# A Unified Addition Structure for Moduli Set $\{2^n-1, 2^n, 2^n+1\}$ Based on a Novel RNS Representation

Somayeh Timarchi[1,2], Mahmood Fazlali[1,2], and Sorin D.Cotofana[2]
[1]*Department of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran*
[2]*Department of Computer Engineering, TUDelft, Delft, Netherlands*
*s_timarchi@sbu.ac.ir;fazlali@cc.sbu.ac.ir;S.D.Cotofana@tudelft.nl*

***Abstract*—Given that modulo $2^n\pm1$ are the most popular moduli in Residue Number Systems (RNS), a large variety of modulo $2^n\pm1$ adder designs have been proposed based on different number representations. However, in most of the cases, these encodings do not allow the implementation of a unified adder for all the moduli of the form $2^n-1$, $2^n$, and $2^n+1$. In this paper, we address the modular addition issue by introducing a new encoding, namely, the stored-unibit RNS. Moreover, we demonstrate how the proposed representation can be utilized to derive a unified design for the moduli set $\{2^n-1,2^n,2^n+1\}$. Our approach enables a unified design for the moduli set adders, which opens the possibility to design reliable RNS processors with low hardware redundancy. Moreover, the proposed representation can be utilized in conjunction with any fast state of the art binary adder without requiring any extra hardware for end-around-carry addition.***

## I. Introduction

Residue Number Systems (RNS) can represent large numbers with a set of smaller residues according to the assumed moduli set. Subsequently, arithmetic operations, e.g., addition and multiplication, can be performed on each residue independently without any need for carry propagations between them, which leads to the reduction of the carry propagation chain [1]. This facilitates the realization of high-speed and low-power arithmetic units. Therefore, RNS based arithmetic units could be of potential interest for embedded processors, such as those found in mobile devices, for which high speed and low-power consumption are critical. Furthermore, RNS is extremely appropriate for addition and multiplication dominated applications such as digital signal processing [2], Digital filtering [3], communications [4], and cryptography [5], all of which are extremely important in computing today. The main drawback associated to RNS based computation however relates to the overhead introduced by the input and output conversions from binary to RNS and vice versa [6].

Generally speaking, the performance of an RNS processor depends on aspects, like the number and the form of the selected moduli set and the utilized digit encoding. In this paper we focus on the design of area effective modulo adders for the moduli set in form of $\{2^n-1,2^n, 2^n+1\}$ and address this issue from the digit encoding point of view. We note that up to date several representations have been proposed for deriving efficient architectures for modulo $2^n$-1 and $2^n$+1 arithmetic: weighted representation [12], diminished-1 [13], and signed-lsb representation [14]. Weighted representation use ($n$+1)-bit operands in modulo $2^n$+1, thus one bit more than it is required for modulo $2^n$-1. In diminished-1 and signed-lsb number systems, each operand is represented by $n$ weighted positions for modulo $2^n$+1 to remove the problem of using ($n$+1)-bit operands in the weighted representation. However, diminished-1 requires extra circuits for zero detection and correction. Besides, it is appropriate just for modulo $2^n$+1. Therefore, the previously mentioned encodings cannot be applied to all the three moduli of the form $2^n$-1, $2^n$, $2^n$+1 and as a consequence they cannot result in a unified adder design.

In this paper, our goal is to design adders for the moduli set $\{2^n-1,2^n,2^n+1\}$ based on a unified structure able to handle all the moduli in the set. The unified design for the modular adders allows us to synthesize reconfigurable adders that can process inputs for different moduli. First we propose a new number system namely, stored-unibit RNS, that is appropriate for these moduli. The main characteristics of the proposed representation are as follows:

*a)* *It removes the zero detection and correction stages required by the diminished-1 representation, and solves the problem of (n+1)-bit operands specific to the weighted representation.*

*b)* *It provides an appropriate encoding for all the three moduli $2^n$-1, $2^n$+1, and $2^n$, which potentially results in a fault-tolerant circuit.*

*c)* *It employs conventional parallel-prefix carry computation unit (or any fast addition methods) without any extra stage for End-Around-Carry (EAC) of modulo $2^n\pm1$ additions.*

*d)* *It doesn't require any modifications of the parallel-prefix adder, as it is the case for the method in [13]. Therefore, the proposed method is less complex than other state of the art modulo $2^n\pm1$parallel-prefix adders.*

We note that stored-unibit encoding, called Stored-Unibit-Transfer RNS (SUT-RNS), has been suggested as an effective encoding for redundant RNS [10,11,15]. In this paper, we demonstrate that by employing stored-unibit representation in RNS and redundant RNS, we enable fault-tolerant realization of adders for the moduli set $\{2^n-1, 2^n,$

$2^n+1\}$ at the expense of low hardware redundancy.

The rest of this paper is organized as follows. We review modulo $2^n\pm1$ representations and their addition algorithms in Section 2. The proposed representation for modulo addition is introduced in Section 3. Section 4 describes our new modulo $2^n\pm1$ adders. Comparison and discussion on the adders are presented in Section 5. Finally, the paper is summarized in the last section.

## II. Backgrounds on Modulo ($2^N\pm1$) Adders

Modulo $2^n\pm1$ addition is computed by end-around-carry addition for $2^n$-1 and inverted end-around-carry addition for $2^n$+1. However the direct feedback of output carry or inverted output carry to the input carry of an adder is not very attractive as it creates a combinational loop with all the consequences this may have. To remove this problem, some methods have been proposed like:

1) Carry-Look-Ahead (CLA) adder [12].

2) Parallel-prefix unit along with an extra level for EAC addition [12].

3) Modified parallel-prefix without any need for the extra EAC level [8].

The third method (called TPP), depicted in Fig. 1, offers a logic depth of $\log_2 n$ prefix levels. However it requires significantly more cells and interconnects area than the two first methods. Moreover, the TPP method makes use of a customized parallel-prefix adder that cannot be replaced by alternative $n$-bit adders.

In the last years, several representations have been proposed to design modulo $2^n$+1 adders. These representations are based on the following three number systems:

**1) *Weighted number system:***

Normal unsigned encoding is utilized to represent the RNS digits. The standard weighted binary representation uses ($n$+1) bits for modulo $2^n$+1 operands. So, it is not a faithful representation, because there are some numbers that do not represent valid residues.

The most efficient modulo addition circuits for $2^n$+1 have been reported in [7] and [13]. The adders embed a modified parallel-prefix structure without any need for the extra level for EAC as introduced in [8]. The structure proposed in [7] is depicted in Fig. 1.

**2) *Diminished-1 number system:***

In order to accelerate the modulo $2^n$+1 arithmetic operations, the diminished-1 representation has been introduced in [14]. In this 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. The ordinary addition can be performed with an end-around-carry parallel-prefix adder with $c_{in} = \overline{c_{out}}$ [12]. Efficient diminished-1 adder has been described in [8]. The adder follows the parallel-prefix paradigm without any need to the extra level for EAC. A diminished-1 modulo adder with TPP structure is depicted in Fig. 2.

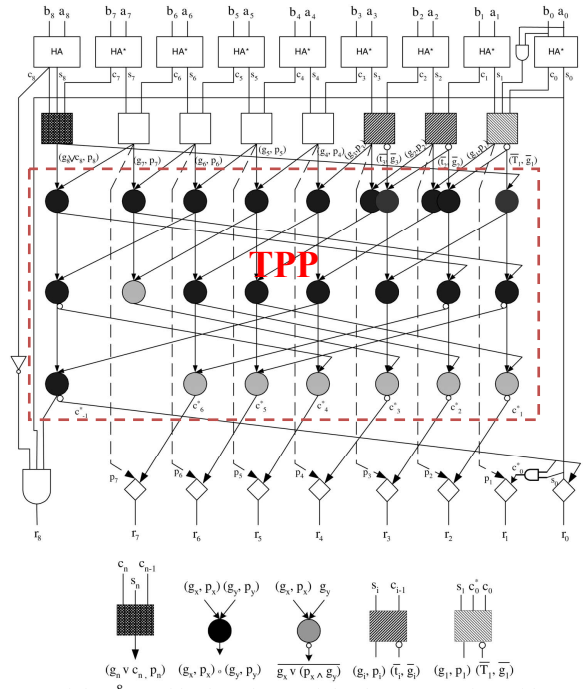



Fig. 1. Modulo $2^8$+1 adder based on weighted representation without carry increment stage (Weighted-TPP) [7].

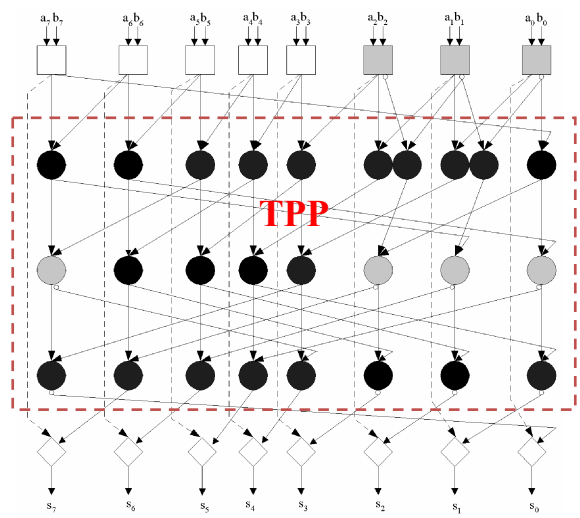



Fig. 2. Modulo $2^8$+1 adder based on diminished-1 representation without carry increment stage (diminished-TPP) [8].

**3) *Signed-lsb number system:***

Signed-lsb representation requires $n$ weighted positions, with two bits in the least significant position (lsp) and $n$-1 bits in the other positions [9]. The lsb contains a normal bit in the range {0,1} (posibit) and a negated bit in the range {-1,0} (negabit), while the other positions contain $n$-1 posibits. The representation of posibit and negabit is presented in Table I. Signed-lsb representation can remove the end-around-carry addition. Therefore the end-around-carry for modulo $2^n$+1 addition is stored in the least-significant bit without any need to propagate it through the higher positions. Moreover, the same addition scheme can be applied for modulo $2^n$-1addition. A TPP adder implementation of signed-lsb representation has been presented in [9].

Table I. Introduction to posibit, negabit and unibit.

| Bit Name | Lower and upper values | Dot notation | Symbolic notation | Lower value representation | Upper value representation | Arithmetic value |
|---|---|---|---|---|---|---|
| Posibit | {0,1} | ● | $x$ | 0 (0) | 1 (1) | $x$ |
| Negabit | {-1,0} | ○ | $X$ | $\underline{0}$ (-1) | $\underline{1}$ (0) | $X$-1 |
| Unibit | {-1,1} | □ | $x'$ | $\underline{\underline{0}}$ (-1) | $\underline{\underline{1}}$ (1) | $2x'$-1 |

We note that modulo $2^n$+1 residues fit in the range [0, $2^n$]. There are two kinds of representations for this range of integer values: faithful and non-faithful representation. The standard weighted binary representation with ($n$+1) bits is not a faithful one as there are some numbers that do not represent valid residues. Both diminished-1 and signed-lsb representations are faithful, because all values in this range have one representation.

Like diminished-1, Signed-lsb removes the problem of ($n$+1)-bit operands of weighted representation for modulo $2^n$+1. However, diminished-1 representation is just designed for modulo $2^n$+1, whereas, signed-lsb representation is also appropriate for modulo $2^n$-1 and $2^n$.

## III. Proposed Stored-Unibit RNS Representation

In modulo $2^n$-1 and $2^n$+1 addition, an output carry $c_{out}$ (of weight $2^n$) can be reentered as $c_{out}$ and $-c_{out}$ in the lsp, respectively. A solution to avoid the addition or subtraction of $c_{out}$ is to simply store it in lsp. To this end we use a specific bit that can store ±1. The bit is called a unibit and it is in the range {-1,1}. Table II describes the new number representation. The black circles indicate normal bits (posibits). The single white circle represents a negated bit (negabit) and the white square a unibit [16].

Table II: The Stored-unibit RNSencoding for modulo $2^n$ +1.

| Range | Bit Representation |
|---|---|
| [$-2^{n-1}$-1, $2^{n-1}$] | ○●. . . ●● □ |

As shown in Table II, the encoding consists of a main part [$-2^{n-1}$, $2^{n-1}-1$] in signed form and a transfer part in the range {−1,1}. The new proposed representation can encode $2^n$+2 numbers in the range [$-2^{n-1}$-1, $2^{n-1}$].

We underline the negabits and draw two lines under the unibits in SUT-RNS bit representation, as indicated in Table I. A negabit with inverted encoding is encoded by using logical $\underline{1}$ to denote the arithmetic value 0 and logical $\underline{0}$ to denote the arithmetic value ↖1. A unibit is also encoded using logical $\underline{\underline{1}}$ to denote the arithmetic value 1 and logical $\underline{\underline{0}}$ to denote the arithmetic value ↖1. The arithmetic value of a negabit and unibit are shown in Table I.

The new proposed representation has the ability of storing the end-around-carry in lsp without propagating it to the next positions. In the next section, we demonstrate that this representation leads to efficient modulo adders.

## IV. New Modulo ($2^N$± 1) Adders

Modulo $2^n$+1 addition for stored-unibit RNS encoding can be performed according to Fig. 3. First, in Step 1, the two input unibits ($a'_0$ and $b'_0$) are accumulated by simple gates as described in Table III.

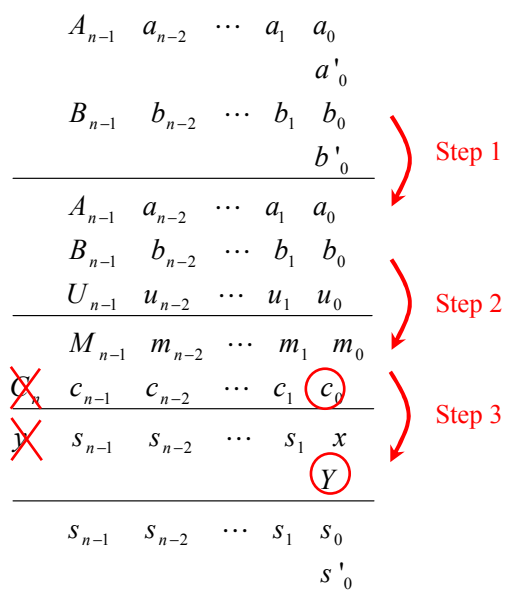



Fig. 3. General method for stored-unibit RNS modulo $2^n$+1 addition.

Table III: Unibit accumulation in two's complement format.

| $a'_0$ $b'_0$ | □ ($a'_0$, $b'_0$) | $U_{n-1}$ | $u_{n-2}$ | … | $u_2$ | $u_1$ | $u_0$ |
|---|---|---|---|---|---|---|---|
| $\underline{\underline{0}}$ $\underline{\underline{0}}$ | -2 | $\underline{0}$ | 1 | …. | 1 | 1 | 0 |
| $\underline{\underline{0}}$ $\underline{\underline{1}}$ | 0 | $\underline{1}$ | 0 | …. | 0 | 0 | 0 |
| $\underline{\underline{1}}$ $\underline{\underline{0}}$ | 0 | $\underline{1}$ | 0 | …. | 0 | 0 | 0 |
| $\underline{\underline{1}}$ $\underline{\underline{1}}$ | 2 | $\underline{1}$ | 0 | …. | 0 | 1 | 0 |

Step 2 can be implemented by an $n$-bit carry-save adder (CSA). The output negabit $C_n$ is stored as the posibit $c_0$, which is equal to the inverted $C_n$. In Step 3 the two $n$-bit vectors are accumulated. The only difference between modulo $2^n$-1 and $2^n$+1 addition is the polarity of the reentering output bits. In modulo $2^n$-1 addition, the polarity of the output bit is preserved.

The last step of addition in Fig. 3 can be implemented by an $n$-bit ripple-carry adder. The proposed modulo $2^n$-1 and $2^n$+1 adders are depicted in Fig. 4 and 5, respectively. The black (white) circles inside the Full-Adder (FA) blocks denote posibits (negabits), and the square indicates a unibit. The output sum and carry are determined according to the input polarities.

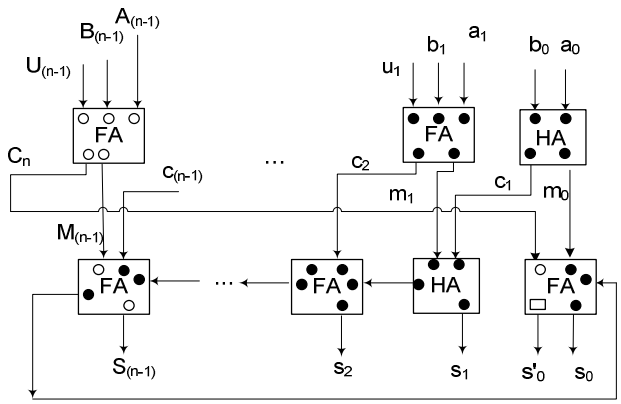



Fig. 4. Proposed stored-unibit RNS modulo $2^n$-1 ripple-carry adder.

The structures depicted in Fig. 4 and 5 are similar, except the existing of two inverters for the end-around-carries in modulo $2^n$+1 adder. As suggested in the figures, the end-around-carries are absorbed in lsp and don't propagate to the higher positions. They produce the output least significant bits, $s_0$ and $s'_0$. Thus the overall latency of the adder is $(n\text{-}2)T_{FA}+2T_{HA}+T_{XOR}$, which equals to the delay of $2n$ unit gates (UG) in unit-gate model [17]; 2UG for FA, 1UG for HA cells, and 2UG for the final XOR gate (FA in the right bottom).

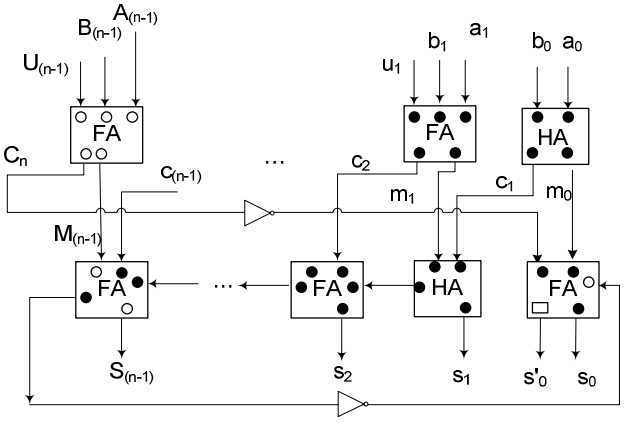



Fig. 5. Proposed stored-unibit RNS modulo $2^n$+1 ripple-carry adder.

The last stage in Fig. 3 can be implemented with a fast adder to reduce the total latency. The proposed parallel-prefix modulo adder and its component structures are depicted in Fig. 6 and Table IV, respectively.

First, the inputs are converted to two vectors by CSA. Subsequently, a binary addition can be performed via a prefix network for the carry calculation. Several tree structures have been proposed in [18], [19], [20], [21] for prefix carry computation. The adder structures have distinct implementation area, speed, and fan-out characteristics. For example, adders with a Ladner-Fischer prefix structure [18] require less implementation area, but have large fan-out when compared to adders with a Kogge-Stone prefix structure [20]. On the other hand, adders with a Kogge-Stone prefix structure are faster [21]. Ladner-Fischer and Kogge-Stone prefix structures, according to [21], are the end cases of minimum implementation area and maximum speed, respectively, of a large family of addition tree structures, which all offer the minimum logical depth property.

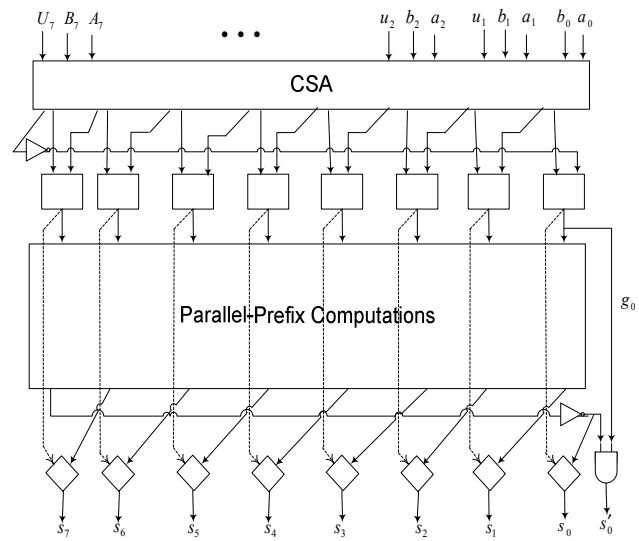
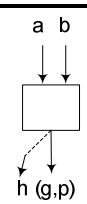
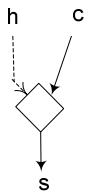



Fig. 6. Proposed stored-unibit RNS modulo $2^n$+1 parallel-prefix adder.

Table IV. Component structures of the proposed design

| Block Diagram | Design Logic |
|---|---|
|   | $h = a \oplus b$<br>$p = a + b$<br>$g = a.b$ |
|   | $s = h \oplus c$ |

As suggested in Fig. 6, the proposed method removes the problems of ($n$+1)-bit operands of the weighted representation and zero detection and correction of diminished-1 representation. Moreover, the stored-unibit technique is not a specific adder like the method proposed in [9] for signed-lsb method, and can employ any parallel-prefix carry computation unit without any modifications.

## V. COMPARISONS

To evaluate the proposed modulo adders, we implemented three previous modulo addition techniques based on the three different number systems. These techniques were compared to the proposed modulo adders in this paper, with respect to the hardware redundancy in RNS. The first technique is based on the method presented in [7], which uses the weighted representation. The second one is the diminished-1 addition technique of [8] and the third one is signed-lsb technique proposed in [9]. These adders are referred to as Weighted, Diminished-1, and Signed-lsb. Our approach is referred to Stored-unibit RNS.

At first, we use the unit-gate model reported in [17]. This model assumes that each gate, except the exclusive-OR gate, counts as one elementary gate for both area and delay, and an exclusive-OR gate counts for two elementary gates.

According to this model, the latencies of the Diminished-1, Weighted, Signed-lsb, and Stored-unibit RNS adders are presented in Table V. The overall delay of stored-unibit RNS is $2\log_2 n + 5$ unit gates: two UG for CSA, one UG for the square operator in Fig. 6, $2\log_2 n$ for the parallel-prefix computations and finally, and 2 UG for the final diamonds.

One can observe in the Table that the stored-unibit RNS has the least delay when conventional parallel-prefix carry computation is utilized.

When the designs are based on the modified parallel-prefix scheme (TPP), the stored-unibit RNS adder outperforms the one in [7] and has the same delay as [8] and [9].

Moreover, as indicated in the last column of the Table, only signed-lsb and stored-unibit representations result in unified designs for the three moduli of $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$, thus provide support for reliable RNS processors. The unified design leads to the possibility of providing reliability with low hardware redundancy by using reconfigurable adders that can process inputs for different moduli. As described in [9], such a reconfigurable modular adder enables fault-tolerant designs with much lower hardware redundancy than the full replication. One way to do this is to implement four such adders for the three moduli set and then configure them to perform the three different modular additions $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$, with one of them kept as a spare. If a fault occurred in one of the available adders, the spare can be configured accordingly and employed instead of the faulty adder.

To evaluate the speed, area and power dissipation of the considered architectures we implemented them in CMOS technology. The structural VHDL descriptions of the modulo $2^8$+1 adders have been first generated. After verifying the correctness of each description, we synthesized them for 130nm CMOS technology with the Synopsys Design Vision tool. A typical corner (1.2V, 25°C) was considered. The results of total power, delay, and area for each adder are included in Table VI. Delay, area, and power results are given in ns, $\mu m^2$, and mW, respectively.

The results indicate that area, delay and power of Stored-Unibit RNS and Signed-LSB are comparable with Weighted and Diminished-1. However they both provide reliability support. We note that the stored-unibit approach offers several advantages as it make use of the same prefix computation unit as in the non-modulo adder, without requiring any circuits for treating zero operands or carry increment stage as indicated in Fig. 6. Moreover, we can apply any fast adder to the proposed design. Therefore, the proposed adder has a simpler implementation than other existing modulo $2^n$+1 adders and requires a simple design and modification procedure.

Table V. Delay of modulo $2^n$+1 parallel-prefix adders in UG.

| Modulo Addition Method | Conventional Parallel-Prefix | Modified Parallel-Prefix (TPP) | Reliability |
|---|---|---|---|
| Weighted [7] | $2\log_2 n + 9$ | $2\log_2 n + 6$ | No |
| Diminished-1 [8] | $2\log_2 n + 7$ | $2\log_2 n + 5$ | No |
| Signed-LSB [9] | $2\log_2 n + 7$ | $2\log_2 n + 5$ | Yes |
| Stored-Unibit RNS (NEW) | $2\log_2 n + 5$ | $2\log_2 n + 5$ | Yes |

Table VI: Synthesized Comparison Results for modulo $2^8$+1 parallel-prefix adders.

| Modulo Addition Method | Area ($\mu m^2$) | Average Power (mW) | Delay (ns) |
|---|---|---|---|
| Weighted [7] | 2320 | 2.67 | 0.57 |
| Diminished-1 [8] | 2371 | 2.42 | 0.51 |
| Signed-LSB [9] | 2034 | 2.29 | 0.57 |
| Stored-Unibit RNS (NEW) | 2287 | 2.54 | 0.61 |

## VI. CONCLUSIONS

In this paper we proposed a new redundant number representation namely, the Stored-Unibit RNS, which can be efficiently utilized for the moduli set $\{2^n\text{-}1,\ 2^n,\ 2^n\text{+}1\}$. It presents an encoding that is suitable for all modulo $2^n$-1, $2^n$+1 and $2^n$ which enables the construction of a unified design for the three moduli adders. In this way one can construct fault-tolerant RNS processors at the expense of low hardware redundancy.

Our proposal makes use of an *n*-bit binary adder in combination with a small amount of additional logic. Another advantage of the new representation is that it simply employs conventional parallel-prefix carry computation unit (or any fast addition methods) without any extra stage for end-around-carry of modulo $2^n\pm1$ additions. Moreover, the proposed method is less complex than other state of the art modulo $2^n\pm1$ parallel-prefix adders.

## REFERENCES


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

[2] Kouretas, I.; Paliouras, V., "High-radix residue arithmetic bases for low-power DSP systems," 16th International Conference on Digital Signal Processing, 5-7 Jul. 2009, pp.1- 6.

[3] Conway R. and Nelson, J., "Improved RNS FIR filter architectures", IEEE Trans. Circuits and Systems-II: Express Briefs, vol. 51, no. 1, pp. 26-28, Jan. 2004.

[4] Madhukumar, A.S. and Chin, F., "Enhanced architecture for residue number system-based CDMA for high-rate data transmission," IEEE Trans. Wireless Communications, vol. 3, no. 5, pp. 1363-1368, Sep. 2004.

[5] Bajard, J., and Imbert, L., "A full RNS implementation of RSA," IEEE Trans. on Computers, vol. 53, no. 6, pp. 769-774, Jun. 2004.

[6] Sabbagh, A., Navi,K., Dadkhah, Ch., Kavehei, O., and Timarchi, S., "Efficient reverse converter designs for the new 4-moduli sets $\{2^n\text{–}1,\ 2^n,\ 2^n\text{+}1,\ 2^{2n+1}\text{–}1\}$ and $\{2^n\text{–}1,\ 2^n\text{+}1,\ 2^{2n},\ 2^{2n}\text{+}1\}$ based on new CRTs", IEEE Trans. Circuit and Systems I, vol.57, no.4, Apr. 2010.

[7] Efstathiou, C., Vergos H.T. and Nikolos D., "Fast parallel-prefix $2^n$+1 adder", IEEE Trans. Computers, vol. 53, no. 9, Sep. 2004

[8] Vergos, H.T., et al., "Diminished-one modulo 2n+1 adder design," IEEETrans. Computers, vol. 51, pp. 1389-1399, 2002.

[9] Jaberipur, G., Parhami, B., "Unified Approach to the Design of Modulo-($2^n\pm1$) Adders Based on Signed-LSB Representation of Residues", 19th IEEE International Symposium on Computer Arithmetic, 2009.

[10] Timarchi S., and Navi, K., "Efficient class of redundant residue number system", IEEE International Symposium on Intelligent Signal Processing (WISP), Madrid, Spain, pp. 475-480, 3-5 October 2007.

[11] Timarchi, S., and Navi, K., "Arithmetic circuits of redundant SUT-RNS", IEEE Trans. Instrumentation and Measurement, vol.58, no.9, Sep. 2009, pp.2959-2968.

[12] Zimmermann, R., "Efficient VLSI implementation of modulo ($2^n\pm1$) addition and multiplication," Proc. of the 14th IEEE Symposium on Computer Arithmetic (ARITH-14), pp. 158-167, Apr. 1999.

[13] Patel, R.A., Benaissa, M., Boussakta, S., Powell, N., "Power-delay-area efficient modulo 2n+1 adder architecture for RNS," IEEE Electronic Letter, vol. 41, Issue 5, pp. 231–232, 3 Mar. 2005.

[14] Leibowitz L.M., "A simplified binary arithmetic for the fermat numbertransform," IEEE Trans. Acoustics, Speech, Signal Processing, vol. 24, pp. 356-359, 1976.

[15] Timarchi, S., Fazlali, M., 'An Efficient Power-Area-Delay Modulo $2^n$-1 Multiplier', to appear in the 15th CSI international Symposium on Computer Architecture and Digital Systems (CADS), 2010.

[16] Jaberipur, G., Parhami, B., and Ghodsi, M., "Weighted two-valued digit-set encodings: unifying efficient hardware representation schemes for redundant number systems," IEEE Trans. Circuits and Systems I, vol. 52, no. 7, pp. 1348-1357, Jul. 2005.

[17] Tyagi, A.; "A reduced-area scheme for carry-select adders", IEEE Trans. Computers, 42 , pp. 1163 – 1170, 1993.

[18] R.E. Ladner and M.J. Fischer, "Parallel Prefix Computation," J. ACM, vol. 27, no. 4, pp. 831-838, Oct. 1980.

[19] R.P. Brent and H.T. Kung, "A Regular Layout for Parallel Adders," IEEE Trans. Computers, vol. 31, no. 3, pp. 260-264, Mar. 1982.

[20] P.M. Kogge and H.S. Stone, "A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations," IEEE Trans. Computers, vol. 22, no. 8, pp. 783-791, Aug. 1973.

[21] S. Knowles, "A Family of Adders," Proc. 15th IEEE Symp. Computer Arithmetic, pp. 277-281, Apr. 2001.