Abstract—In this paper, a preprocessing architecture for folding Analog-to-Digital converter (ADC) based on a new number system called Robust Folding Number System (RFNS) is presented. The RFNS is developed from robust symmetrical number system (RSNS) that was proposed by Pace et al [4]. The enhanced Dynamic Range (DR) of RFNS due to the introduction of the folding bit results in higher ADC resolution. The Symmetrical Residues (SRs) that are obtained in the intermediate step during processing of analog signals are converted to equivalent residues by simple mapping. This results in a conversion architecture that is less complex compared to that proposed in [6].

I. INTRODUCTION

Symmetrical Number Systems (SNS) are modular systems that extract maximum information from symmetrically folded waveforms [1]. Processing analog signals using SNS consists in decomposing an analog signal into a number of sub operations (moduli) that are of smaller computational complexity. In each sub operation, integers (SR) are extracted from the symmetrically folded waveform whose folding period is equal to the modulus of the sub channel. Higher resolution in conversion is obtained when results from different SNS moduli are recombined. Optimum Symmetrical Number System (OSNS) with a larger DR than that of SNS was proposed in [2]. Although the OSNS ADC uses fewer comparators, encoding errors occur when the input signal lies about any code transition point [2]. To eliminate these errors, the Robust Symmetrical Number System (RSNS) was proposed and investigated in [3]-[6].

RSNS is a modular system in which integer values within each modulus (comparator state) change one at a time at the next code position (integer Gray-code properties). Each channel modulus \( m_i \) symmetrically folds the analog signal with folding period equal to \( 2N m_i \), where \( N \) represents the number of channels. For the RSNS, each channel requires \( m_i \) comparators. However, DR of RSNS is smaller compared to that of OSNS due to the repetition of SRs in each modulus. Use of RSNS in ADC eliminates encoding errors that occur due to asynchronous switching of comparators. One major disadvantage in using RSNS in ADC is that conversion of SRs into binary representation is complex [6]. Complexity in conversion arises due to ambiguities introduced in folding the analog signal (SR is same for two different input voltages due to folding). Due to ambiguities, Chinese Remainder Theorem (CRT) cannot be directly applied in SR to binary conversion.

In this paper, we propose a novel number system called Robust Folding Number System (RFNS) that overcomes ambiguities introduced due to folding. Further, by appending folding bit (FB) that is derived from the folding circuit, this number system has a larger DR that results in a higher resolution using the same number of comparators as that used in RSNS converters. While RSNS needs General CRT for reverse conversion to binary equivalents, a vastly simpler conversion algorithm from RFNS to binary equivalents by applying CRT directly is proposed with the use of FB.

II. ROBUST SYMMETRICAL NUMBER SYSTEMS

A. Robust Symmetrical Number System

The RSNS is based on the following sequence:

\[
x_m = [0, 1, \ldots, m-1, m, m-1, \ldots, 2, 1]. \tag{1}
\]

To form an \( N \)-sequence RSNS, where \( N \) is the number of times each term (SR) in (1) is repeated in succession. Thus, an \( N \)-sequence RSNS is of the form:

\[
x_h = [0, 0, \ldots, 0, 1, \ldots, m-1, m-1, \ldots, m-1, 1, \ldots, 1].
\]

In this sequence, each value within the row vector is in \( N \) times in succession. This sequence is repeated in both directions, forming a periodic sequence with the period \( P_{RSNS} = 2mN \). Each sequence corresponding to \( m_i \) is left-shifted by \( s_i = i - 1 \) places for \( i \in \{1, \ldots, N\} \). The shift values \( \{s_1, s_2, \ldots, s_N\} \) form a complete residue system modulo \( N \). For example, Fig. 1 shows integer \( h \) and corresponding SRs in each modulus for an \( N = 3 \) RSNS with \( m_1 = [3 4 5] \),

\[
s_i = [0 1 2] \tag{5}.
\]

\[
X_3 = a \times \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{bmatrix} = X \tag{6}
\]

Fig. 1. Three-sequence RSNS structure.

There are three types of ambiguities in each sequence as illustrated in Fig. 2 for \( m = 5 \). They are labeled Type 0, Type 1, and Type 2. The ambiguities from period to period are defined as Type 0. The ambiguity on the rise of the fold sequence
and on the opposite side of the fold is defined as Type 1. Each term within the sequence is repeated $N = 3$ times and is defined as a Type 2 ambiguity [5]. Type 0 ambiguity of RSNS is the same as other number systems. Due to Type 1 and Type 2 ambiguities, the RSNS-to-Binary conversion is complicated [6].

For the case $[e, e, e]^T$, and $[o, o, o]^T$, or
\[ h \equiv 3d_1 + 1 (mod 6m_1) \]
\[ h \equiv 3d_2 + 1 (mod 6m_2) \]
\[ h \equiv 3d_3 - 2 (mod 6m_3) \] (4)

For the case $[e, e, o]^T$, and $[o, o, e]^T$, or
\[ h \equiv 3d_1 + 2 (mod 6m_1) \]
\[ h \equiv 3d_2 - 1 (mod 6m_2) \]
\[ h \equiv 3d_3 - 1 (mod 6m_3) \] (5)

For the case $[e, o, o]^T$, $[o, e, e]^T$. The value $h$ that is obtained using one of the above sets of congruences will be within smallest DR. All the systems have the same DR is $6m_1m_2m_3$. It is easy to prove by GCRT. Therefore, the DR of the moduli set $\{m_1, m_2, m_3\}$ in RFNS is $M_{RFNS} = 6m_1m_2m_3$.

The RSNS Gray-code properties make it particularly attractive for error control. However, the DR of RSNS is small. For example, consider moduli set $[3 4 5]$, the DR is 43. Due to small DR value, this RSNS moduli set can be used for a 6-bit ADC only. To overcome this disadvantage of RSNS, we propose RFNS that inherits Grey-code properties like RSNS. However, the RFNS has DR that is much larger than RSNS.

B. Robust Folding Number System

1) Definition: To eliminate the Type 1 ambiguity in RSNS, we introduce the folding bit. The SRs in RFNS are the same as those in RSNS. However, ambiguity due to folding is eliminated by the introduction of a folding bit (FB) that indicates either the rising or falling nature of the folded waveform. During the first half of folding (+ slope) the FB ($f$) is 0 and that during the second half (- slope) is 1. If $d$ is the decimal offset within the folding period $T$, then SRs corresponding to any $d$ can be determined as follows.

\[
\begin{align*}
  s &= d, & f &= 0 & \text{if } d \leq m_i \\
  s &= 2m_i - d, & f &= 1 & \text{if } m_i + 1 \leq d \leq 2m_i - 1
\end{align*}
\] (2)

(2) shows the relationship between SRs and decimal offsets within $T$ for $f = [0, 1]$. From (2), SR and FBs can be determined once $d$ is known or vice versa.

2) Properties: The DR of $\{m_1, m_2, m_3\}$ moduli set in RFNS is $M_{RFNS} = 6m_1m_2m_3$ and this is larger than that in RSNS. For example, consider $[3 4 5]$ moduli set and the SRs, offset and FB are shown in Fig. 3. The DR in RSNS is 43 while that for the same set in RFNS is 360. This can be proved when General Chinese Remainder Theorem (GCRT) is applied for converting SRs into their decimal equivalent $h$.

The SR parity of RSNS in Fig 1 and RFNS in Fig. 3 is identical as shown in Fig. 4, where $e$ is for even and $o$ is for odd parity of SR. Depending on the value of SR parity, a decimal $h$ that is obtained after reverse conversion will correspond to one of the following sets of congruence equations.

\[
\begin{align*}
  h &\equiv 3d_1 (mod 6m_1) \\
  h &\equiv 3d_2 (mod 6m_2) \\
  h &\equiv 3d_3 (mod 6m_3)
\end{align*}
\] (3)

III. ADC BASED ON RFNS

In this work, we propose a folding ADC preprocessing that employs RFNS. An 8-bit ADC preprocessing using $[3 4 5]$ moduli RFNS system is illustrated as an example in Fig. 5. The proposed method can be applied to higher resolution ADC preprocessing.
Fig. 3. Symmetrical residue \( s \), offset \( d \) and folding bit \( f \) of moduli set \([3, 4, 5]\).

Fig. 5. Schematic diagram of 8-bit RFNS-based ADC with \( m_1 = 3 \), \( m_2 = 4 \) and \( m_3 = 5 \).

A. Analog Preprocessing and FB generation

The analog signal in each sub channel is folded based on the modulus using folding circuits like those in [1]-[3]. SRs are obtained by applying folded waveforms across a bank of comparators with proper thresholds as shown in Fig. 6. FBs for each folded waveform is extracted from the folding circuit in sub channels [9][10]. The number of comparators required in each sub channel is equal to the modulus of the sub channel. In proposed 8-bit RFNS ADC preprocessing, the total number of comparators required is \( (3 + 4 + 5 = 12) \) which is the same as that used in RSNS. However, the ADC using RSNS is only 6-bits. Moreover, the conversion of SRs in RFNS to binary is vastly simpler compared to SRs in RSNS.

B. RFNS to binary equivalent converter

The preprocessed output from comparators are converted to SRs as described above. Ambiguity Type 1 of SRs is eliminated by appending the FB. The reverse conversion from SRs to binary equivalent is much simpler than RSNS reverse conversion. Consider a three moduli RFNS system \( \{m_1, m_2, m_3\} \) with DR \( 6m_1m_2m_3 \). Let the unknown incoming value \( h \), \( 0 \leq h \leq 6m_1m_2m_3 - 1 \), have SRs and \( f \) as \( (f_1, s_1), (f_2, s_2), (f_3, s_3) \) with respect to offsets \( d_1, d_2, d_3 \).

Lemma

If \( h \equiv d_1 \text{ (mod } 6m_1) \) and \( a_1 = d_1 \text{ mod } m_1 \) then \( h \equiv a_1 \text{ (mod } m_1) \).

Proof

Since \( h \equiv d_1 \text{ (mod } 6m_1) \) \( \Rightarrow \) \( h = x6m_1 + d_1 \)

Since \( a_1 = d_1 \text{ mod } m_1 \)

\( \Rightarrow d_1 = ym_1 + a_1 \Rightarrow h - 6xm_1 = ym_1 + a_1 \)

\( \Rightarrow h = (6x + y)m_1 + a_1 \Rightarrow h \equiv a_1 \text{ (mod } m_1) \)

For example, we have \( 23 \equiv 5 \text{ (mod } 18) \). Then 23 also satisfies \( 23 \equiv 2 \text{ (mod } 3) \). Where 2 is from \( 2 = 5 \text{ mod } 3 \).

The SR parity shown in Fig. 4 is used to select the system of congruences to be solved to obtain the binary equivalent. \( h \) satisfies (3) whenever the parity of SRs is of the form \([e, e, e]^T\) and \([o, o, o]^T\) and satisfies (4) whenever the parity is of the form \([e, e, o]^T\), \([e, o, o]^T\), and \([o, e, e]^T\) and finally, satisfies (5) whenever the parity is of the form \([e, o, o]^T\), and \([e, e, e]^T\). The following steps form the basis for RFNS to binary conversion.

Step 1 When FB \( f \) is 0, we retain the SRs as offsets \( d_i = s_i \) and when \( f = 1 \), we convert the SRs to offsets by using a simple mapping as \( d = 2m_i - s \). Hence \( d_i = s_i \) if \( f = 0 \), \( d = 2m_i - s \) if \( f = 1 \).

Step 2 From the parity table in Fig. 4 we choose which system of congruences, viz., (3) (4) (5) is to be solved. That is, to look for \( h \) that satisfies:

\[
\begin{align*}
    h &\equiv x_1 \text{ (mod } 6m_1) \, , \\
    h &\equiv x_2 \text{ (mod } 6m_2) \, , \\
    h &\equiv x_3 \text{ (mod } 6m_3) \\
\end{align*}
\]

Here (6) takes care of all representations given by (3) (4) (5).

We compute \( a_1 = x_1 \text{ mod } m_1 \), \( a_2 = x_2 \text{ mod } m_2 \), \( a_3 = x_3 \).
\( h \equiv a_1 (\text{mod } m_1) \)
\( h \equiv a_2 (\text{mod } m_2) \)
\( h \equiv a_3 (\text{mod } m_3) \)  

(7)

Now, our problem is to solve the system of (7).

**Step 3** Apply CRT to solve (7). The basic solution of (7) computed by CRT is \( h = h_0 \), \( 0 \leq h_0 \leq L/6 \). Out of the many solutions for \( h \) we choose \( h \) so that \( 0 \leq h_0 \leq L/6 \) and \( h_1 = h_0 + L/6, h_2 = h_0 + 2L/6, h_3 = h_0 + 3L/6, h_4 = h_0 + 4L/6, h_5 = h_0 + 5L/6 \). Only one value for \( h \), one of \( \{h_0, \ldots, h_5\} \) will satisfy (6) and that is the binary equivalent of the analog input.

**Example** Consider \((f_1, s_1) = (0, 2), (f_2, s_2) = (0, 0)\) and \((f_3, s_3) = (1, 1)\) with respective to the moduli set \([3 4 5]\).

**Step 1:** Convert to offset domain \( d_1 = s_1 = 2, d_2 = s_2 = 0, d_3 = 2 \times 5 - 1 = 9 \).

**Step 2:** Because \((s_1, s_2, s_3) = (2, 0, 1) = [e, e, 0]^T\) so that we choose (4) to solve to get the decimal \( h \).

\[
\begin{align*}
 h &\equiv 7 \pmod{18} \\
 h &\equiv 1 \pmod{24} \\
 h &\equiv 25 \pmod{30}
\end{align*}
\]

We compute

\[
\begin{align*}
 a_1 &= x_1 \pmod{m_1} = 7 \pmod{3} = 1 \\
 a_2 &= x_2 \pmod{m_2} = 1 \pmod{4} = 1 \\
 a_3 &= x_3 \pmod{m_3} = 25 \pmod{5} = 0 \\
\end{align*}
\]

We look for \( h \) that satisfies

\[
\begin{align*}
 h &\equiv 1 \pmod{3} \\
 h &\equiv 1 \pmod{4} \\
 h &\equiv 0 \pmod{5}
\end{align*}
\]

(9)

**Step 3:** Apply CRT to (9) to get solution in the range \( 0 \leq h < 60 \), we get \( h = 25 \). Then we try all other \( h \)'s that are different 60. We see that only \( h = 25 \) that satisfies the (8). Thus, \( h = 25 \) is the solution.

**C. Discussion**

The preprocessing (folding part) has been simulated using a 0.35 \( \mu m \) CMOS process with a 3.3 V power supply in HSPICE. To increase the ADC preprocessing resolution, more moduli channels can be added in parallel. If the resolution is increased, more folding circuits are required in each channel.

The reverse converter and the logic have been implemented in Project Navigator with Xilinx XC3S200 Field Programmable Gate Array (FPGA). An adder based residue to binary converter for \( \{2^n - 1, 2^k, 2^k + 1\} \) moduli set proposed by Y. Wang et al in [7][8] is used in the implementation of reverse converters of both RSNS and RFNS. A brief comparison between the reverse converters using RSNS and RFNS is given in Table I. In the comparison, an input/output block (IOB) refers to a collection or a group of basic elements that implement the input and output functions of the device.

**REFERENCES**


