n+1 adder provides the computation of the arithmetic addition by using a system called RNS. RNS speeds up the arithmetic operations by dividing into smaller parallel operations. This paper proposes improved area efficient weighted modulo 2^n+1adders. This is achieved by modifying existing diminished-1 modulo 2^n +1adders to incorporate simple correction schemes. Our proposed adders can produce modulo sums within the range {0, 2^n}, which is more than the range {0, 2^n −1} produced by existing diminished-1 modulo 2^n +1 adders using different prefix structures. Different prefix structures are used for comparing the adder in Area, Delay, Speed. This paper is implemented in Xilinx 10.1 using Verilog and it is simulated in ISE simulator.

Keywords -- Weighted Modulo Adder, Residue Number System and Arithmetic modulo.

I. INTRODUCTION

Arithmetic modulo 2^n+1 computation is one of the most common Residue Number System (RNS) operations. The RNS has been employed for efficient parallel carry-free arithmetic computations. RNS based computations can achieve significant speedup over the binary-system-based computation. RNS is widely used in DSP processors, FIR filters and communication components. Modulo 2^n+1 adder are also utilized as the last stage adder of modulo 2^n+1 multipliers. The modulo 2^n+1 addition is the most crucial step among the commonly used moduli sets.

The modulo arithmetic: Two numbers a an d b are said to be equal or congruent modulo if N/ (a-b), i.e. if their difference is exactly divisible by N. Usually a, b, are non negative and N is a positive integer. a=b (mod N). The word 'modulo' means 'to the modulus'. For any positive integer n, let S be the complete set of residues {0,1,2,3…n-1}. Then addition modulo n on s is defined as for a and b in S, take the usual sum of a and b as integers, and let r be the element of S to which the result is congruent (modulo n); the sum a=b(mod N) is equal to r. Similarly, multiplication modulo n is defined by taking ab(mod n) to be equal to s, where s is the element of S to which the usual product of a and b is congruent modulo n. The set of numbers congruent to a modulo N is denoted $[a]_N$. If bЄ $[a]_N$ then, by definition, N/(a-b), or, in other words, a and b have the same remainder of division by N. Since there are exactly N possible remainders of division by N, there are exactly N different sets $[a]_N$. Quite often these N sets are simply identified with the corresponding remainders: $[0]_N$=0,$[1]_N$=1,…,$[N-1]_N$=N-1. Remainders are often called *residues* accordingly, [a]'s are also known as the *residue classes*. It's easy to see that if a=b (mod N) and if c=d(mod N) then (a+c)=(b+d) (mod N). The same is true for multiplication. These allow us to introduce an algebraic structure into the set {$[a]_N$: a=0,1…n-1}. A first application field is in Residue Number Systems (RNS). In an RNS based application, every number X is represented by a sequence of residues (X_1, X_2, . . . ,X_M), where X_i =X

International Journal of Electronics, Electrical and Computational System
IJEECS
ISSN 2348-117X
Volume 6, Issue 8
August 2017

mod p_i, where p_i =prime integers. Every RNS operation on two operands is defined as.

$(Z_1; Z_2; \ldots ; Z_M) = (X_1; X_2; \ldots ;X_M) \Diamond (Y_1;Y_2; \ldots ;Y_M)$

For most RNS applications ,◊ denotes addition, subtraction, or multiplication. Significant speedup over the corresponding binary operations can be therefore achieved because each Z_i is computed in parallel in a separate arithmetic unit (channel) since its computation depends only on X_i, Y_i, and p_i. Addition in such systems is performed using three channels that, in fact, are a modulo 2^n-1 (equivalently, one's complement), a modulo 2^n, and a modulo 2^n+1 adder. The addition delay in an RNS application which uses the above modulo is dictated by the modulo 2^n+1 channel. Modulo 2^n+1 adder are also utilized as the last stage adder of modulo 2^n+1 multipliers. Modulo 2^n+1 multipliers find applicability in many applications such as cryptography, fermit numbers etc.

II. LITERATURE SURVEY

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, here after 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 :

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

Following the general residue adder architecture, implementation 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. 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 is computed in parallel. 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, by observing that most carry propagate and generate signals for both cases are common and therefore an augmented single Carry Look-Ahead (CLA) unit is sufficient. Finally, parallel-prefix weighted adders have been presented. These have been shown to be the fastest available and more efficient than the others. 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 also perform weighted modulo 2^n+ 1 addition, by reducing both the area and the delay of the resulting components.

(a) Vergos and Efstathiou

Given two (n+1)-bit numbers A and B, where $0 \leq A,B \leq 2^n$, the values of diminished-1 of A and B are denoted by $A^*=A-1$ and $B^*= B-1$, respectively. The diminished-1 sum S^* can be computed by

$$\mathbf{S^* = |S - 1|_{2^n+1} = |A+B-1|_{2^n+1} = |A^* + B^*|_{2^n} + c_{out}}$$

Where $|X|_Z$ is defined as modulo Z of X, and c_{out} is denoted as the inverted end-around carry of the diminished-1 modulo 2^n sum of n-bit A^* and B^*. Suppose that $A= a_n\ a_{n-1}\ a_{n-2} \ldots .a_1\ a_0$ and $B = b_n\ b_{n-1}\ b_{n-2} \ldots .b_1\ b_0$ denote two operands in the composed of the least significant bits of A and B , respectively. Equivalently, we have that $A=a_n\times 2^n+A_n$ and $B=b_n\times 2^n+B_n$. weighted representation. Let A_n and B_n denote the n-bit vectors composed of the least significant bits of A and B , respectively. Equivalently, we have that $A=a_n\times 2^n+A_n$ and $B=b_n\times 2^n+B_n$.

The previous analysis indicates that the A_n, B_n vectors , and D can be added by the inverted end-around carry save adder stage presented in Fig. 4. We can then drive its outputs Y and U to a diminished-1 adder, that will provide at its output Y+U+1 mod 2^n+1 , that is, it will provide the n least significant bits of the weighted modulo addition of A and B. The most significant bit of the weighted sum should be at 1, only when A+B mod 2^n+1 is equal to 2^n, or equivalently when Y+U+1 mod 2^n+1 equal to 2^n, or equivalently (since) when $Y+U = 2^n-1$, that is, when Y and U are bitwise complementary.

International Journal of Electronics, Electrical and Computational System
IJEECS
ISSN 2348-117X
Volume 6, Issue 8
August 2017

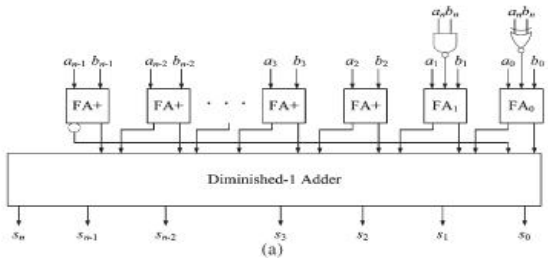


Figure 1: Architecture of the weighed modulo 2^n+1 adder for Vergos and Efstathiou

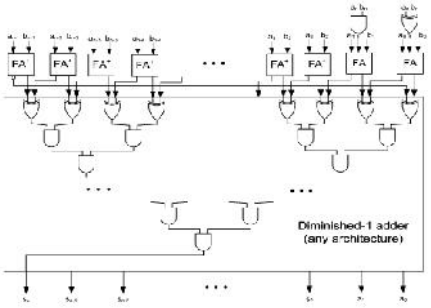


Figure 2: Modulo 2^n+1adder for weighted operands built using a diminished-1 adder.

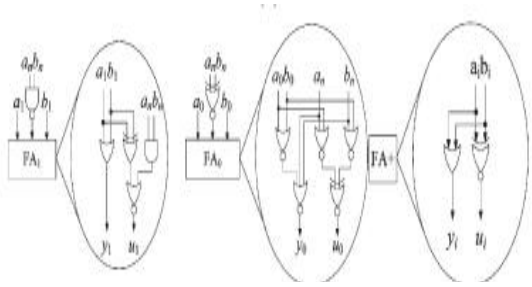


Figure 3: Architecture of FA1, FA0, and FA+,

This condition can be easily detected as the logical AND of the XOR of the bits of Y and U with the same weight. Since in every fast adder architecture there is a preprocessing stage that apart from the generate and propagate terms also computes the half-sum term, that is the XOR of the corresponding input operands bits, the extra hardware required for the most significant bit of the weighted addition is small; just AND ing the half-sum terms together. It should be noted that this operation will not add any delay on the critical path of the adder.The figure presents the implementation that results from the previous analysis for a modulo 2^n+1adder for operands in the weighted representation. It is composed of a diminished-1 adder and an inverted end-around-carry CSA stage. The full adders of the CSA stage perform the $|A_n + B_n + D|_{2^n+1}$ addition. The FAs at bit positions 2 up to (n-1) are denoted as FA s since one of their operands coming from vector is 1. Therefore, they have hardware and delay complexity equal to that of a half adder. Several simplifications can also be performed to the two rightmost FAs along with the accompanying NAND and XNOR gates by considering that $a_n(b_n)$ and a_1 or a_0 (and b_1 or b_0) cannot be simultaneously at 1. Fig. 6 presents examples of simplified circuits at these least significant bit positions. However, even more aggressive simplification is possible within the pre-processing stage of the diminished-1 adder by considering that the propagate and generate signals at bit position 1 depend on signals y_0 and u_1 that are both dependent on a_n and b_n. The most significant carry bit produced by the CSA is inverted and then driven to the least significant position. The sum and carry bits of equal weight produced by the CSA stage are then driven to the diminished- 1 adder. Obviously, any architecture proposed, can be used for the latter. The diminished-1 adder's result forms the n least significant bits of the weighted sum. The indication of complementary input vectors at the diminished-1 adder is the most significant bit of the weighted sum.

(b) Vergos and Bakalis (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*and B* denote two n-bit vectors such A* + B* = A + B − 1. According to (9), we then have that :

$|A + B|_{2^n+1}$ = A + B − $(2^n + 1)$, if A + B ≥ $2^n + 1$

= A + B, otherwise.

or equivalently that :

$|A + B|_{2^n+1}$ = (A + B − 1) − 2^n, if A + B − 1 ≥ 2^n

= (A + B − 1) + 1, otherwise.

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

$(|A + B|_{2^n+1})_{2^n}$ = (A*+ B*), if (A* + B*) ≥ 2^n

= (A*+ B*) + 1, otherwise.

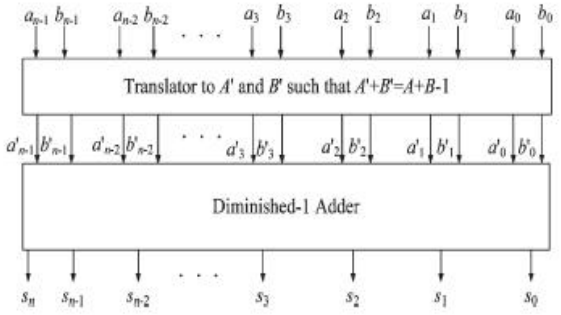
Let c_{out} denote the carry output of the (A* + B*) n-bit integer addition. Using it, we can unify the two cases

as : $|A + B|_{2^n+1}{}_{2^n}$ = $|A^*+ B^*|_{2^n} + c_{out}$

Relation 12 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

International Journal of Electronics, Electrical and Computational System
IJEECS
ISSN 2348-117X
Volume 6, Issue 8
August 2017

adder) provided that the sum of its inputs is decreased by 1, that is, if we use as inputs the A* and B* 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 since $0 \le A, B \le 2n-1$ reduces to $A^*+B^* = 2^n-1$, that is, when A* and B* are bit-wise complementary. This condition can be easily detected as the logical AND of the XOR of the bits of A*and B* with the same weight. Since in every fast adder architecture there is a pre-processing 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 2logn+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 result from the previous analysis. A translator circuit accepts the n-bit vectors A and B and provides the vectors A* and B*. 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 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.



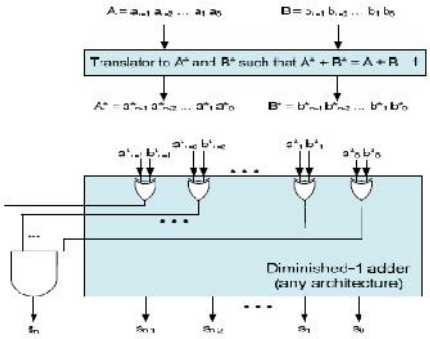


Figure 4: Architectures of the weighed modulo 2^n+1 adder for proposal

The architecture proposed in [1] makes use of a constant time operator, which is composed of a simplified carry-save adder stage, leading to efficient modulo 2^n+1 adder. The architecture proposed in [2] can be applied in the design of area-efficient residue generators and multi operand modulo adders. However, it should be noted that, in [1], the values that are subtracted by the inputs A and B are not constants. In [2], the way to implement the translator for decreasing the sum of two inputs by 1 was not mentioned. Furthermore, in [2], the ranges of two inputs A and B are less than the one proposed in [1] (i.e., $\{0, 2^n - 1\}$ versus $\{0, 2^n\}$). To remedy these problems, we will propose our area-efficient weighted modulo 2^n+1 adder design.

III. AREA-EFFICIENT WEIGHTED MODULO $2^N + 1$ ADDER DESIGN

Instead of subtracting the sum of A and B by D, which is not a constant, we use the constant value $-(2^n + 1)$ to be added by the sum of A and B. In addition, we make the two inputs A and B to be in the range $\{0, 2^n\}$, which is 1 more than $\{0, 2^n -1\}$ as proposed. In the following, we present the designs of our proposed weighted modulo $2^n +1$ adder.

Given two (n+1)-bit inputs $A = a_n a_{n-1}. . . a_0$ and $B = b_n b_{n-1}, . . . , b_0$, where $0 \le A,B \le 2^n$. The weighted modulo 2^n+1 of A+B can be represented as follows:

$$|A+B|_{2^n+1} = A+B-(2^n+1), \quad \text{if } (A+B) > 2^n$$
$$= A+B, \quad \text{otherwise.}$$

Equation (1) can be stated as

$$||A + B|_{2^n+1}|_{2^n} = |A+B-(2^n+1)|_{2^n}, \quad \text{if } (A+B)>2^n$$

International Journal of Electronics, Electrical and Computational System
IJEECS
ISSN 2348-117X
Volume 6, Issue 8
August 2017

Academic Science

= $|A+B-(2^n+1)|_{2^n}+|(2^n+1)|_{2^n}$, otherwise.

This can then be expressed as

$||A + B|_{2^n+1}|_{2^n} = |A+B-(2^n+1)|_{2^n}$, if $(A + B) > 2^n$

= $|A+B-(2^n+1)|_{2^n}+1$, otherwise.

From (3), it can easily be seen that the value of the weighted modulo 2^n+1 addition can be obtained by first subtracting the value of the sum of A and B by (2^n+1) (i.e., 0111, . . . , 1) and then using the diminished-1 adder to get the final modulo sum by making the inverted end-around carry as the carry-in. Now, we present the method of weighted modulo 2^n+1 addition of A and B as follows. Denoting Y' and U' as the carry and sum vectors of the summation of A, B and $-(2^n + 1)$, where $Y'=y'_{n-2}\, y'_{n-3}, \ldots, y'_0\, y'_{n-1}$ and $U'= u'_{n-1}\, u'_{n-2}, \ldots, u'_0$, the modulo addition can be expressed as follows:

$|A+B-(2^n+1)|_{2^n} =|$

$$
\begin{aligned}
&|A+B-(2^n+1)|_{2^n}\\
&=\left|\sum_{i=0}^{n-2}\left(2^i\times(a_i-b_i)\right)+2^{n-1}\right.\\
&\quad\left.\times(2a_n+2b_n+a_{n-1}-b_{n-1})+0\underbrace{11\ldots11}_{n\text{ bits}}\right|_{2^n}\\
&=\left|\sum_{i=0}^{n-2}\left(2^i\times(a_i-b_i+1)\right)+2^{n-1}\right.\\
&\quad\left.\times(2a_n+2b_n+a_{n-1}-b_{n-1}+1)\right|_{2^n}\\
&=\left|\sum_{i=0}^{n-2}\left(2^i\times(2y'_i+u'_i)\right)+2^{n-1}\right.\\
&\quad\left.\times(2a_n+2b_n+a_{n-1}-b_{n-1}+1)\right|_{2^n}.
\end{aligned}
$$

For i=0 to n − 2, the values of y_i' and u_i' can be expressed as $y_i' = a_i \vee b_i$ and $u_i' = a_i \oplus b_i$, respectively ($\vee$ is denoted as logic OR operation). Since the bit widths of Y' and U' are only n bits, the values of y'_{n-1} and u'_{n-1} are required to be computed taking the values of a_n, b_n, a_{n-1}, and b_{n-1} into consideration (i.e., y'_{n-1} and u'_{n-1} are the values of the carry and the sum produced by $2a_n + 2b_n + a_{n-1} + b_{n-1} + 1$, respectively). It should be noted that $0 \le A,B \le 2^n$, which means $a_n = a_{n-1} = 1$ or $b_n = b_{n-1} = 1$ will cause the value of A or B to exceed the range of $\{0, 2^n\}$. Thus, these input combinations (i.e., $a_n = a_{n-1} = 1$ or $b_n = b_{n-1} = 1$) are not allowed and can be viewed as don't care conditions, which can help us simplify the circuits for generating y'_{n-1} and u'_{n-1}. That is, the maximum value of $2a_n + 2b_n + a_{n-1} + b_{n-1} + 1$ is 5, which occurs at $a_n= b_n=1$ (i.e., the maximum value of y'_{n-1} is 2).The truth table for generating y'_{n-1}, u'_{n-1} and FIX is given in Table I, where × is denoted as don't care. The reason for FIX is that, under some conditions, $y'_{n-1} = 2$ (e.g., $a_n=b_n=1$ and $a_{n-1}=b_{n-1}=0$), which cannot be represented by 1-bit line (marked as "∗" in Table I); therefore, the value of y'_{n-1} is set to 1, and the remaining value of carry (i.e., 1) is set to FIX. Notice that FIX is wired-OR with the carry-out of Y'+ U' (i.e., c_{out}) to be the inverted end around carry (denoted by $c_{out} \vee$ FIX) as the carry-in for the diminished-1 addition stage later on. When $y'_{n-1} = 2$, FIX = 1; otherwise, FIX = 0.

TABLE 1. Truth Table For Generating y'_{n-1}, u'_{n-1} AND FIX (∗: CONDITIONS WHEN $y'_{n-1}= 2$)

a_n	b_n	a_{n-1}	b_{n-1}	u'_{n-1}	y'_{n-1}	FIX
0	0	0	0	1	0	0
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	1	1	0
0	1	0	0	1	1	0
0	1	0	1	×	×	×
0	1	1	0	0	1*	1
0	1	1	1	×	×	×
1	0	0	0	1	1	0
1	0	0	1	0	1*	1
1	0	1	0	×	×	×
1	0	1	1	×	×	×
1	1	0	0	1	1*	1
1	1	0	1	×	×	×
1	1	1	0	×	×	×
1	1	1	1	×	×	×

According to Table 1, we can have $y'_{n-1} = (a_n \vee b_n \vee a_{n-1} \vee bn-1)$, $u'_{n-1}=a_{n-1} \oplus b_{n-1}$, and FIX = $a_n\, b_n \vee b_n\, a_{n-1} \vee a_n\, b_{n-1}$, respectively.

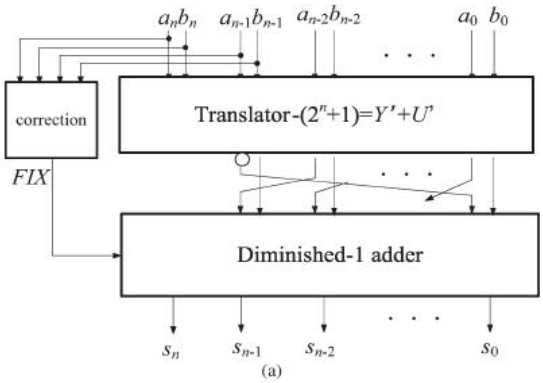


Figure 5: Architecture of proposed weighted modulo 2^n+1 adder with the correction

International Journal of Electronics, Electrical and Computational System
IJEECS
ISSN 2348-117X
Volume 6, Issue 8
August 2017

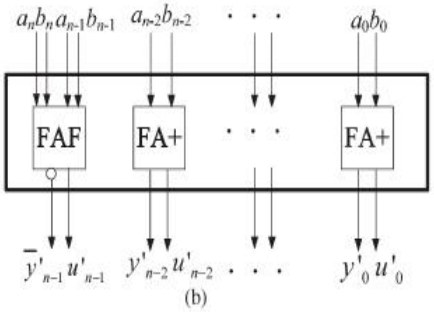


Figure 6: Architecture of the translator − (2^n + 1)

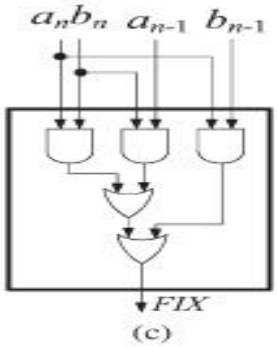


Figure 7: Architecture of the correction scheme

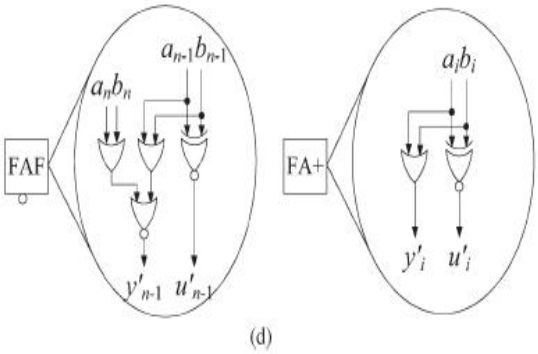


Figure 8: Architecture of FAF and FA+

From Fig. 3, the signal of FIX can be computed in parallel with the translation to Y'+U', leading to efficient correction. In addition, the hardware cost for our correction scheme and FAF are less than the one proposed in [1], due to the fact that there are two inconstant numbers that should be processed [i.e., FA1 and FA0, as shown in Fig. 1(b)] in the translation stage. We used Sklansky-style and Brent–Kung style parallel-prefix structures for the diminished-1 adder implementations. The diminished-1 adder based on the Sklansky-style parallel-prefix structure with correction circuits for our proposed weighted modulo 2^8+1 adder is shown in Fig. 4(a). The square ( ) and diamond (♦) nodes denote the pre- and post processing stages of the operands, respectively. The black nodes (•) evaluate the prefix operator, and the white nodes ( ) pass the unchanged signals to the next prefix level. The four nodes with detailed implementations are given in Fig. 4(b). It should be noted that the value of s_n (i.e., c_{out}) can be computed by wiring-AND all propagate signals (i.e.,$s_n = \prod_{i=0}^{n-1} p_i$).

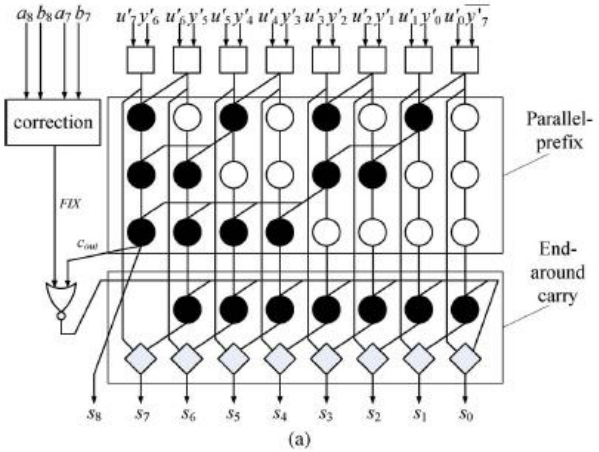


Figure 9: Diminished-1 adder based on the Sklansky-style parallel-prefix Structure

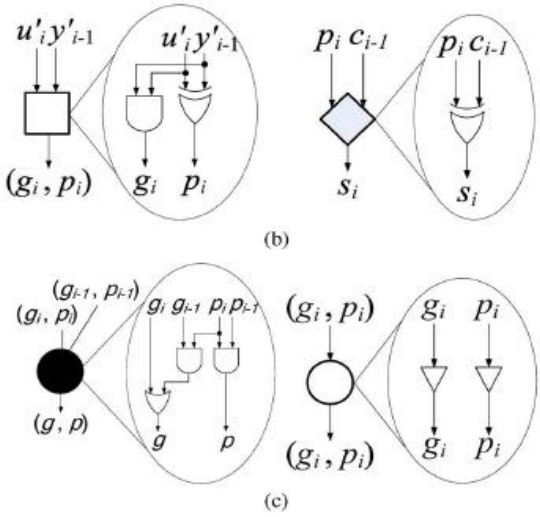


Figure 10: Square and (♦) diamond nodes, respectively

(•) Black and (◦) white nodes, respectively

3.3 Parallel Prefix Structures:

The different prefix structures are provided for the implementation of the modulo addition:

õ **Sklansky prefix structure:** This structure provided the basic prefix operations in the modulo performance. This requires the less number of the

International Journal of Electronics, Electrical and Computational System
IJEECS
ISSN 2348-117X
Volume 6, Issue 8
August 2017

gates when compared to the other structures. These are 4 layered structures.

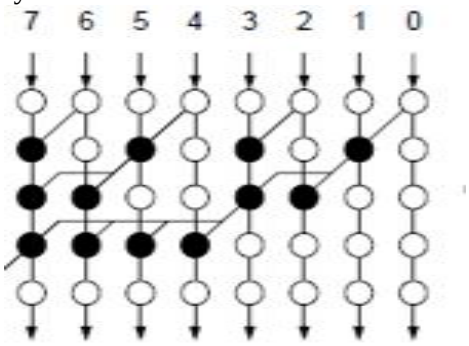


Figure 11: Sklansky prefix structure

Koggestone prefix structure: This structure requires the more number of the gates when compared to the other structures. These are also 4 layered structures.

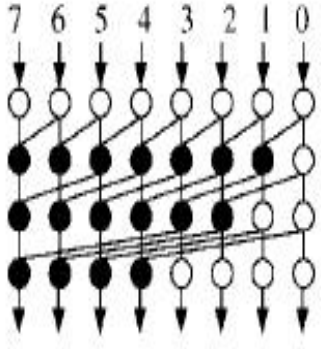


Figure 12: Koggestone structure

Lander-Fisher prefix structure: This structure is similar to the Sklansky prefix structure. This structure requires the more number of the gates when compared to the other structures. These are also 4 layered structures

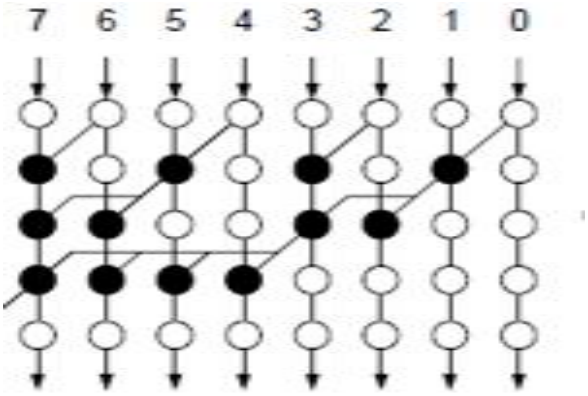


Figure 13: Lander-Fisher structure

Brent-Kung Structure: The Brent-Kung adder is a parallel prefix adder that requires 2(log2N)-1 stages. It was originally proposed as a simple and regular design of a parallel adder that addresses the problems of connecting gates in a way to minimize chip area. Accordingly, it is considered one of the better tree adders for minimizing wiring tracks, fan out, and gate count and is used as a basis for many other networks.

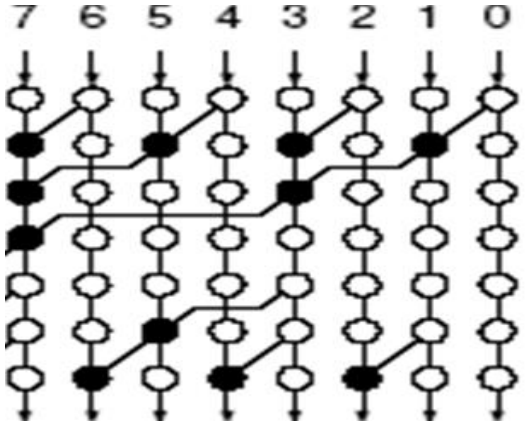


Figure 14: Brent-Kung parallel-prefix structure:

Example 1: Suppose n = 4, A = 16_{10} = 10000_2, and B =15_{10} = 01111_2.

Step 1: (A + B) − (2^n + 1) => Y' = 1110_2, U' = 0000_2, FIX = 1.

Step 2: Y' + U' = 1110_2, c_{out} = 0,

=> Y' + U' + c_{out} ∨ FIX = 1110_2 = $|16 + 15|_{17}$ = 14_{10}.

Example 2: Suppose n = 4, A = 11_{10} = 01011_2, and B = 5_{10} = 00101_2.

Step 1: (A + B) − (2^n + 1) => Y' = 1110_2, U' = 0001_2, FIX = 0.

Step 2: Y' + U' = 1111_2, c_{out} = 0,

=> Y' + U' + c_{out} ∨ FIX = 10000_2 = $|11 + 5|_{17}$ = 16_{10}

IV. RESULTS DISCUSSIONS

This section présents the évaluation paramètres such as area, delay and these parameters can be utilized to select the efficient design for real time applications.

Academic Science

Table 2: Comparison of area (8 -bit)

Parameter	SKLANSKY	KOGGESTONE	BRENTKUNG
No of Slices	15	48	15
No of 4 i/p LUT's	26	85	26
No of I/O's	27	51	27
No of bounded I/O's	27	51	27

Table 3: Comparison of area (16-bit)

Parameter	SKLANSKY	KOGGESTONE	BRENTKUNG
No of Slices	48	119	49
No of 4 i/p LUT's	85	210	85
No of I/O's	51	27	51
No of bounded I/O's	51	27	51

Table 4: Comparison of Delay(8-bit)

Prefix Structures Model	8 BIT	16 BIT
SKLAN SKY	7.132 ns	32.250 ns
KOGGEESTONE	9.347 ns	25.139 ns
BRENTKUNG	18.863ns	30.710

V.CONCLUSION & FUTURE SCOPE

Conclusion: In this brief, an improved area-efficient weighted modulo $2^n + 1$ adder has been proposed. This has been achieved by modifying the existing diminished-1 modulo adders to incorporate simple correction schemes. The proposed adders can perform weighted modulo $2^n + 1$ addition and produce sums that are within the range $\{0, 2^n\}$. Synthesis results show that our proposed adders can outperform previously reported weighted modulo adder in terms of area and delay constraints.

Future scope: The design can be extended to applications where the area and delay are the major parameters. The design can also be implemented in 32 and 64 bits to improve area and delay and it can be used for low power CMOS applications like digital signal processing and nanotechnology.

REFERENCES

[1] . R. Zimmermann, "Efficient VLSI implementation of modulo 2n±1 addition and multiplication," in Proc. 14th IEEE Symp. Comput. Arithmetic, Apr. 1999, pp.158–167.

[2] . H. T. Vergos, C. Efstathiou, and D. Nikolos, "Diminished-one modulo 2n + 1 adder design," IEEE Trans. Comput., vol. 51, no. 12, pp. 1389–1399, Dec. 2002.

[3] . C. Efstathiou, H. T. Vergos, and D. Nikolos, "Modulo 2n ± 1 adder design using select-prefix blocks," IEEE Trans. Comput., vol. 52, no. 11, pp. 1399–1406, Nov. 2003.

[4] . S.-H. Lin and M.-H. Sheu, "VLSI design of diminished-one modulo 2n + 1 adder using circular carry selection," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 55, no. 9, pp. 897–901, Sep. 2008.

[5] . T.-B. Juang, M.-Y. Tsai, and C.-C. Chiu, "Corrections on 'VLSI design of diminished-one modulo 2n + 1 adder using circular carry selection'," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 56, no. 3, pp. 260–261, Mar. 2009.

[6] . H. T. Vergos and C. Efstathiou, "A unifying approach for weighted and diminished-1 modulo 2n + 1 addition," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 55, no. 10, pp. 1041–1045, Oct. 2008.

[7] . H. T. Vergos and D. Bakalis, "On the use of diminished-1 adders for weighted modulo 2n + 1 arithmetic components," in Proc. 11th EUROMICRO Conf. Digit. Syst. Des. Archit., Methods Tools, Sep. 2008, pp. 752–759.