Efficient VLSI Implementation of 2^n Scaling of Signed Integer in RNS $\{2^n-1, 2^n, 2^n+1\}$

Thian Fatt Tay, Chip-Hong Chang, and Jeremy Yung Shern Low

***Abstract*— Scaling is a problematic operation in residue number system (RNS) but a necessary evil in implementing many digital signal processing algorithms for which RNS is particularly good. Existing signed integer RNS scalers entail a dedicated sign detection circuit, which is as complex as the magnitude scaling operation preceding it. In order to correct the incorrectly scaled negative integer in residue form, substantial hardware overheads have been incurred to detect the range of the residues upon magnitude scaling. In this brief, a fast and area efficient 2^n signed integer RNS scaler for the moduli set $\{2^n-1, 2^n, 2^n+1\}$ is proposed. A complex sign detection circuit has been obviated and replaced by simple logic manipulation of some bit-level information of intermediate magnitude scaling results. Compared with the latest signed integer RNS scalers of comparable dynamic ranges, the proposed architecture achieves at least 21.6% of area saving, 28.8% of speedup, and 32.5% of total power reduction for n ranging from 5 to 8.**

***Index Terms*— Chinese Remainder Theorem, residue number system, scaling, signed integer.**

I. Introduction

Residue number system (RNS), with its inherited modularity, parallelism, and localized carry propagation arithmetic operations, has emerged as a promising substitute for the conventional two's complement system for the data representation and computation of specific applications [1]–[4]. The moduli set selected for an RNS has an influence on its implementation efficiency. Of the known moduli sets, $\{2^n-1, 2^n, 2^n+1\}$ has been most extensively studied. Due to its abundant number of well-developed residue arithmetic operations and reverse converter architectures, many applications have been built around this moduli set [1], [4], [5].

One problem inherited from the nonpositional representation of RNS is the truncation of a number in residue domain, which makes overflow avoidance unwieldy. To overcome the magnitude scaling problem, an adaptive channel equalization filter with a large number of multiply-accumulations was recently implemented with an allied RNS-binary system to achieve significant power reduction over the two's complement system without compromising throughput [2]. Scaling was performed in binary via residue-to-binary conversion to avoid the overflow of dynamic range. In the latest RNS implementation of a 32-b low-pass finite impulse response filter [3], signed number was represented in sign-magnitude form with its magnitude converted to RNS. This atypical RNS representation of signed integer suffers from the dual zero representations and the need to remap the magnitude for regular addition and multiplication, which undermine the advantages of RNS. RNS scaler is usually designed based on either the Chinese Remainder Theorem (CRT) or the mixed-radix conversion (MRC) [6]–[10]. Almost all of the existing RNS scalers focus only on magnitude scaling and have either downplayed or glossed over the problem of sign scaling. Because sign detection itself is a difficult operation in RNS, the challenge of implementing a signed over an unsigned integer RNS scaler is the overheads required for sign detection in order to correctly map the scaled signed residues to the legitimate range. The state-of-the-art signed integer RNS scaler [9] comprises an unsigned integer RNS scaler and a correction circuit. The correction circuit involves a dedicated RNS sign detection circuit which is slow and consumes large logic area. In this brief, a unified architecture is proposed for the signed integer RNS scaling. The targeted moduli set and scaling factor are the popular $\{2^n-1, 2^n, 2^n+1\}$ and 2^n, respectively. Instead of using a dedicated RNS sign detection circuit, the residue representation of the signed integer scaling result is obtained by manipulating the intermediate computation results. Owing to its simplicity, the proposed architecture is faster, smaller, and consumes lower power than the latest architectures for the same scaling factor and the same moduli set [9], and a different moduli set of comparable dynamic range [10].

Manuscript received April 13, 2012; revised July 27, 2012; accepted September 13, 2012.

The authors are with the School of Electrical and Electronic Engineering, Nanyang Technological University, 639798 Singapore (e-mail: tayt0015@e.ntu.edu.sg; echchang@ntu.edu.sg; jere0017@e.ntu.edu.sg).

Digital Object Identifier 10.1109/TVLSI.2012.2221752

II. Unsigned and Signed Integers in RNS

An RNS is defined by a set of N pairwise relatively prime integers $\{m_1, m_2, \ldots, m_N\}$, where m_i is called a modulus. An unsigned integer X within the range of $[0, M-1]$ can be uniquely represented by an N-tuple $(x_1, x_2, \ldots, x_N)$, where the dynamic range $M = \prod_{i=1}^{N} m_i$. The residue x_i is the least positive remainder of the division of X by m_i, and is usually represented as $X \bmod m_i$ or $|X|_{m_i}$.

Let $\tilde{X}$ be a signed integer in the range $[-M/2, M/2-1]$ if M is even, or in the range $[-(M-1)/2, (M-1)/2]$ if M is odd [6]. $\tilde{X}$ can also be uniquely represented by an N-tuple $(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_N)$ in signed RNS representation. The relationship between the residue representations of X in unsigned RNS and $\tilde{X}$ in signed RNS under the same moduli set is

$$\tilde{X} = X \quad \text{if} \quad \tilde{X} \geq 0 \quad \text{and} \quad X - M \quad \text{if} \quad \tilde{X} < 0. \tag{1}$$

When $\tilde{X} \geq 0$, the residue representation of X can be mapped to that of $\tilde{X}$ in the range $[0, M/2-1]$ if M is even, or in the range $[0, (M-1)/2]$ if M is odd. When $\tilde{X} < 0$, the residue representation of X can be mapped to that of $\tilde{X}$ in the range $[M/2, M-1]$ if M is even, or in the range $[(M+1)/2, M-1]$ if M is odd.

III. Proposed Signed Integer RNS Scaler Circuit

A. Magnitude Scaling Error in Signed RNS

Let $Y \equiv (y_1, y_2, y_3)$ be the residues obtained by scaling $X \equiv (x_1, x_2, x_3)$ by k in the residue domain of RNS $\{2^n-1, 2^n, 2^n+1\}$. For $m_1 = 2^n-1$, $m_2 = 2^n$, and $m_3 = 2^n+1$, $|M_1^{-1}|_{m_1} = 2^{n-1}$, $|M_2^{-1}|_{m_2} = 2^n - 1$, and $|M_3^{-1}|_{m_3} = 2^{n-1}+1$, respectively [11], where $M_i = M/m_i$, and $|M_i^{-1}|_{m_i}$ is the multiplicative inverse of $|M_i|_{m_i}$. According to CRT [1], [4], the integer Y can be recovered by

$$\begin{aligned} Y &= \left| \sum_{i=1}^{N} M_i \left| \left| M_i^{-1} \right|_{m_i} y_i \right|_{m_i} \right|_M \\ &= \left| 2^n \left(2^n+1\right) 2^{n-1} y_1 + \left(2^n-1\right)\left(2^n+1\right)\left(2^n-1\right) y_2 \right. \\ &\quad \left. + \left(2^n-1\right) 2^n \left(2^{n-1}+1\right) y_3 \right|_{(2^{2n}-1)2^n}. \end{aligned} \tag{2}$$

However, if X is treated as a signed integer $\tilde{X}$ and is represented in signed RNS, (2) may not yield the correct result of scaling X by k. This is because, the scaling of magnitude in unsigned RNS creates an ambiguity in the range partitioning of signed RNS, causing the scaled residues to be incorrectly mapped for negative integers.

The range mismatch can be resolved by detecting the sign of the input operand. Unfortunately, the sign of an integer is not distinguishable from its residue representation. Existing solutions to the RNS sign detection problem are mainly based on the CRT, MRC, or core function [5], [12], [13]. ROM matrices are required

to find the orthogonal projections in these approaches, making the overall circuit in conjunction with the magnitude scaling module to be more expensive than a residue-to-binary converter. To significantly lower the cost of signed integer scaling circuit, the sign detection and correction circuit must be kept simple and coherent with the magnitude scaling module.

B. Correction Circuit for Signed RNS Scaling

From the definition of scaling in RNS, the following expressions of the scaled residues (y_1, y_2, y_3) in terms of (x_1, x_2, x_3) of an unsigned integer X can be derived [4], [7]:

$$y_1 = |x_1 - x_2|_{m_1} \tag{3}$$

$$y_2 = \left|\left|\left(2^{2n-1} + 2^{n-1}\right)x_1 - 2^n x_2 + \left(2^{2n-1} + 2^{n-1} - 1\right)x_3\right|_{m_1 m_3}\right|_{m_2} \tag{4}$$

$$y_3 = \left|x_2 + 2^n x_3\right|_{m_3} \tag{5}$$

where $k = 2^n$, and $\lfloor . \rfloor$ denotes the least integer function.

In order to obtain the correct residue representation of $\tilde{Y}$, y_1, y_2, and y_3 need to be corrected by detecting the sign of $\tilde{X}$ to map X to $\tilde{X}$ based on their relationship expressed in (1).

No correction is required when $\tilde{X} \geq 0$ because $\tilde{X} = X$ according to (1). Hence, $\tilde{y}_1 = y_1$, $\tilde{y}_2 = y_2$, and $\tilde{y}_3 = y_3$.

However, when $\tilde{X} < 0$, $\tilde{X} = X - M$. Since $k = m_2$,

$$\tilde{Y} = \left\lfloor \tilde{X}/k \right\rfloor = \lfloor (X - M)/k \rfloor = \lfloor (X/k) - m_1 m_3 \rfloor . \tag{6}$$

Since $m_1 m_3$ is an integer, (6) can be rewritten as

$$\tilde{Y} = \left\lfloor X/k \right\rfloor - m_1 m_3 = Y - m_1 m_3. \tag{7}$$

The residue representation of $\tilde{Y}$ when $\tilde{X} < 0$ can be computed as follows [9]:

$$\tilde{y}_1 = |Y - m_1 m_3|_{m_1} = \left||Y|_{m_1} - |m_1 m_3|_{m_1}\right|_{m_1} = |y_1 - 0|_{m_1} = y_1 \tag{8}$$

$$\tilde{y}_2 = |Y - m_1 m_3|_{m_2} = \left||Y|_{m_2} - |m_1 m_3|_{m_2}\right|_{m_2} = |y_2 + 1|_{m_2} \tag{9}$$

$$\tilde{y}_3 = |Y - m_1 m_3|_{m_3} = \left||Y|_{m_3} - |m_1 m_3|_{m_3}\right|_{m_3} = |y_3 - 0|_{m_3} = y_3. \tag{10}$$

From (8) and (10), we see that no correction is needed for $\tilde{y}_1$ and $\tilde{y}_3$ when $\tilde{X} < 0$, but y_2 needs to be incremented by one to obtain $\tilde{y}_2$ as indicated in (9). To detect $\tilde{X} < 0$ from its residues, a full sign detection circuit is required. This is what we would like to avoid here.

Since the dynamic range $M = 2^{3n} - 2^n$ of the moduli set $\{2^n - 1, 2^n, 2^n - 1\}$ is always even, the residue representation of X can be mapped to that of $\tilde{X}$ for X in the range of $[0, M/2 - 1]$ if $\tilde{X} \geq 0$ and in the range of $[M/2, M - 1]$ if $\tilde{X} < 0$. In order to correct Y in unsigned RNS to $\tilde{Y}$ in signed RNS, it is necessary to know when becomes negative. The ranges of the scaled unsigned integer Y for positive and negative $\tilde{X}$ are determined as follows.

When $\tilde{X} \geq 0$, $0 \leq X \leq M/2 - 1$, $0 \leq Y \leq \lfloor (M - 2)/2k \rfloor$, and

$$0 \leq Y \leq 2^{2n-1} - 1. \tag{11}$$

When $\tilde{X} < 0$, $M/2 \leq X \leq M - 1$, $\lfloor M/2k \rfloor \leq Y \leq (\lfloor M - 1)/k \rfloor$, and

$$2^{2n-1} - 1 \leq Y \leq 2^{2n} - 2. \tag{12}$$

From (11) and (12), we understand that when $Y > 2^{2n-1} - 1$, $\tilde{Y}$ is negative and when $Y < 2^{2n-1} - 1$, $\tilde{Y}$ is positive. Thus, the condition when $\tilde{Y}$ is negative can be detected by the $(2n-1)$th bit of Y, that is, $(Y)_{2n-1}$ being "1". According to (9), this bit can be added at the LSB of y_2 to correct the sign error of the magnitude-scaled residue, that is, $\tilde{y}_2 = |y_2 + (Y)_{2n-1}|_{2^n}$. By adopting [7] for magnitude scaling, no additional circuit is needed to determine $(Y)_{2n-1}$, because Y is available in the result of (4) before taking the modulo of m_2.

However, when $Y = 2^{2n-1} - 1$, $\tilde{Y}$ can be either positive or negative according to (11) and (12). Based on the definition of magnitude scaling, $Y = \lfloor X/2^n \rfloor$, the range of X for $Y = 2^{2n-1} - 1$ is given by

$$2^n\left(2^{2n-1} - 1\right) \leq X \leq 2^n\left(2^{2n-1} - 1\right) + 2^n - 1$$
$$2^{3n-1} - 2^n \leq X \leq 2^{3n-1} - 1. \tag{13}$$

From (13), we find that there are altogether 2^n integers in $[2^{3n-1} - 2^n, 2^{3n-1} - 1]$. From (1), we know that when $X \geq M/2 = 2^{3n-1} - 2^{n-1}$, $\tilde{X} < 0$. Therefore, for $\tilde{Y} = 2^{2n-1} - 1$, when $\tilde{X} \geq 0$ and $\tilde{Y} \geq 0$

$$2^{3n-1} - 2^n \leq X \leq 2^{3n-1} - 2^{n-1} - 1 \tag{14}$$

and when $\tilde{X} < 0$ and $\tilde{Y} < 0$

$$2^{3n-1} - 2^{n-1} \leq X \leq 2^{3n-1} - 1. \tag{15}$$

Since the magnitude of X is not directly available at the input, it can only be determined from the residue representation. By taking the modulo 2^n operation on (15), the range of the residue x_2 is found to be $2^{n-1} \leq x_2 \leq 2^n - 1$. Since $\tilde{x}_2 \equiv x_2$, we have $2^{n-1} \leq \tilde{x}_2 \leq 2^n - 1$. Let $(x_i)_j$ denote the jth bit of x_i. It is observed that the bit $(\tilde{x}_2)_{n-1}$ is always "1" for $\tilde{x}_2$ in the range of (15) and "0" otherwise. Hence, when $Y = 2^{2n-1} - 1$, $(\tilde{x}_2)_{n-1}$ alone is sufficient to determine the sign of $\tilde{Y}$, that is, $(\tilde{x}_2)_{n-1}$ is "1" when $\tilde{Y} < 0$, and "0" when $\tilde{Y} \geq 0$.

In the above discussion, we have proven that the sign of $\tilde{Y}$ can be simply detected by using the bit $(Y)_{2n-1}$ except for the case when $Y = 2^{2n-1} - 1$ where we need an extra bit $(\tilde{x}_2)_{n-1}$ to determine the sign of $\tilde{Y}$. The condition of $Y = 2^{2n-1} - 1$ can be determined by checking if $2n - 1$ LSBs of Y are 1s and the MSB of Y is "0". However, this implementation results in a very high fan-in multilevel logic structure, which is very slow and costly when n is large. We will show that the fan-in of the multilevel logic gate circuit for the detection of the condition $Y = 2^{2n-1} - 1$ can be reduced from $2n$ b to only $n + 2$ b, namely the n-bit residue y_1, and two other bits $(Y)_{2n-1}$ and $(y_2)_{n-1}$.

When $Y = 2^{2n-1} - 1$, $(Y)_{2n-1}$ = "0" and

$$y_1 = \left|2^{2n-1} - 1\right|_{2^n-1} = \left|(2^n)\left(2^{n-1}\right) - 1\right|_{2^n-1} = 2^{n-1} - 1. \tag{16}$$

Using (16), y_1 can be computed as $2^{n-1} - 1$ when $Y = 2^{2n-1} - 1$. However, y_1 alone is not sufficient to determine if $Y = 2^{2n-1} - 1$, because there are $2^n + 1$ different integers of Y that have $y_1 = 2^{n-1} - 1$ in their residue representation. Among these integers, $2^{n-1} + 1$ of them have values less than or equal to $2^{2n-1} - 1$. When $y_1 = 2^{n-1} - 1$ and $Y \leq 2^{2n-1} - 1$, Y have values of $2^{n-1} - 1$, $2^{n-1} + 2^n - 2, \ldots, 2^{2n-1} - 2^n$, $2^{2n-1} - 1$. Thus, $y_2 = 2^{n-1} - 1, 2^{n-1} - 2, \ldots, 0, 2^n - 1$. Among these y_2 values, only when $Y = 2^{2n-1} - 1$ can y_2 have value greater than $2^{n-1} - 1$ (i.e., $(y_2)_{n-1}$ = "1"). In other words, $(Y)_{2n-1}$, y_1, and $(y_2)_{n-1}$ are sufficient to determine if $Y = 2^{2n-1} - 1$. Fig. 1 depicts the proposed architecture of the signed integer RNS scaler. The architecture of unsigned integer RNS scaler [7] is enclosed in the dashed-line box, and the correction circuit is enclosed in the solid-line box. The role of the control signal generation under correction circuit is to detect the condition when $Y = 2^{2n-1} - 1$ and $\tilde{Y}$ is in negative range. Under this condition, "1" will be selected as the LSB of the correction factor. Otherwise, $(Y)_{2n-1}$ will be selected. The control signal generation circuit can be built using $n+2$ two-input AND gates with y_1, $(\tilde{x}_2)_{n-1}$, $(Y)_{2n-1}$, and $(y_2)_{n-1}$ as the inputs.

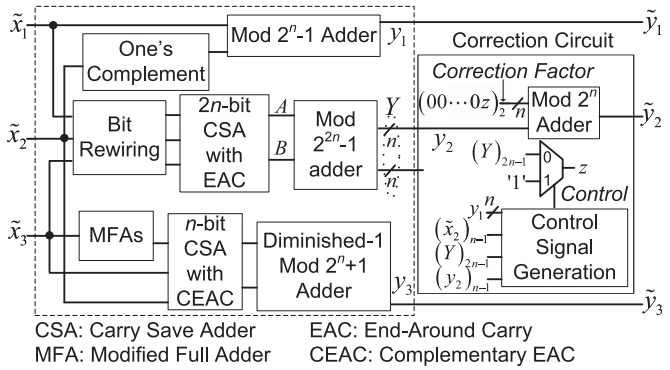

Fig. 1. Proposed two-stage signed integer RNS scaler.

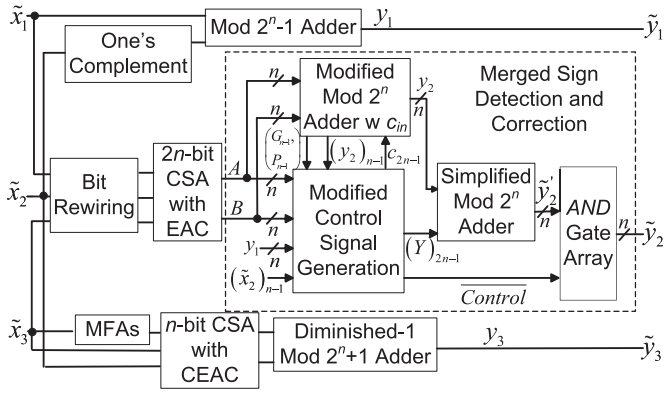

Fig. 2. Proposed simplified signed integer RNS scaler.

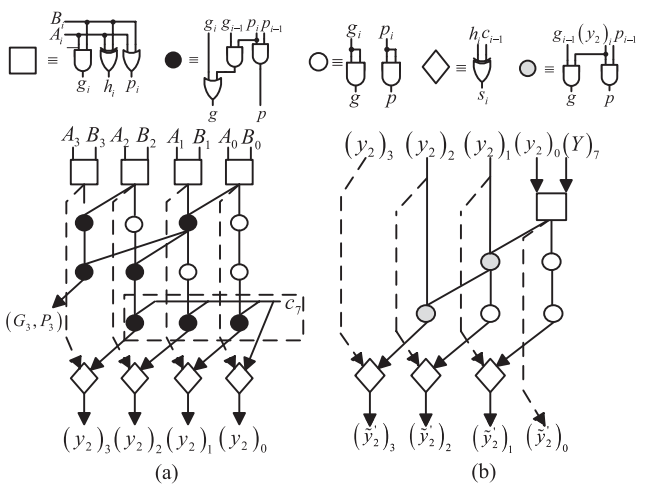

Fig. 3. (a) Modified mod 2^n adder with c_{in} for $n = 4$. (b) Simplified Mod 2^n adder for $n = 4$.

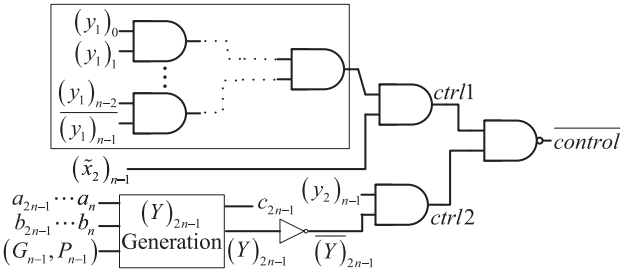

Fig. 4. Modified control signal generation module.

C. Unified Signed Integer RNS Scaler Architecture

The architecture depicted in Fig. 1 is a direct implementation based on the mathematical manipulation discussed earlier. It has modest speed and area complexity. The area and speed can be further improved by merging the two stages of computations into one by eliminating some redundancies in the arithmetic circuits. The single-stage architecture is shown in Fig. 2. The modulo 2^{2n-1} adder of the magnitude scaler and the correction circuit of Fig. 1 are integrated into a "merged sign detection and correction" module. It comprises a modified mod 2^n adder with c_{in}, a simplified mod 2^n adder, a modified control signal generation block, and an AND gate array as shown in Fig. 2.

According to [7]

$$\big||A+B|_{2^{2n}-1}\big|_{2^n} = \begin{cases} ||A+B|_{2^n}+1|_{2^n}, & \text{if } A+B \geq 2^{2n} \\ |A+B|_{2^n}, & \text{otherwise.} \end{cases} \tag{17}$$

From (17), $||A+B|_{2^{2n}-1}|_{2^n}$ can be implemented by a simple mod 2^n adder provided that when $A+B \geq 2^{2n}$, the sum is incremented by one. For simplicity, this simplified implementation of (17) is succinctly denoted by

$$\big||A+B|_{2^{2n}-1}\big|_{2^n} = |A+B+c_{\text{in}}|_{2^n} \tag{18}$$

where $c_{\text{in}} \in \{0,1\}$ is the carry input to the modified mod 2^n adder with c_{in} shown in Fig. 2.

Fig. 3(a) depicts an example of the proposed mod 2^n adder with c_{in} for $n = 4$. The computation of y_2 in (4) can be performed by using the proposed modified mod 2^n adder with c_{in}. This adder adds two n-bit operands taken from the n LSBs of the sum and carry vectors A and B, produced by the $2n$-bit carry-save adder (CSA) with end-around carry (EAC) block shown in Fig. 2. This adder has a similar structure as a standard mod 2^n adder with an additional prefix level. This extra level of prefix operators is enclosed in the dashed-line box of Fig. 3(a). They are used to generate the carry signals due to c_{in} [14]. c_{in} in this case is equal to c_{2n-1}, which is the carry output generated from the $A+B$ operation. By using this adder, we can avoid the use of a large and slow mod 2^{2n-1} adder.

Besides being one of the inputs to the modified control signal generation circuit, $(Y)_{2n-1}$ is also the LSB of the correction factor to map y_2 to $\tilde{y}_2$. The mapping is done by using a simplified mod 2^n adder which has a structure shown in Fig. 3(b), where $(Y)_{2n-1}$ is added to y_2 to obtain $\tilde{y'}_2$, that is, $\tilde{y'}_2 = |y_2 + Y_{2n-1}|_{2^n}$. Since all bits except the LSB of the correction factor are "0", the prefix operator, denoted by "●", is simplified to "○" to reduce the critical path delay.

Fig. 4 shows the circuit of the modified control signal generation block which is similar to that in Fig. 1 with the addition of a $(Y)_{2n-1}$ generation block. The role of the $(Y)_{2n-1}$ generation block is to compute $(Y)_{2n-1}$ and bit c_{2n-1} based on the n MSBs of A and B, and G_{n-1} and P_{n-1} from the modified mod 2^n adder with c_{in}. The c_{2n-1} bit is fed as the carry input c_{in} to the modified mod 2^n adder with c_{in} shown in Fig. 3(a).

In Fig. 1, a multiplexer is needed to select the correction factor to be added to y_2. As $(y_2)_{n-1}$ is an input to the control signal generation block, addition of the correction factor to y_2 will be delayed. In Fig. 2, $(Y)_{2n-1}$is added to y_2 before the corrected output is selected by an AND gate array. This efficient implementation is derived as follows.

When $Y = 2^{2n-1}-1$ and $\tilde{Y}$ is in the negative range

$$\tilde{y}_2 = |y_2+1|_{2^n} = \big|2^n-1+1\big|_{2^n} = \big|2^n\big|_{2^n} = 0. \tag{19}$$

TABLE I
UNIT-GATE AREA AND DELAY OF THE PROPOSED DESIGN

n	Merged Sign Detection and Correction							Others		Overall	
	A_{MMA}	A_{MCS}	A_{SMA}	A_{AND}	D_{MCS}	D_{SMA}	D_{AND}	A	D	A	D
5	57	45	21	5	13	6	1	282	5	410	25
6	72	51	25	6	13	7	1	336	5	490	26
7	87	57	32	7	13	8	1	390	5	573	27
8	105	66	39	8	13	8	1	444	5	662	27

TABLE II
COMPARISON OF NUMBER OF TRANSISTORS AND UNIT-GATE DELAY. VALUE IN PARENTHESES IS THE PERCENTAGE REDUCTION

DR (Bit)	Number of Transistors			Unit-Gate Delay		
	This Brief	[9]	[10]	This Brief	[9]	[10]
15	1640	1972 (16.84)	39 822 (95.88)	25	42 (40.48)	56 (55.36)
18	1960	2360 (16.95)	164 881 (98.81)	26	42 (38.10)	57 (54.39)
21	2292	2748 (16.59)	694 345 (99.67)	27	42 (35.71)	61 (55.74)
24	2648	3136 (15.56)	2 366 381 (99.88)	27	42 (35.71)	62 (56.45)

For other conditions, we just need to add $(Y)_{2n-1}$ to y_2 as follows:

$$\tilde{y}_2 = \left| y_2 + (Y)_{2n-1} \right|_{2^n}. \tag{20}$$

The corrected output derived from (19) and (20) is either 0 or $|y_2 + (Y)_{2n-1}|_{2^n}$. Instead of using n-way multiplexer, n two-input AND gates will suffice to obtain the correct output.

IV. EVALUATION AND COMPARISON

In this section, our design is compared against the latest signed integer RNS scaler architectures proposed in [9] and [10] for four different dynamic ranges of 15, 18, 21, and 24 b. The three-moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ is used in our proposed design and in [9]. For a fair comparison, the moduli sets {23, 29, 31}, {53, 59, 61}, {109, 113, 127}, and {239, 241, 251} with comparable dynamic ranges (DRs) of 15, 18, 21, and 24 b and the same scaling factor are used for [10] as all its moduli must be odd integers. These moduli sets are made up of adjacent prime numbers and are chosen in favor of [10] such that their exact dynamic ranges just fall below those of $\{2^n - 1, 2^n, 2^n + 1\}$ for $n = 5$, 6, 7, and 8, respectively.

Unit-gate model [15] is adopted for the hardware area and delay estimation, where a two-input monotonic gate is assumed to have one unit of area and one unit of delay, an XOR gate has two units of area and two units of delay, and an inverter has zero unit of area and delay.

Channels m_1 and m_3 of our proposed design have similar architectures as those proposed in [7] and their unit-gate areas and delays are excerpted from Tables III and IV of [7], respectively. Channel m_2 consists of bit rewiring, $2n$-bit CSA with EAC, and merged sign detection and correction blocks. The bit rewiring and $2n$-bit CSA with EAC blocks are built up of n OR gates and $2n$ full adders (FAs). The unit-gate areas for $n = 5, 6, 7$, and 8 are listed in Table I, where A_{MMA}, A_{MCS}, A_{SMA}, and A_{AND} denote the unit-gate areas of the modified mod 2^n adder with c_{in}, modified control signal generation block, simplified mod 2^n adder and AND gate array, respectively. The unit-gate areas of all other modules are added together under A in the "Others" column of Table I. All component areas are summed to obtain the overall unit-gate area of the proposed design. Channel m_2 has the longest path delay among the three parallel channels. The longest path of the merged sign detection and correction module goes from the modified control signal generation block to the AND gate array through the simplified mod 2^n adder. Their unit-gate delays are also listed in Table I, where D_{MCS}, D_{SMA} and D_{AND} denote the unit-gate delays of the modified control signal generation block, simplified mod 2^n adder, and AND gate array, respectively. The unit-gate delays of the bit rewiring and $2n$-bit CSA with EAC blocks are consolidated into D in the "Others" column of Table I.

The total unit-gate area and unit-gate delay of the architecture of [9] are estimated to be $13.5n\lceil \log_2 n \rceil + 56.5n + 8$ and $8\lceil \log_2 n \rceil + 18$ units, respectively, based on the same unit-gate analysis as in [15]. In [10], the arithmetic operations are implemented by lookup tables, which are usually implemented with ROM modules. Based on the ROM model of [8], the number of transistors and unit-gate delay of the architecture of [10] are estimated and are shown in Table II. The unit-gate areas of the proposed design and [9] are also converted to transistor counts and are compared in Table II, where one unit-gate area is equivalent to four transistors. This conversion assumes that an FA is implemented with 28 transistors in static CMOS logic style. The results show that the proposed architecture consumes at least 15% and 95% lesser transistors than [9] and [10], respectively, and is at least 35% and 54% faster than [9] and [10], respectively, for DR = 15, 18, 21, and 24 b.

Each design is also specified at gate level using Verilog HDL, synthesized, and technology mapped to TSMC 0.18-μm CMOS technology standard cell library using Synopsys Design Compiler. The designs are independently optimized for speed to obtain their minimum achievable delays. The areas and delays after logic synthesis and optimization are shown in Table III. The results show that our proposed design is at least 21% and 89% smaller, and at least 28% and 63% faster than those of [9] and [10], respectively, for DR = 15, 18, 21, and 24 b. The comparison with [9] suggests that more aggressive area savings have been obtained from the actual implementation of our design than the theoretical estimation based on transistor count. The delay improvement estimated by the unit-gate analysis is somewhat optimistic for the lower dynamic range, but is fairly accurate otherwise. Comparing with [10], the speed improvement obtained by the unit-gate delay model has been underestimated.

The power consumptions of all circuits are measured using Synopsys PrimeTime PX at the same clock rate and the same supply voltage of 1.62 V. For each dynamic range, a common clock period is set based on the slowest design. Monte Carlo simulation method [16] is used with randomly generated inputs to obtain the average power dissipation with 99% confidence that the error is bounded below

TABLE III
COMPARISON OF SYNTHESIZED AREAS, DELAYS, AND TOTAL POWER (VALUE IN PARENTHESES IS THE PERCENTAGE REDUCTION)

DR (Bit)	Synthesized Area (μm^2)			Synthesized Delay (ns)			Total Power (μW)		
	This Brief	[9]	[10]	This Brief	[9]	[10]	This Brief	[9]	[10]
15	9094	11 599 (21.59)	87 152 (89.57)	2.00	2.81 (28.83)	5.42 (63.10)	1602	2374 (32.52)	15 338 (89.56)
18	9164	17 643 (48.06)	278 00 (96.70)	1.94	3.02 (35.76)	6.80 (71.47)	1468	3010 (51.23)	24 888 (94.10)
21	10 096	20 710 (51.25)	648 937 (98.44)	2.09	3.26 (35.89)	8.30 (74.82)	1297	3252 (60.12)	33 364 (96.11)
24	13 725	21 652 (36.61)	2 143 865 (99.36)	2.00	3.34 (40.12)	9.10 (78.02)	1538	3173 (51.53)	68 862 (97.77)

TABLE IV
COMPARISON OF POSTLAYOUT SIMULATION RESULTS

	This Brief	[9]
Core area (μm^2)	21 050	33 298
Delay (ns)	2.00	3.34
Leakage power (μW)	0.623	1.054
Total power (mW)	0.524	1.186
Final core utilization ratio	0.636	0.685

2.5%. The total power dissipations including both dynamic and leakage powers are listed in Table III. Despite operating at a higher data rate, our design saves more than 32% and 89% of power over [9] and [10], respectively. Due to the replacement of mod 2^{2n-1} adder with merged sign detection and correction, our augmented signed RNS scaler is even 22.7% smaller and consumes 23.1% less power on average than the simplest unsigned RNS scaler [7] synthesized under the same technology library. The ineluctable delay overhead due to the sign correction has been reduced to slightly below 30% on average.

Based on the synthesis results, the most competitive contender [9] and our design for the moduli set {255, 256, 257} were physically placed and routed using Cadence SoC Encounter with four metal layers and the same initial core utilization ratios. The postlayout netlists were back-annotated for the same timing and power simulations at 1.62 V and 3.34-ns clock period. Both designs were able to attain their respective minimum delay in Table III after the placement and routing. The physical synthesis results are summarized in Table IV. The final core utilization ratios of our design and [9] are 0.636 and 0.685, respectively. Our proposed design is 36.78% smaller and consumes 55.82% lower power, which are slightly better than the prelayout results due to our design's lower interconnect complexity.

V. CONCLUSION

In this brief, a low complexity high-speed and low-power 2^n signed integer RNS scaler for moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ was proposed. By simplifying and merging the expensive sign detection and correction circuits, the complexity of implementing the signed integer RNS scaler was reduced substantially. We benchmarked our proposed design against two latest signed integer RNS scalers using the unit-gate analysis and logic synthesized results. The logic synthesized results showed that our proposed design was on average 39.38% smaller, 35.15% faster and 48.60% more power-efficient than the most competitive contender over four different dynamic ranges. This superiority was further corroborated by the physical synthesis results based on the same TSMC 0.18-μm standard cell implementation for $n = 8$, where our proposed design ran 40% faster, consumed 56% lower power, and occupied 37% lesser silicon area.

REFERENCES

[1] A. R. Omondi and B. Premkumar, *Residue Number Systems: Theory and Implementation*. London, U.K.: Imperial College Press, 2007.

[2] G. L. Bernocchi, G. C. Cardarilli, A. D. Re, A. Nannarelli, and M. Re, "Low power adaptive filter based on RNS components," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2007, pp. 3211–3214.

[3] P. Maj and G. S. Rath, "A novel design approach for low pass finite impulse response filter based on residue number system," in *Proc. 3rd Int. Conf. Electron. Comput. Technol.*, Apr. 2011, pp. 74–78.

[4] P. V. A. Mohan, *Residue Number Systems: Algorithms and Architectures*. Boston, MA: Kluwer Acad. Publishers, 2002.

[5] T. Tomczak, "Fast sign detection for RNS ($2^n - 1$, 2^n, $2^n + 1$)," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 6, pp. 1502–1511, Jul. 2008.

[6] C. C. Su and H. Y. Lo, "An algorithm for scaling and single residue error correction in residue number systems," *IEEE Trans. Comput.*, vol. 39, no. 8, pp. 1053–1064, Aug. 1990.

[7] C. H. Chang and J. Y. S. Low, "Simple, fast, and exact RNS scaler for the three-moduli set 2^n-1, 2^n, 2^n+1," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 11, pp. 2686–2697, Nov. 2011.

[8] Z. D. Ulman and M. Czyzak, "Highly parallel, fast scaling of numbers in nonredundant residue arithmetic," *IEEE Trans. Signal Process.*, vol. 46, no. 2, pp. 487–496, Feb. 1998.

[9] Y. Ye, S. Ma, and J. Hu, "An efficient 2^n RNS scaler for moduli set $\{2^n-1, 2^n, 2^n+1\}$," in *Proc. Int. Symp. Inform. Sci. Eng. Conf.*, Dec. 2008, pp. 511–515.

[10] S. Ma, J. Hu, Y. Ye, L. Zhang, and X. Ling, "A 2^n scaling scheme for signed RNS integers and its VLSI implementation," *Sci. China, Ser. F, Inform. Sci.*, vol. 53, no. 1, pp. 203–212, Jan. 2010.

[11] R. Conway and J. Nelson, "Fast converter for 3 moduli RNS using new property of CRT," *IEEE Trans. Comput.*, vol. 48, no. 8, pp. 852–860, Aug. 1999.

[12] H. Henkelmann and W. Anheier, "Implementation of sign detection in RNS using mixed radix representation," in *Proc. IEEE Int. Conf. Electron., Circuits Syst.*, Sep. 1999, pp. 323–326.

[13] M. Abtahi and P. Siy, "The non-linear characteristic of core function of RNS numbers and its effect on RNS to binary conversion and sign detection algorithms," in *Proc. Annu. Meeting North Amer. Fuzzy Inform. Process. Soc. Conf.*, Jun. 2005, pp. 731–736.

[14] R. Zimmermann, "Efficient VLSI implementation of modulo ($2^n \pm 1$) addition and multiplication," in *Proc. 14th IEEE Symp. Comput. Conf.*, Apr. 1999, pp. 158–167.

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

[16] R. Burch, F. N. Najim, P. Yang, and T. N. Trick, "A Monte Carlo approach for power estimation," *IEEE Trans. Very Large Scale Int. (VLSI) Syst.*, vol. 1, no. 1, pp. 63–71, Mar. 1993.