INTERNATIONAL JOURNAL OF MERGING TECHNOLOGY AND ADVANCED RESEARCH IN COMPUTING

IJMTARC – VOLUME – VI – ISSUE – 24, OCT-DEC, 2018

ISSN: 2320-1363

DESIGN AND IMPLEMENTATION OF HIGH SPEED AND SMALL AREA ADDER BASED SIGN DETECTOR FOR RNS

K.DIVAKAR REDDY[1], T.BALA OBULA REDDY[2]

[1]PG Student, Dept of ECE(VLSI-SD), SRIT, Proddatur, AP, India.

[2]Assistant Professor, Dept of ECE, SRIT, Proddatur, AP, India.

***Abstract*— The moduli set $\{2^n\ 1, 2^n, 2^n\ \ 1\}$ has been widely used in residue number system (RNS)-based computations. Its sign extraction problem, albeit fundamentally important in magnitude comparison and other difficult algorithms in RNS, has received considerably less attention than its scaling and reverse conversion problems. This brief presents a new algorithm for the design of a fast adder-based sign detector. The circuit is greatly simplified by shrinking the dynamic range to eliminate large modulo operations with the help of the new Chinese remainder theorem. Our synthesis results with the XILINX ISE, show that the proposed design outperforms all the existing adder-based sign detectors reported for this moduli set in area and speed for *n* ranges from 5 to 25 in the step of 5.**

***Index Terms*— Chinese remainder theorem (CRT), computer arithmetic, residue number system (RNS), RNS scaling, sign detection.**

I. INTRODUCTION

In addition, sign recognition within an RNS isn't as efficient as modular procedures, for example addition, subtraction, and multiplication, due to its complexity. A higher-efficiency sign recognition unit for that moduli set is presented. The sign recognition unit is concurrent and appropriate for VLSI implementation in line with the suggested sign recognition formula. Sign recognition plays an important role in branching procedures, magnitude evaluations, and overflow recognition. Since the sign details are hidden in every residue digit inside a residue number system (RNS), sign recognition within an RNS is much more difficult than that within the weighted number system, where the sign is easily the most significant bit (MSB) [1]. The sign recognition problem continues to be investigated by many people scientists. An over-all theorem comes by creating the required conditions for sign recognition. The sign recognition for any selected type of RNS is transported out like a sum modulo 2 of numbers within the connected mixed radix system (MRS). Inside sign recognition technique according to fractional representation is suggested to lessen the sum modulo M within the conversion formula to some sum modulo 2. Inside sign recognition formula in line with the new Chinese remainder theorem (CRT) II is presented. The modulo procedures within the sign recognition formula are bounded by size vM. Inside sign recognition formula uses the nth mixed radix digit in mixed-radix conversion (MRC) to identify the sign function. Up to now, may be the only brief to make use of the combinational logic to apply an indication recognition formula according to. However, the technique can't be extended with other moduli sets. The moduli set, only including the kinds of 2n and 2n-1, continues to be researched extensively recently due to its efficiency for modulo procedures and reverse conversion. Within this brief, an indication recognition formula is presented for that moduli set . The dynamic power is as stated by the synopsis power compiler, which models the switching activity using static probability and toggle rate. The experimental results indicate the suggested sign recognition unit offers significant savings in delay, area and power in comparison using the sign recognition models. First, an indication recognition formula is presented for that restricted moduli set including modulo 2n within the RNS. The suggested sign recognition formula requires only adding the modulo2n. Then, a brand new sign recognition unit is produced for the moduli set in line with the suggested sign recognition formula. The Kodak play touch camcorder only includes a carry save adder (CSA), a comparator, along with a carry generation unit. The suggested formula may be the first suggested for that moduli set [2]. The accomplished efficiency is preferable to those of other techniques, for example calculations according to ROM technology and specialized calculations based. This brief is organized the following. The suggested sign recognition formula for that restricted moduli set. Is definitely the sign recognition unit for that moduli set . The particulars the implementation from the suggested unit with evaluations of area, time, and power after which is definitely the comparison results.

In this paper, Section 2 presents a literature review of the published work related to sign detection in RNS. Section 3 basic preliminaries and notations. Section 4 presents the proposed sign detection algorithm, while Section 5 presents the proposed hardware implementation of the algorithm. Section 6 performs hardware synthesis and modeling for the proposed work and for the most competitive published structures and compares the results.

II. LITERATURE REVIEW

The general idea behind sign detection in RNS is based mainly on reverse decoding of residue digits into their binary equivalent. The binary value is then checked if it belongs to the positive or negative range. However, such an approach is very demanding [2], [3], [4]. Nevertheless, there have been considerable efforts to reduce the time and hardware requirements of sign detection operation [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. Vu [5] presented a sign detector based on fractional representation where each residue digit is applied to a ROM. The outputs of the ROMs, expressed as fractional values, are applied to multi-operand adders. The sign is the most significant bit of the sum. Alia and Martinelli [6] presented a sign detection structure that uses a base extension and requires two multi-operand modulo adders and two modulo multipliers. Tomczak [7] presented a sign detector for the moduli set. The structure of this sign detector consists of a multi-operand adder, two carrygeneration circuits and a post-processing circuit. Wang et al. [8] presented a residue to binary converter for the moduli set which can be easily customized to be a sign detector. It requires a 2n bit carry-save adder network and a 2n bit carry-generation circuit. Xu et al. [9] introduced an algorithm for sign detection for the moduli set.

III. PRELIMINARIES AND NOTATIONS

RNS is characterized by a set of *N* co prime numbers, known as the moduli set {*m*1, *m*2, . . . ,*mN* }, i.e., GCD*(mi ,m j)* = 1∀*i* _= *j* . Any integer *X* can be represented by an *N*-tuple (*x*1, *x*2, . . . , *xN)* in this moduli set. Each residue *xi* is the least nonnegative remainder computed by dividing *X* by the modulus *mi* , which can be expressed mathematically as *xi* = |*X*|*mi* for *i* = 1, 2, . . . , *N*. The product of all moduli is called the dynamic range *M*, i.e., *M* = _*N i*=1 *mi*. Any integer *X* that lies within 0 ≤ *X* < *M* will have a unique residue representation.

An integer *X* within the dynamic range can be recovered from its residue representation (*x*1, *x*2, . . . , *xN)* by applying the CRT [5]

$$X=|\sum_{i=1}^{N} M_i \,||M_i^{-1}|_{m_i} x_i|_{m_i}|_M \qquad (1)$$

where = *M/mi* and |*M*−1*i* |*mi* is the multiplicative inverse of |*Mi* |*mi* . ˆ

To represent a signed integer *X* in RNS, *M* is divided into two symmetrical half ranges for the representation of positive and negative integers. When *M* is even, the range of signed integers that can be unambiguously represented in RNS is [−*M/*2*, M/*2 − 1]. Similarly, for odd *M*, the range of unambiguously represent able signed integers in RNS is [−*(M* − 1*)/*2*, (M* − 1*)/*2]. The signed integer ˆ*X* can be represented using the same residue representation as an unsigned integer *X* for the same moduli set. The relationship between ˆ*X* and *X* is given as follows:

$$\hat{X}=|[X+\frac{M}{2}]|_M\text{-}[\frac{M}{2}] \qquad (2)$$

When ˆ*X* ≥ 0, the residue representation of *X* can be mapped to that of ˆ*X* in the range of [0, *M/*2 − 1] if *M* is even and [0, *(M* − 1*)/*2] if *M* is odd. In a similar way, when ˆ*X* < 0, the residue representation of *X* can be mapped to that of ˆ*X* in the range of [*M/*2*, M* − 1] if *M* is even and [*(M* + 1*)/*2*, M* − 1] if *M* is odd [22]. Thus, the sign of ˆ*X* can be detected as follows.

When *M* is even

$$\text{Sign}[\hat{X}]=\begin{cases} 0,\ if\ X\epsilon[0,\left(\frac{M}{2}\right)-1] \\ 1, if\ X\epsilon[\frac{M}{2}, M-1] \end{cases} \qquad (3)$$

When *M* is odd

$$\text{Sign}[\hat{X}]=\begin{cases} 0,\ if\ X\epsilon[0,\frac{(M-1)}{2}] \\ 1, if\ X\epsilon[\frac{(M+1)}{2}, M-1] \end{cases} \qquad (4)$$

Properties 1 and 2 [5] are employed in order to simplify some arithmetic operations in the derivation of our proposed sign detection circuit for RNS {2*n* − 1, 2*n*, 2*n* + 1}.

Property 1: The modulo 2*n* − 1 multiplication of an *n*-bit binary number *x* and *r* exponent of two is equivalent to a circular left shift (CLS) of the binary bits of *x* by *r* position.

$$|2rx|_{2^n-1}=\text{CL}\boldsymbol{s_n}(\text{x,r}) \qquad (5)$$

where CLS *n(x*, *r)* represents the circular shift of an *n*-bit binary number *x* by *r* bits to the left.

Property 2: As a corollary of Property

$$|-2rx|_{2^n-1}=|2r(2n-1-x)|_{2^n-1}=|2r\overline{x}|_{2^n-1}=CLs_n(\bar{x},\text{r}) \ (6)$$

where *x* is the one's complement of integer *x*

IV. PROPOSED SIGN DETECTION ALGORITHM FOR RNS {2*n* − 1, 2*n*, 2*n* +1}

ISSN: 2320-1363

Let (*x*1, *x*2, *x*3*)* be the residue representation of an integer *X* with respect to the moduli set {*m*1, *m*2, *m*3} = {2*n* − 1*,* 2*n*, 2*n* + 1}. Since the dynamic range *M* of this moduli set can be factored into 2*n* and 22*n* − 1, the sizes of the modulo operations required for detecting the sign of ˆ*X* from its equivalent residue representation of *X* can be substantially reduced by scaling (*x*1, *x*2, *x*3*)* in the residue domain by 22*n* − 1. This will map the lower half range [0*,* 23*n*−1 − 2*n*−1*)* of *X* to the lower half range [0*,* 2*n*−1*)* of the scaled integer *Y* and the upper half range [23*n*−1 − 2*n*−1*,* 23*n* − 2*n)* of *X* to the upper half range [2*n*−1*,* 2*n)* of *Y* , as shown in Fig1.

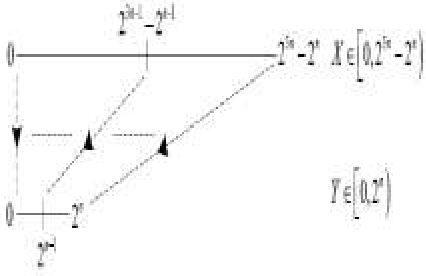


Fig.1. Mapping of the half ranges of integer *X* in [0, *M)* to the half range of its scaled integer *Y* in [0, *M_*).

By shrinking the dynamic range from *M* = 23*n* − 22*n* to *M_* = 2*n*, its half range can be easily detected from the MSB of the scaled integer *Y* .This new concept of sign detection in {2*n* −1, 2*n*, 2*n* +1} can be made very efficient provided that scaling by 22*n* − 1 as well as the reverse conversion of the scaled residues into *Y* can be computed efficiently from the residues *x*1, *x*2, and *x*3. As only the MSB of *Y* is needed for the sign detection of ˆ*X* , a full reverse conversion from (*x*1, *x*2, *x*3*)* is not required.

To simplify the scaling by 22*n* −1 in the residue domain, the new CRT [23], also known as CRT-I, is used to convert *X* into a weighted sum of its residues modulo 22*n* − 1. According to CRT-I

$$X=x_3+m_3|k_1(x_1-x_{3)+K_2m_1(x_2-x_1)|m_1m_2} \qquad (7)$$

where *k*1*m*3 = |1|*m*1*m*2 and *k*2*m*3*m*1 = |1|*m*2 . With *m*1 = 2*n*−1, *m*2 = 2*n*, and *m*3 = 2*n* + 1, we have

$$X=\boldsymbol{x_3}+(\boldsymbol{2^n}+1)|\boldsymbol{k_1(x_1-x_3)}+\boldsymbol{k_2(2^n-1)(x_2-x_1)}|_{\boldsymbol{2^n(2^n-1)}} \qquad (8)$$

It can be proved that the multiplicative inverses of |2n + 1|2n (2n−1) and |22*n* − 1|2*n* are given by *k*1 = 22*n*−1 − *(*2*n* − 1*)* and *k*2 = −1, respectively. These closed form expressions of *k*1 and *k*2 are proved as follows.

Proof of k1 = 2^{2n}−1 − (2^n− 1)

$$|k_1(2^n+1)|_{2^n(2^n-1)}=|[2^{2n-1}\text{-}(2^n-1)](2^n+1)|_{2^n(2^n-1)}=|2^{3n-1}\text{-}2^{2n-1}+1||_{2^n(2^n-1)}$$
$$=|2^{2n-1}(2^n-1)+1||_{2^n(2^n-1)}=1.$$

proof of $k_2=-1$:

$$|k_2(2^{2n}-1)||_{2^n}=|-1\times(2^{2n}\text{-}1)|_{2^n}=|\text{-}2^{2n}+)|_{2^n}=1.$$

Substituting the values 0f k_1 and k_2 in to (8), we have

$$X=x_3+(2^n+1)|2^{2n-1}-(2^n-1(x1--x3)-2n-1(x2-x1)|2n(2n-1)$$
$$=x_3+(2^n+1)||2^{2n-1}(x_1-x_3)-(2^n-1)(x_2-x_{3)||_{2^n(2^n-1)}} \qquad (9)$$

By scaling X by 2^{2n}-1,the scalied integer Y can be obtained by

$$Y=\left\lfloor\frac{X}{2^{2n}-1}\right\rfloor=\left\lfloor\frac{x_3}{2^{2n}-1}\right\rfloor+\left\lfloor\frac{(2^n+1)Z}{2^{2n}-1}\right\rfloor \qquad (10)$$

Where

Z = |22n−1(x1 − x3) − (2n − 1)(x2 − x3)|2n(2n−1).

Since *x*3 [0*,* 2*n*]*, x*3 *<* 22*n* − 1. Therefore *_(x*3*/*22*n* − 1*)* = 0, and *Y* can be written as

$$Y=\left\lfloor\frac{(2^n+1)Z}{2^{2n}-1}\right\rfloor=\left\lfloor\frac{Z}{2^n-1}\right\rfloor$$
$$=\left\lfloor\frac{|2^{2n-1}(x_1-x_3)-(2^n-1)(x_2-x_3)|_{2^n(2^n-1)}}{(2^n-1)}\right\rfloor. \qquad (11)$$

As *_(*|*x*|*m*1*m*2 */m*1*)* = |*_(x/m*1*)* |*m*2 from [11], (11) can be rewritten as

$$Y=\left|\left\lfloor\frac{2^{2n-1}(x_1-x_3)-(2^n-1)(x_2-x_3)}{(2^n-1)}\right\rfloor\right|_{2^n}$$
$$=\left|\left\lfloor\frac{2^{2n-1}(x_1-x_3)}{2^n-1}\right\rfloor+x_3-x_2\right|_{2^n}. \qquad (12)$$

Let *H* = 22*n*−1*(x*1 − *x*3*)*. Since *H* = *m_(H/m)* + |*H*|*m* for any integer *H* and *m*, we have

$$H=(2^n-1)\left\lfloor\frac{H}{2^n-1}\right\rfloor+|H|_{2^n-1}. \qquad (13)$$

Taking mod 2*n* operation on both the sides of (13), we have

$$|H|_{2^n}=\left|(2^n-1)\left\lfloor\frac{H}{2^n-1}\right\rfloor\right|_{2^n}+||H|_{2^n-1}|_{2^n}. \qquad (14)$$

Since

|*H*|2*n* = |22*n*−1*(x*1 − *x*3*)*|2*n* = 0 and |2*n* − 1|2*n* = −1

$$\left|\left\lfloor\frac{H}{2^n-1}\right\rfloor\right|_{2^n}=||H|_{2^n-1}|_{2^n}. \qquad (15)$$

Substituting (15) into (12), we have

$$Y=||H|_{2^n-1}+x_3-x_2|_{2^n}$$
$$=||2^{2n-1}(x_1-x_3)|_{2^n-1}+x_3-x_2|_{2^n}. \qquad (16)$$

If *Y€* [0*,* 2*n*−1*)*, *X* falls in the lower half range of M and ($x1$, $x2$, $x3$) represents a positive integer, i.e., $\hat{X} \geq 0$. Otherwise, if Y [$2n-1$, $2n$), X falls in the upper half range of M and ($x1$, $x2$, $x3$) represents a negative integer, i.e., $\hat{X} < 0$.

IV. HARDWARE IMPLEMENTATION

The residues $x1$, $x2$, and $x3$ can be represented in a binary form as $x1 = x1,n-1x1,n-2 \ldots x1,0$, $x2 = x2,n-1x2,n-2 \ldots x2,0$ and $x3 = x3,n\ x3,n-1 \ldots x3,0$, respectively, where xi, j denotes the j th bit of the residue xi . The binary vectors of $x1$ and $x2$ are of n bits but the binary vector of $x3$ is of $n + 1$ bits. In (16), one of the terms in the modulo $2n - 1$ sum involves the operation $|-22n-1\ x3|2n-1$, which cannot be directly implemented by Property 2, since $x3$ has n+1 bits. To apply the CLS property on the one's complement of $x3$ as in (6), $x3$ is expressed as $x3 = 2n\ x3,n + x3,n-1x3,n-2 \ldots x3,0$. Since $|2n\ x3,n|2n-1 = x3,n$, the MSB $x3,n$ of $x3$ can be logically OR with $x3,0$ to form an n-bit binary vector x_3= $|x3|2n-1$ =$|x3,n-1x3,n-2 \ldots$ x_3,0 $|2n-1$, where x_3,0= $x3,0 \vee x3,n$ and $\vee$ denotes a logical OR operator.

$|H|2n-1$ in (16) can then be implemented using the CLS operations of Properties 1 and 2 to obtain

$$Y = ||u1 + u2|2^n-1 + x3 - x2|2^n \quad (17)$$

where

$u1 = |22n-1\ x1|2n-1 = \text{CLS}\ n\ (x1,\ 2n - 1) = x1,0\ x1,n-1 \ldots x1,1$ (18)

$u2 = |22n-1\ \overline{x3}|2n-1 = \text{CLS}n(\overline{x\_3}, 2n - 1) = \overline{x}_{3,0}\ x3,n-1 \ldots \overline{x}3,1$ (19)

The term $|u1 + u2|2n-1$ can be expressed as

$|u1 + u2|2n-1 =$ _$|u1 + u2 + 1|2n$, if $u1 + u2 \geq 2n - 1$ $u1+u2$,otherwise. (20)

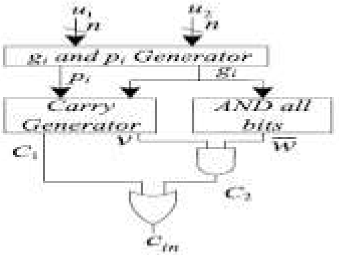


Figure 2 Generation of carry-in signal C_{in}

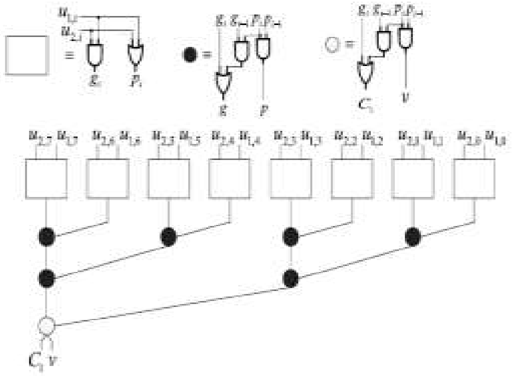


Figure 3 Example of the generation of the carry signal C1 and v for n=8

Hence,

$||u1 + u2|2n-1|2n = |u1 + u2 + cin|2n$, where $cin \in \{0, 1\}$. As $|-x2|2n = 2n - x2 = \overline{x2} + 1$, (17) can be written as

$$Y = |u1 + u2 + c_{in} + x3 + \overline{x2} + 1|2^n. \quad (21)$$

The generation of the carry-in signal cin is shown in Fig. 2. The condition $u1 + u2 \geq 2n$ is detected by $C1 = 1$ and the signal $C1$ can be generated by parallel prefix operators [2]. As an example, the carry signal $C1$ for $n = 8$ can be generated by the circuit shown in Fig. 3. The condition $u1+u2 = 2n-1 = 11 \ldots 11$ can be detected

by $C2 = 1$. $C2$ is generated by $\overline{w} \wedge v$, where $w = \wedge n-1\ i=0\ gi$ and $v = pn-1:0 = \wedge n-1\ i=0\ pi$, where $\wedge$ denotes a logical AND operator. The signals gi and $pn-1:0$ have already been generated in the computation of $C1$. Consequently, the condition $u1+u2 \geq 2n - 1$ for $cin = 1$ can be detected by

$$c_{in} = C1 \vee C2. \quad (22)$$

The two addends, $u2$ and $x3$, in (21) can be further simplified as follows:

$$\begin{aligned} |u_2 + x_3|_{2^n} &= ||2u_2|_{2^n} - u_2 + |x_3|_{2^n}|_{2^n} \\ &= \Big|\underbrace{\bar{x}_{3,n-1}\bar{x}_{3,n-2}\ldots\bar{x}_{3,1}0}_{n} + |x_3|_{2^n} - u_2\Big|_{2^n} \\ &= \Big|\underbrace{\bar{x}_{3,n-1}\bar{x}_{3,n-2}\ldots\bar{x}_{3,1}\bar{x}_{3,0}}_{n} - \bar{x}_{3,0} + |x_3|_{2^n} - u_2\Big|_{2^n} \\ &= ||\bar{x}_3|_{2^n} + |x_3|_{2^n} - u_2 - \bar{x}_{3,0}|_{2^n} \\ &= |2^n - 1 - u_2 - \bar{x}_{3,0}|_{2^n} = |\bar{u}_2 - \bar{x}_{3,0}|_{2^n} \\ &= |x'_{3,0}x_{3,n-1}x_{3,n-2}\ldots x_{3,1} - \bar{x}_{3,0}|_{2^n}. \end{aligned} \quad (23)$$

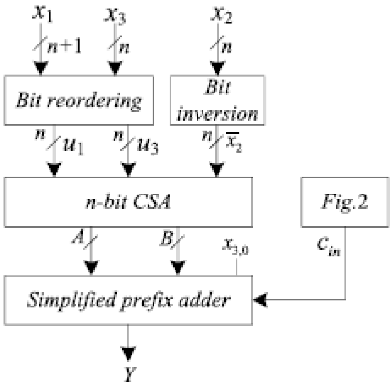


Figure 4 Proposed sign detection for {2*n* − 1, 2*n*, 2*n* +1}

When *x*3,*n* = 0, *x*_ 3,0 = *x*3,0 ∨ 0 = *x*3,0. Then |u2 + x3|2n = |x3,0x3,n−1x3,n−2 . . . x3,1 − x3,0|2n . (24)

When *x*3,*n* = 1, since *x*3 ∈ [0, 2*n*], *x*3,*n*−1*x*3,*n*−2 . . . *x*3,0 =00 . . . 0. Hence, *x*_3,0= *x*3,0 ∨ *x*3,*n* = 1 and

$$\begin{aligned} |u_2 + x_3|_{2^n} &= \Big|\underbrace{100\ldots0}_{n} - 1\Big|_{2^n} = \underbrace{011\ldots1}_{n} \\ &= \Big|\underbrace{x_{3,0}x_{3,n}x_{3,n}\cdots x_{3,n}}_{n} - \bar{x}_{3,0} + x_{3,n}\Big|_{2^n} \\ &= \Big|\underbrace{x_{3,n}00\ldots0}_{n} - \bar{x}_{3,0}\Big|_{2^n}. \qquad (25) \end{aligned}$$

To satisfy both (24) and (25)

|u2 + x3|2n = |u3 − x3,0|2n (26)

where the *n*-bit binary vector *u*3 is given by

u3 = (x3,0 ∨ x3,n)x3,n−1x3,n−2 ..x3,1. (27)

Substituting (26) into (21), we have

Y = |u1 + u3 + x2 + cin + 1− x3,0|2n . (28)

If *x*3,0 = 1, 1 − x3,0 = 1, and if *x*3,0 = 0, 1 − x3,0 = 0. Hence, the term 1− x3,0 in (28) can be replaced by *x*3,0 and

Y = |u1 + u3 + x2 + cin + x3,0|2n . (29)

The sign of ˆ*X* can be detected by the MSB of *Y*. An *n*-bit carry save adder (CSA) can be used to add the three *n*-bit operands, *u*1, *u*3, and x2, to produce an *n*-bit sum *A* = *an*−1*an*−2 . . . *a*0 and an *n*-bit carry vector *B* = *bn*−1*bn*−2 . . . *b*10. Due to the modulo 2*n* addition, the final carry output bit *bn* of the CSA need not be generated. As *b*0 = 0, it can be replaced by *x*3,0 of (29) before the MSB of *Y* is computed by a simplified parallel prefix adder of *A* and *B* with the input carry bit *c*−1 = *c*in. The prefix adder is simplified by keeping only the carry generation network for the computation of carry signal *cn*−1, from which the sign of ˆ*X* can be detected by sign(ˆ*X*) = *an*−1 ⊕ *bn*−1 ⊕*cn*−1. The architecture of the proposed sign detector is shown in Fig. 4, where the circuit diagram for the simplified prefix adder is depicted in Fig. 5 for *n* = 8.

Example 1: For *n* = 5, {*m*1, *m*2, *m*3} = {31, 32, 33}, *M* = 31 × 32 × 33 = 32 736, and *M/*2 = 16 368. The signed integer ˆ*X* = −11 161 can be represented by the residue representation (*x*1, *x*2, *x*3*)* = *(*30, 7, 26*)* corresponding to the unsigned integer *X* = 21 575 in the same moduli set.

The binary representation of the residues are *x*1 = 111102, *x*2 = 001112, and *x*3 = 0110102. According to (18), (19), and (27), *u*1 = 011112, *u*2 = 100102, and *u*3 = 011012. Also, *x*3,0 = 0. Since *u*1 + *u*2 = 01111 + 10010 = 33 > 32, *C*1 = 1. Since 33 _= 31, *C*2 = 0. According to (22), *c*in = *C*1 ∨ *C*2 = 1. The computation of *Y* in (29) is illustrated in Fig. 6. Since MSB of *Y* = 1, the integer ˆ*X* represented by (30, 7, 26*)* is negative.

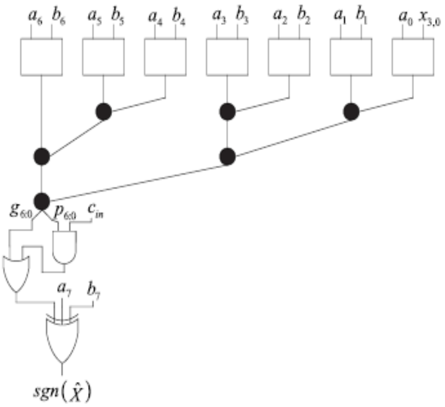
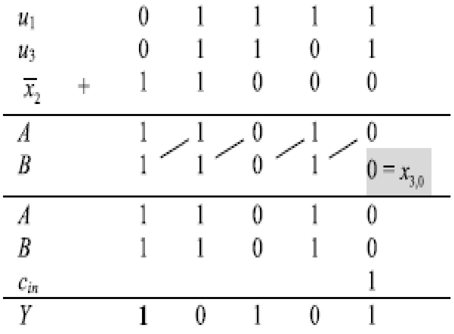


Figure 5 Simplified prefix adder for n=8



u_1		0	1	1	1	1
u_3		0	1	1	0	1
$\overline{x}_2$	+	1	1	0	0	0
A		1	1	0	1	0
B		1	1	0	1	$0 = x_{3,0}$
A		1	1	0	1	0
B		1	1	0	1	0
c_{in}						1
Y		1	0	1	0	1

Figure 6 Computation of Y for example 1

V. SIMULATION & SYNTHESIS RESULTS

The proposed adder based sign detector is synthesised and simulated using XILINX ISE. Fist the design is coded using Verilog and then is synthesised in XILINX and verified the design without any errors. Next the design is implemented on SPARTAN 3E FPGA family. And the stimulus to the design is applied trough test feature, to very the design.

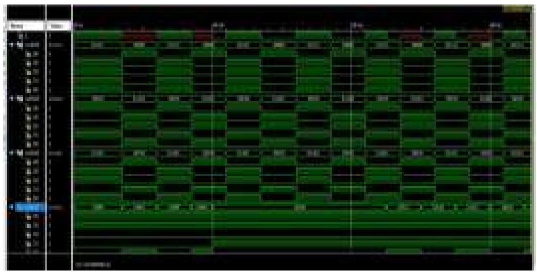


Figure 7 simulation result of proposed adder based sign detector

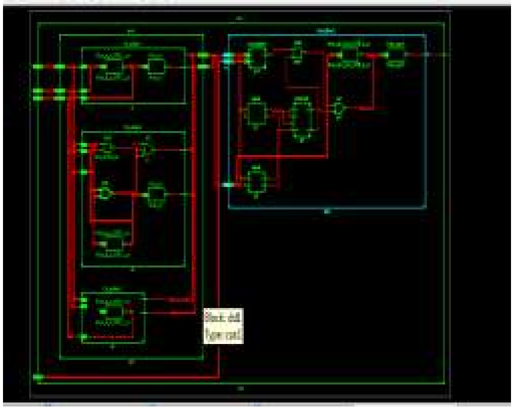


Figure 8 RTL schematic of proposed adder based sign detector

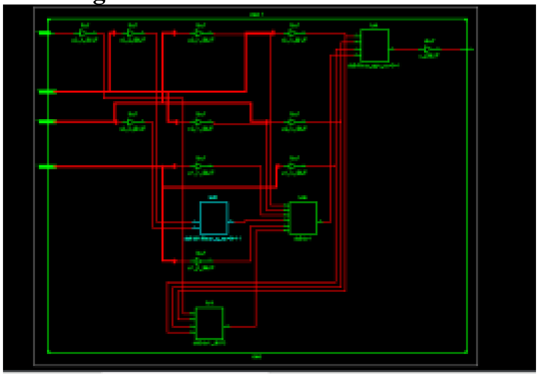
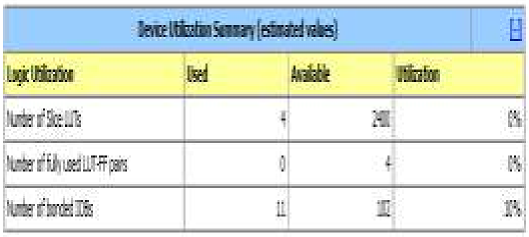


Figure .9 Technological schematic diagram.



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	4	2400	0%
Number of fully used LUT-FF pairs	0	4	0%
Number of bonded IOBs	11	108	10%

Figure.10 Summary of device utilization

COMPARISON TABLE

	Area	Delay
Existing Method	Number of slices used=3986	89.55ns
Proposed method	Number of slices used=4	8.105ns

VI. CONCLUSION

In this brief, a new sign detection algorithm for RNS $\{2^n - 1, 2^n, 2^n + 1\}$ is proposed, which leads to a high-speed and area-efficient adder-based implementation. Our experimental results show that the proposed circuit smaller, faster, and more power efficient than the latest existing sign detectors, respectively. The work presented in this thesis can be extended in several ways. As RNS seems to be suitable for many modern algorithms, investigating more applications is one possibility of future work. Further applications in signal processing, e.g., echo cancellations are worth further investigation, as well as the circuits with imprecision.

REFERENCES

[1] R. Muralidharan and C.-H. Chang, “Area-power efficient modulo 2n−1 and modulo 2n +1 multipliers for {2n −1, 2n,2 n +1} based RNS,”IEEE Trans. Circuits Syst. I, Reg. Papers , vol. 59, no. 10, pp. 2263–2274, Oct. 2012.

[2] H. T. Vergos and G. Dimitrakopoulos, “On modulo 2n+1 adder design,” IEEE Trans. Comput., vol. 61, no. 2, pp. 173–186, Feb. 2012.

[3] H. T. Vergos, “A family of area-time efficient modulo 2n+1 adders,” in Proc. IEEE Comput. Soc. Annu. Symp. VLSI, Lixouri, Kefalonia, Greece, Jul. 2010, pp. 442–443.

[4] L.-S. Didier and L. Jaulmes, “Fast modulo 2n−1 and 2 n+1 adder usingcarry-chain on FPGA,” in Proc. Asilomar Conf. Signals, Syst., Comput., Pacific Grove, CA, USA, Nov. 2013, pp. 1155–1159.

[5] N. S. Szabó and R. I. Tanaka, Residue Arithmetic and Its Applications to Computer Technology. New York, NY, USA: McGraw-Hill, 1967.

[6] R. Conway and J. Nelson, "Improved RNS FIR filter architectures," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 51, no. 1, pp. 26–28, Jan. 2004.

[7] E. Vassalos, D. Bakalis, and H. T. Vergos, "RNS assisted image filtering and edge detection," in Proc. IEEE 18th Int. Conf. Digit. Signal Process., Fira, Santorini, Greece, Jul. 2013, pp. 1–6.

[8] J.-C. Bajard, L.-S. Didier, and T. Hilaire, "ρ-direct form transposed and residue number systems for filter implementations," in Proc. IEEE 54th Int. Midwest Symp. Circuits Syst., Seoul, Korea, Aug. 2011, pp. 1–4.

[9] Y. Liu and E. M.-K. Lai, "Design and implementation of an RNS-based 2-D DWT processor," IEEE Trans. Consum. Electron., vol. 50, no. 1, pp. 376–385, Feb. 2004.

[10] T. F. Tay, C.-H. Chang, and J. Y. S. Low, "Efficient VLSI implementation of 2n scaling of signed integer in RNS {2n−1, 2n,2 n+1}," IEEE Trans.Very Large Scale Integr. (VLSI) Syst. , vol. 21, no. 10, pp. 1936–1940, Oct. 2013.

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

[12] Z. Wang, G. A. Jullien, and W. C. Miller, "An improved residue-tobinary converter," IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 47, no. 9, pp. 1437–1440, Sep. 2000.

[13] H. Brönnimann, I. Z. Emiris, V. Y. Pan, and S. Pion, "Computing exact geometric predicates using modular arithmetic with single precision," in Proc. 13th Annu. Symp. Comput. Geometry, Nice, France, Jun. 1997, pp. 174–182.

[14] Z. D. Ulman, "Sign detection and implicit-explicit conversion of numbers in residue arithmetic," IEEE Trans. Comput., vol. 32, no. 6, pp. 590–594, Jun. 1983.

[15] T. Van Vu, "Efficient implementations of the Chinese remainder theorem for sign detection and residue decoding," IEEE Trans. Comput., vol. 34, no. 7, pp. 646–651, Jul. 1985.

[16] M. Xu, Z. Bian, and R. Yao, "Fast sign detection algorithm for the RNS moduli set {2n+1−1,2n−1,2n}," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 2, pp. 379–383, Feb. 2015.

[17] C. V. Niras and Y. Kong, "Fast sign-detection algorithm for residue number system moduli set {2n−1, 2n,2 n+1−1}," IET Comput. Digit.Techn. , pp. 1–5, Sep. 2015. DOI: 10.1049/iet-cdt.2015.0050

[18] M. Xu, R. Yao, and F. Luo, "Low-complexity sign detection algorithm for RNS {2n −1, 2n,2 n +1}," IEICE Trans. Electron., vol. E95-C,no. 9, pp. 1552–1556, Sep. 2012.

[19] T. Tomczak, "Fast sign detection for RNS (2n −1, 2n,2 n +1)," IEEETrans. Circuits Syst. I, Reg. Papers , vol. 55, no. 6, pp. 1502–1511, Jul. 2008.

[20] M. Akkal and P. Siy, "Optimum RNS sign detection algorithm using MRC-II with special moduli set," J. Syst. Archit., vol. 54, no. 10, pp. 911–918, Oct. 2008.

[21] L. Sousa and P. Martins, "Efficient sign identification engines for integers represented in RNS extended 3-moduli set {2n −1, 2n+k,2 n +1},"Electron. Lett. , vol. 50, no. 16, pp. 1138–1139, Jul. 2014.

[22] C.-H. Chang and S. Kumar, "Area-efficient and fast sign detection for four-moduli set RNS {2n−1, 2n,2 n +1, 22n +1}," in Proc. IEEE Int.Symp. Circuits Syst. (ISCAS) , Melbourne, VIC, Australia, Jun. 2014, pp. 1540–1543.

[23] Y. Wang, "New Chinese remainder theorems," in Proc. Conf. Rec. 32nd Asilomar Conf. Signals, Syst., Comput., Pacific Grove, CA, USA, pp. 165–171, Nov. 1998.

[24] A. Tyagi, "A reduced-area scheme for carry-select adders," IEEE Trans. Comput., vol. 42, no. 10, pp. 1163–1170, Oct. 1993.

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