FACTA UNIVERSITATIS (NIŠ)
SER.: ELEC. ENERG. vol. 22, no. 1, April 2009, 125-140

Digital FIR Filter Architecture Based on the Residue Number System

Negovan Stamenković

Abstract: In this paper, architecture of residue number system used in FIR filters, is presented. For many years residue number coding has been recognized as a system which provides capability for implementation of a high speed addition and multiplication. These advantages of residue number system coding for the high speed FIR filters design results from the fact that an digital FIR filter requires only addition and multiplication. The proposed FIR filter architecture is performed as series of modulo multiplication and accumulation across each modulo. A numerical example illustrates the principles of FIR filtering of an 32 order low pass filter. This architecture is compared with FIR filters direct synthesis.

Keywords: Digital signal processing, Residue Number System, Chinese Remainder Theorem, FIR filter,

1 Introduction

THE residue number system is a non-weighted number system which speeds up arithmetic operations by dividing them into smaller parallel operations. Since the arithmetic operations in each modulo are independent of each other, there is no carry propagation among them so residue number system is carry-free addition, multiplication and borrow-free substraction [1]. Residue number system is one of the most effective techniques for power dissipation reduction in VLSI system design [2].

Some application of the residue number system are digital signal processing [3–5]. Digital filters are especially important in DSP because they can be used for a wide variety of applications: noise-reduction, band splitting, band limiting,

Manuscript received on October 5, 2008.
Author is with Faculty of Natural science and Mathematics, Lole Ribara 29, 38220 Kosovska Mitrovica, Serbia (e-mail: negovanstamenkovic@gmail.com).

interpolation, decimation, pulse forming, echo suppression, equalization, etc. Two basic filter types are commonly implemented in DSP: Finite Impulse Response (FIR) filters and Infinite Impulse Response (IIR) filters. Both filter implementations includes binary-to-residue converter at its input, which converts the input data into equivalent residues. The filtering is mainly performed in the central block. Since there are L residues in the residue set, L sub-filters are used to process corresponding residues from the input. Reverse conversion is at the output; translation from residue representation back to binary notation is performed. In this paper architecture for the FIR filters implementation is proposed.

The complexity as well as the efficiency of residue to binary conversion and vice versa, is primary based on the proper selection of the modulo set and the conversion algorithm. Many different modulo set have been suggested, such as $\{2^n-1, 2^n, 2^n+1\}$ [6–9], $\{2^n, 2^n-1, 2^{n-1}-1\}$ [10,11], $\{2^n, 2^n-1, 2^{n+1}-1\}$ [12], $\{2^{2n}+1, 2^n+1, 2^n-1\}$ [13], $\{2^n, 2^{2n}-1, 2^{2n}+1\}$ [14]. $\{2^{2n}, 2^{n+1}+1, 2^{n+1}-1\}$ [15]. In this paper we have used three moduli set $\{2^n-1, 2^n, 2^n+1\}$. This set of modulo is very popular due to simple conversion from a positional binary number system as well as an efficient implementation of some arithmetic operations. Nevertheless, it has disadvantage that residue (2^n+1) requires $(n+1)$ bits to represent 2^n+1 states, which means that almost half of the states remain unused.

This paper is organized in following way: in Section 2 we have introduced the necessary background of the structure of simple finite fields and residue number system. Section 3 discuss a method for design of linear phase recursive digital filters and method for translating 2's-complement binary representation into residue number system and vice versa. Section 4 presents the proposed design methodology and the RNS filter architecture for linear phase FIR filter. Simulation impulse and steady state response of FIR filter in residue arithmetic are illustrated in Section 5.

2 Residue Number System

Let us introduce the basic terminology [6]:

1. The vector $\{m_1, m_2, \ldots, m_k\}$ forms a set of moduli, called the RNS base β, where m_i are relatively prime such that $\gcd(m_i, m_j) = 1$ for $i \neq j$, where gcd means the greatest common divisor of m_i and m_j.
2. M is the product $\prod_{i=1}^{k} m_i$ and defines the dynamic range of the system.
3. The vector $\{x_1, x_2, \ldots, x_k\}$ is the RNS representation of an integer X, $X < M$, where $x_i = \langle X \rangle_{m_i} = X \mod m_i$. Any integer X belonging to $\mathbb{Z}/M\mathbb{Z}$[1] has a

[1] The set of congruence classes in $\mathbb{Z}$ modulo M. Two integers Z_1 and Z_2 are said to be congruent modulo M, if their difference $(Z_1 - Z_2)$ is an integer multiple of M.

unique representation, in base β [16].

The dynamic range of representable numbers is usually partitioned into two approximately equal parts, such that approximately half of the numbers are positive and the rest are negative. Thus, every representable integer, X, which satisfy one of two relations:

$$\begin{aligned} -\frac{M-1}{2} \leq X \leq \frac{M-1}{2} \quad & \text{if } M \text{ is odd} \\ -\frac{M}{2} \leq X \leq \frac{M}{2} \quad & \text{if } M \text{ is even} \end{aligned}$$

can be represented in RNS form.

4. The operations of addition, subtraction and multiplication are defined over the set of congruence classes in $\mathbb{Z}/M\mathbb{Z}$ as:

$$\begin{aligned} A \pm B &= (\langle a_1 \pm b_1 \rangle_{m_1}, \ldots, \langle a_k \pm b_k \rangle_{m_k}) \\ A \times B &= (\langle a_1 \times b_1 \rangle_{m_1}, \ldots, \langle a_k \times b_k \rangle_{m_k}). \end{aligned} \tag{1}$$

These equations illustrate the parallel carry-free nature of the RNS.

5. The binary-to-residue converter is designed according to the following algorithm [17]. A K bit number X can be expressed as:

$$X = \sum_{j=0}^{K_1} b_j 2^j + \sum_{j=K_1+1}^{K} b_j 2^j \tag{2}$$

where b_0 is the sign bit. The residue of X mod m_i, where m_i, $i = 1,2,\ldots,N$, is the i-th modulus used to define the RNS code, can be writen using equation (2),

$$\begin{aligned} \langle X \rangle_{m_i} &= \Big\langle \sum_{j=0}^{K_1} b_j 2^j + \sum_{j=K_1+1}^{K} b_j 2^j \Big\rangle_{m_i} \\ &= \Big\langle \sum_{j=0}^{K_1} b_j 2^j \Big\rangle_{m_i} \oplus \Big\langle \sum_{j=K_1+1}^{K} b_j 2^j \Big\rangle_{m_i} \end{aligned} \tag{3}$$

where $\oplus$ represents modulo addition.

6. The reconstruction of X from its residues $\{x_1, x_2, \ldots, x_k\}$ is based on the Chinese Remainder Theorem:

$$X = \Big\langle \sum_{i=0}^{k} \langle \gamma_i x_i \rangle_{m_i} M_i \Big\rangle_M, \tag{4}$$

where $M = \prod_{i=1}^{k} m_i$; $M_i = M/m_i$; $\gamma_i = \langle M_i^{-1} \rangle_{m_i}$. The notation $\langle M_i^{-1} \rangle_{m_i}$ denotes the multiplicative inverse of M modulo m_i

$$\left\langle M_i^{-1} M \right\rangle_{m_i} = 1, \quad 0 \leq M,\ M_i^{-1} < m_i. \tag{5}$$

7. Another way to convert RNS representation into weighted form X is by using Mixed Radix Conversion [18]. The vector $\{x'_1, x'_2, \ldots, x'_k\}$, $0 \leq x'_i \leq m_i$ is the Mixed Radix System (MRS) representation of an integer X smaller than M, such that:

$$X = x'_1 + x'_2 m_1 + x'_3 m_1 m_2 + \cdots + x'_k \prod_{i=1}^{k-1} m_i \tag{6}$$

where $x'_i \in [0, m_i)$ are the mixed radix digits of X, and

$$\begin{aligned}
x'_1 &= x_1 \mod m_1 \\
x'_2 &= (x_2 - x'_1)c_{12} \mod m_2 \\
x'_3 &= ((x_3 - x'_1)c_{13} - x'_2)c_{23} \mod m_3 \\
&\vdots \\
x'_k &= (\cdots((x_k - x'_1)c_{1k} - x'_2)c_{2k} - \ldots - x'_{k-1})c_{k-1,k} \mod m_k
\end{aligned}$$

The constants c_{ij} are multiplicative inverse of m_i modulo m_j for all $1 \leq i < j \leq k$ ($c_{ij} \cdot m_i = 1 \mod m_j$ for $1 \leq i \leq n$) and can be computed using Euclid's algorithm [18, 19].

8. Comparison and division are very difficult operations to perform on the RNS representation [20–22]. The traditional techniques for the residue number comparison use the Chinese Remainder Theorem or the Mixed Radix Conversion. A direct implementation of the Chinese Remainder Theorem is unprofitable since it is based on the module M operation, where M is the large dynamic range of residue number. The Mixed Radix Conversion is a sequential process requiring long delay. Other techniques use redundant modulus and assume special condition of the moduli set [23].

3 The Design of RNS Digital Filter

Finite Impulse Response (FIR) digital filters have attracted a great deal of interest because they are inherently stable structures which are much less sensitive to quantization errors than filters of the recursive type.

An FIR filter is described by (7), where x_n is the input to the filter, b_k represents the filter coefficients, N is the filter order and y_n is the filter output

$$y_n = \sum_{k=0}^{N} b_k x_{n-k}. \tag{7}$$

For a very large N, filters implemented in traditional binary weighted number system suffer from disadvantages of carry propagation delay in binary adders and multipliers.

In RNS a large integer is broken into smaller residues which are independent of each other. Each residue digit is processed in parallel without carry propagation from one to another. This leads to significant speed up of multiply and accumulate (MAC) operations which in turn results in high data rate for RNS based FIR filters [2, 6].

A modulo set must be selected which provides just enough dynamic range for the FIR filter. A set comprised of a large number of small-valued integers will provide a highly parallel RNS structure while maintaining low memory requirements, for stored-table operations. Consider the 31th-order lowpass linear phase filter described by the magnitude response of Figure 1.

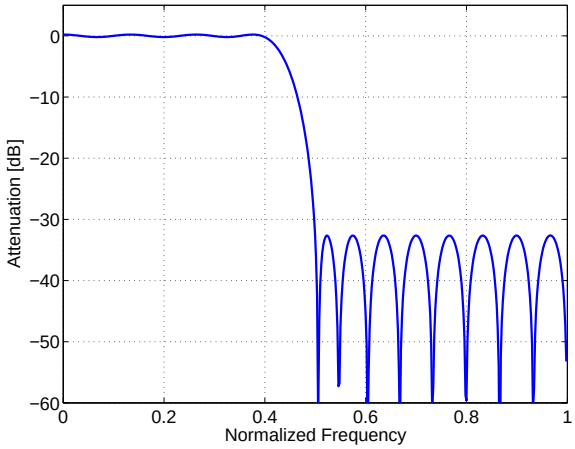

Fig. 1. The attenuation of frequency response linear phase lowpass filter of order 31.

This lowpass filter, which was designed by means of a published program based on the Parks-McClellan Algorithm is representative of a large class of FIR filters which require relatively high accuracy in the coefficients to prevent serious distortion in the frequency response.

The design and numerical computation of an FIR filter was done using MATLAB® [24] using Parks-McClellan algorithm in a two-step process. First is to use the `firpmord` command to estimate the order of the optimal Parks-McClellan FIR filter to meet design specifications. The syntax of the command is as follows: `[n,fo,mo,w]=firpmord(f,m,dev)`, where `f` is vector of band frequencies, vector `m` contains the desired magnitude response values at the passbands and the stopbands of the filter, and the vector `dev` has the maximum allowable devia-

tions of the magnitude response of the filter from the desired magnitude response. The second step is the actual design of the filter, using the `firpm` command `b=firpm(n,fo,mo)` to find the impulse response `b` of the Parks-McClellan FIR filter for our design.

A moduli set must be selected to provide just enough dynamic range for the FIR filter. Consider the 31th-order lowpass linear phase filter described by the magnitude response of Figure 1.

The filter coefficients are shown in Table 1 for double precision (the IEEE 754 standard) and for 10-bit precision in integer notation. The spectrum of the quantization error which results from quantizing coefficients to 10 bits is shown in Figure 2, where it can be seen that 10 bits is sufficient to maintain a quantization error which is 20 dB below the stopband filter response.

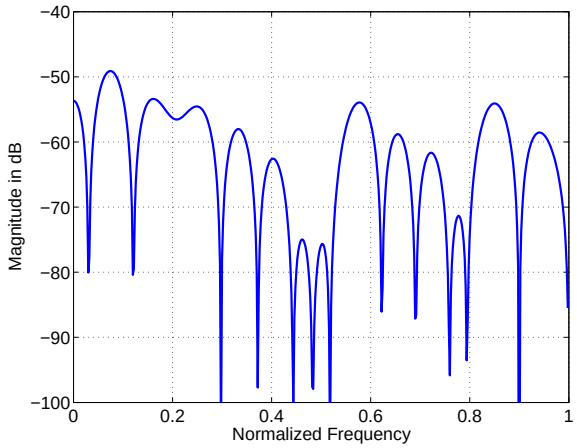

Fig. 2. Quantization error resulting from filter coefficient rounding to 10 bits including sign bit.

Integer values in the third column in Table 1 are transformed from floating point value (second column) in two steps. First step is conversion of floating point filter coefficients `b` in binary string `b_binary` using two MATLAB® functions, `Q_1=quantizer('round',Format)` and `b_binary=num2bin(Q_1,b)`. Value `Format` in `quantizer` MATLAB® function creates parameters of binary numbers: `[wordlength, fractionlength]` for signed fixed-point mode. For 10-bit precision format are `wordlength=12` and `fractionlength=10`.

Second step is conversion of binary string `b_binary` into integer value using two new MATLAB® functions: `q_1=quantizer('round',Format)` and `b_int=bin2num(q_1,b_binary)`. In this case value `Format` is without `fractionlength` i.e. `Format=[12, 0]`. At last, integer values of filter coefficients are transformed in RNS number. This paper investigates binary to residue

converter for the modulo set $\{63,64,65\}$. For example, double precision of filter coefficient b_1 is `b=-0.00039444937475` which is converted to binary number `b_binary=000000010010`, than to integer number `b_int=18`, and at last to RNS number b_RNS=$\{18, 18\ 18\}$.

Assume that the data sequence is quantized to 8 bits (including sign) and that filter must be implemented without rounding error. An absolute upper bound on $|y(n)|$ is given by (8)

$$\begin{aligned} |y(n)| \leq & \max\{|u(n)| \sum_{k=0}^{31} |b_k| \\ & = 128 \times 1874 = 239\,872 \rightarrow 17.8719 \text{ bits}. \end{aligned} \tag{8}$$

The moduli set $\{63,64,65\}$ provides a dynamic range of 17.9996 bits, which is adequate for most practical situations since the bound of 17,8719 bits given by (8) is extremely pessimistic.

Table 1. Coefficients for the 31th-order lowpass filter.

	Double precision	Integer	RNS number
$b_0 = b_{31}$	−0.00039444937475	0	{0 0 0 }
$b_1 = b_{30}$	0.01737509170472	18	{18 18 18 }
$b_2 = b_{19}$	0.00288815025876	3	{3 3 3 }
$b_3 = b_{28}$	−0.01302520660383	−13	{50 51 52 }
$b_4 = b_{27}$	−0.00883062510144	−9	{54 55 56 }
$b_5 = b_{26}$	0.01538085159507	16	{16 16 16 }
$b_6 = b_{25}$	0.01834647657814	19	{19 19 19 }
$b_7 = b_{24}$	−0.01509438918723	−15	{48 49 50 }
$b_8 = b_{23}$	−0.03209078269203	−33	{30 31 32 }
$b_9 = b_{22}$	0.00985336146129	10	{10 10 10 }
$b_{10} = b_{21}$	0.05188388112373	53	{53 53 53 }
$b_{11} = b_{20}$	0.00516582332405	5	{5 5 5 }
$b_{12} = b_{19}$	−0.08475379480646	−87	{39 41 43 }
$b_{13} = b_{18}$	−0.04768004294532	−49	{14 15 16 }
$b_{14} = b_{17}$	0.17953772215585	184	{58 56 54 }
$b_{15} = b_{16}$	0.41309620875676	423	{45 39 33 }

To produce linear phase filters, certain symmetry conditions have to be imposed on $\{b_k\}$, where $\{b_k\}$ are real filter coefficients. Consider transfer function of order N whose transfer function is

$$H(z) = b_0 + b_1 z^{-1} + \ldots + b_{30} z^{-30} + b_{31} z^{-31}. \tag{9}$$

In our paper filter order ($N = 31$) is an odd integer and suppose that $\{b_k\}$ has even

symmetry around $N/2$, that is $b_k = b_{N-k}$

$$H(z) = \sum_{k=0}^{\frac{N-1}{2}} b_k (z^k + z^{N-k}). \tag{10}$$

This structure is called the linear phase direct form.

The block diagram implementation of the transfer function (10) is shown in Figure 3 for odd N. As it can be seen from Figure 3 the basic arithmetic operation is a multiplication followed by an addition. This is usually called a multiply-accumulate (MAC) operation.

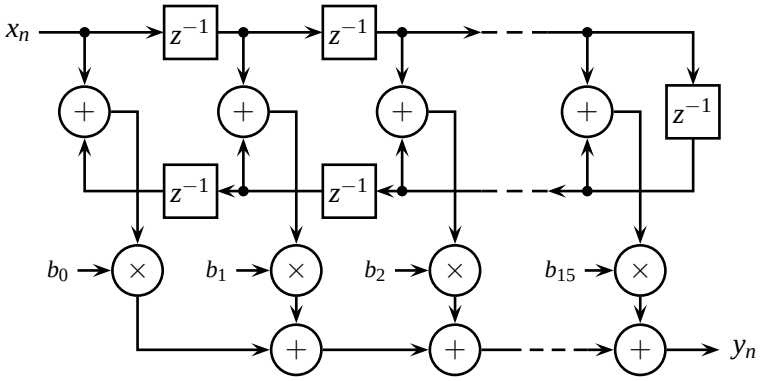

Fig. 3. Realization of linear phase FIR filter of length 32.

4 The Architecture of RNS FIR Filter

The implementation of residue number system based on the finite impulse response of linear phase filter is shown in Figure 4. As it can be noted, finite impulse response filtering is achieved in residue number system domain by using multiple modulo m_i finite impulse response filter blocks. The implementation is generic and assumes that three modulo ($m_1 = 63$, $m_2 = 64$ and $m_3 = 65$) are chosen so as to meet the desired filter precision requirements. The finite impulse response filtering is performed as a series of modulo multiply-and-accumulate (MAC) operations across each modulo m_1 to m_3. Designing and optimizing MAC operator is very important to carry out high performance and low power DSP operations. In general MAC operator is implemented using multipliers and adders.

Block Forward converter is residue-to-binary converter for three modulo set of the form $\{63, 64, 65\}$. Note that forward conversion for the modulo-64 channel is achieved simply by keeping the least significant 6 bits of the 2s complement data.

To calculate the response of FIR filter y_n to arbitrary inputs x_n modulo MAC operation across each moduli m_1, m_2 and m_3 is used. The input signal x_n would be converted into residue form at the filter input. The residue encoded output $\langle y_n \rangle_M$ would be computed in parallel residue circuit

$$\begin{aligned}
\langle y_n \rangle_{m_1} =& \langle b_0 \rangle_{m_1} [\langle x_n \rangle_{m_1} + \langle x_{n-31} \rangle_{m_1}] + \langle b_1 \rangle_{m_1} [\langle x_{n-1} \rangle_{m_1} + \langle x_{n-30} \rangle_{m_1}] + \cdots \\
& + \langle b_{15} \rangle_{m_1} [\langle x_{n-15} \rangle_{m_1} + \langle x_{n-16} \rangle_{m_1}] \\
\langle y_n \rangle_{m_2} =& \langle b_0 \rangle_{m_2} [\langle x_n \rangle_{m_2} + \langle x_{n-31} \rangle_{m_2}] + \langle b_1 \rangle_{m_2} [\langle x_{n-1} \rangle_{m_2} + \langle x_{n-30} \rangle_{m_2}] + \cdots \\
& + \langle b_{15} \rangle_{m_2} [\langle x_{n-15} \rangle_{m_2} + \langle x_{n-16} \rangle_{m_2}] \\
\langle y_n \rangle_{m_3} =& \langle b_0 \rangle_{m_3} [\langle x_n \rangle_{m_3} + \langle x_{n-31} \rangle_{m_3}] + \langle b_1 \rangle_{m_3} [\langle x_{n-1} \rangle_{m_3} + \langle x_{n-30} \rangle_{m_3}] + \cdots \\
& + \langle b_{15} \rangle_{m_3} [\langle x_{n-15} \rangle_{m_3} + \langle x_{n-16} \rangle_{m_3}]
\end{aligned} \tag{11}$$

The result $y(n)$ is obtained by the RNS to the binary conversion block by using the Chinese Remainder Theorem (CRT)

$$y_n = CRT\{\langle y_n \rangle_{m_1}, \langle y_n \rangle_{m_2}, \langle y_n \rangle_{m_3}\}. \tag{12}$$

Clearly, the input and output conversions, constitute a significant overhead in systems implemented in RNS.

Reverse conversion y_n is last step in digital signal processing in residue number arithmetic. Result is integer number `Y_int`. For comparison this results with results obtained trough standard signal processing, we have finished this example with conversion of integer number to fixed point presentation.

First step in conversion of integer number to fixed point number is conversion of integer number to binary number. Notice that the result of the two positive binary numbers $\langle b_k \rangle_{m_1} [\langle x_{n-k} \rangle_{m_1} + \langle x_{n-31+k} \rangle_{m_1}]$ multiplication may be $n+m$ digit long, where multiplicand is n digit long and multiplier is m digit long.

In our example input signal and coefficient are 7 and 10 bits long, respectively, then result is 17 digits long. Thus, format in function `Q_2=quantizer('round',Format)` is `Format=[17 0]`. Using `Y_bin=num2bin(Q_2, Y_int)` we can convert results to binary number. Finally, for binary to fixed point conversion we use following MATLAB® function `q_2=quantizer('round',[19 17])` and `Y_float=bin2num(q_2,Y_bin)`. For example, if `Y_RNS=[29, 11, 61]` then after RNS to integer conversion we obtain `Y_integer=-123829`, which is converted to binary number `Y_biary=11000011100010010 11` which yield to `Y_float= -0.94474029541016`.

At the end of every subfilter output computation, data in data memory need to be shifted so that new data sample x_{n+1} come to place x_n and data value x_n in turn

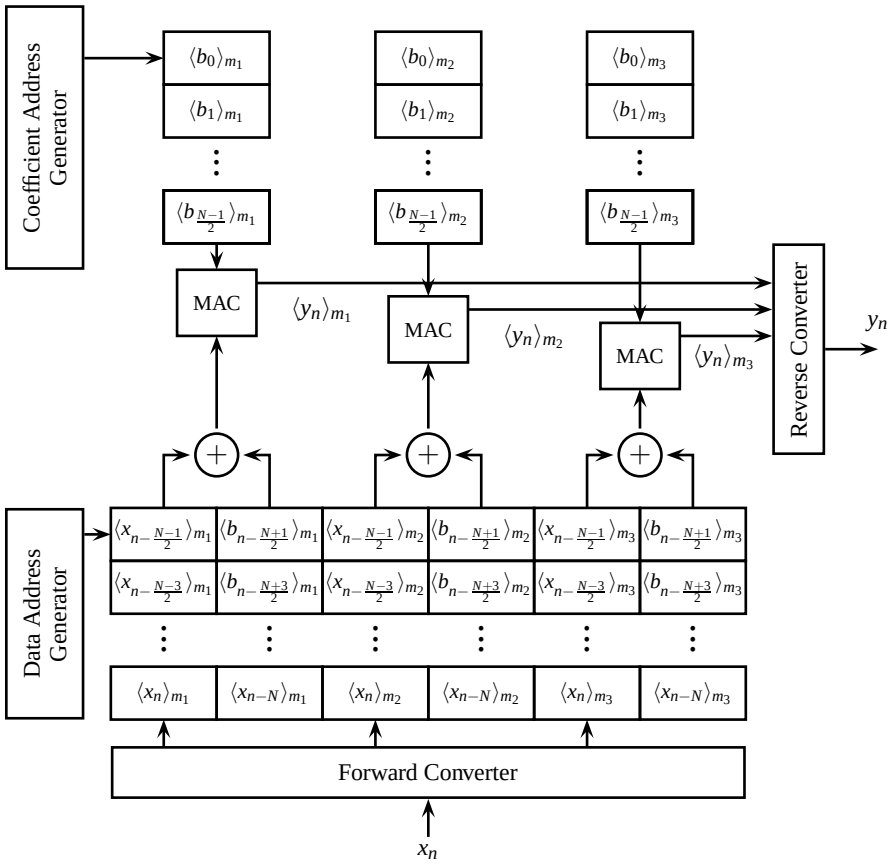

Fig. 4. Tree modulo architecture of linear phase FIR filter of length 32.

replaces the data value x_{n-1}. Due this data movement the clock rate can be maximized by modifying the data address register in order to act as a circular address generator. To implement this, a counter can be used which reset to location 0 after counting up to N. The new data sample read-in in this location and computation of next output is resumed.

If direct form linear phase FIR filter is realized so the input data are stored in one memory, while the coefficients are stored in another memory. Then each output is computed by performing $(N+1)/2$ MAC operations. Thus, this structure requires 50% less multiplications than the direct form.

5 Signal processing in residue arithmetic

Transient response is important characteristic (characterization) of a system, though it is often used as the impulse response and sinusoidal steady state response.

5.1 Impulse response

Impulse response of a discrete-time (or digital) system is defined as the output (response), denoted by $h(n)$, when the input is unit sample $\delta(n)$. In integer notation unit sample is multiplied by scalar 128

$$\delta(n) = \begin{cases} 128, & \text{if } n = 0 \\ 0, & \text{if } n \neq 0. \end{cases} \tag{13}$$

In $\{63, 64, 65\}$ residue number system unit sample is

$$\delta(n) = \begin{cases} \{2, 0, 63\}, & \text{if } n = 0 \\ \{0, 0, 0\} & \text{if } n \neq 0. \end{cases} \tag{14}$$

Figure 5 shows impulse response of RNS subfilters. The coefficient values of the first and the second subfilter must be stored in 8 bits word, but values of the third subfilter must be stored in 9 bit word.

Unit sample on the input for the first subfilter is multiplied by scalar 2, for the second subfilter is multilied by zero, and for the third subfilter is multiplied by scalar 63.

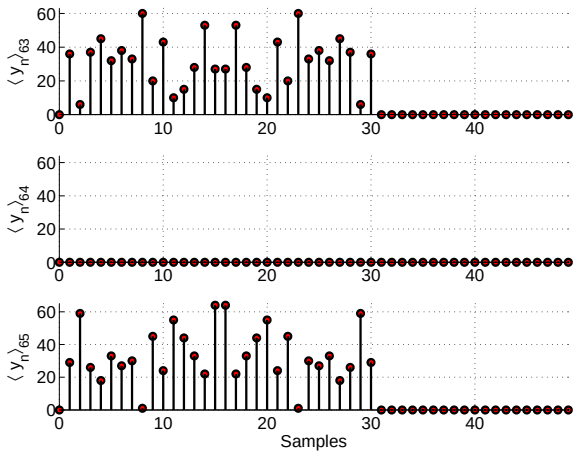

Fig. 5. Impulse response of the RNS lowpass subfilters.

For impulse response of whole filter we can use the Chinese Remainder Theorem in order to convert a number presented in the residue system into conventional number system. This impulse response is shown in Figure 6.

As it was expected, impulse response of linear phase filter (10) and impulse response on the Figure 6 are similar. The quantization error of impulse response,

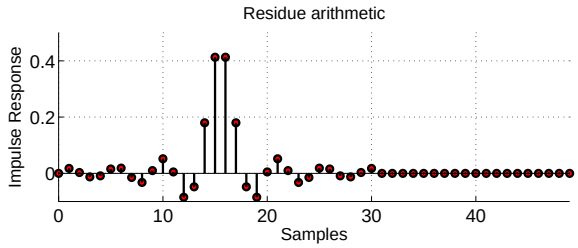

Fig. 6. Impulse respomse of the RNS lowpass filter.

resulting from quantizing the coefficients to 10 bits, is shown in Figure 7. It can be seen that 10 bits is sufficient to maintain error which is less than 4×10^{-4}.

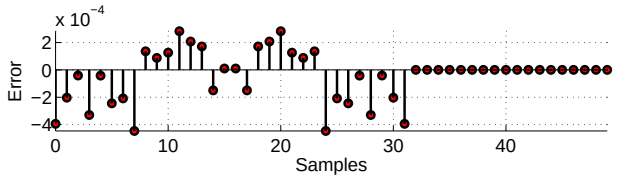

Fig. 7. Quantization error for impulse response resulting from rounding filter coefficients to 10 bits, including sign.

5.2 Steady state response

Steady state response of RNS FIR digital filter is the second example. The term 'steady state response' arise naturally in the context of sinewave analysis. In our example sampled sinewave signal with frequency $f = 1/8$ Hz is $x_k = 127\sin(2\pi k/16)$. Sampling frequency is F_s =2 Hz. Figure 8 shows steady state response of RNS subfilters.

Before signal application to the input of a digital filter, the filter's internal "state" is assumed to be equal to zero. When input sinewave is switched on, the filter takes a while to "settle down" to a perfect sinewave at the same frequency. The filter response during this "settling" period is called the transient response of the filter. The response of the linear and time-invariant filter, after the transient response, is called the steady-state response, and it consists of a pure sinewave at the same frequency as the input sinewave, but with amplitude and phase determined by the filter's frequency response at that frequency. In other words, the steady-state response begins when the LTI filter is fully "warmed up" by the input signal. More precisely, the filter output is the same as it would be if the input signal would be applied since time minus infinity.

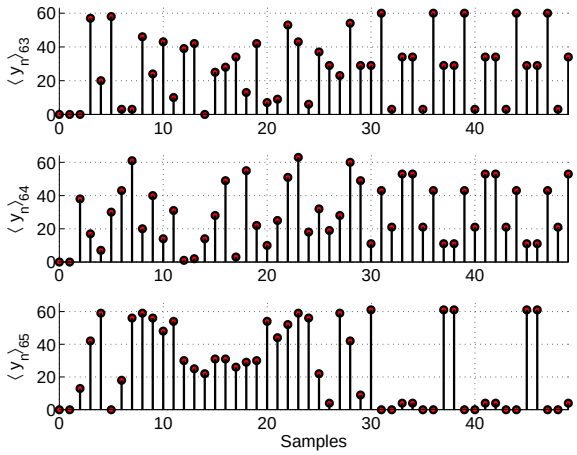

Fig. 8. Steady state response of the RNS lowpass subfilters.

Figure 9 shows steady state response of RNS FIR filter which was done using MATLAB® designed function `rns2num(([y1 y2 y3],RNS)`, where `[y1 y2 y3]` is steady response of subfilters, and `RNS=[63 64 65]` is modulo set.

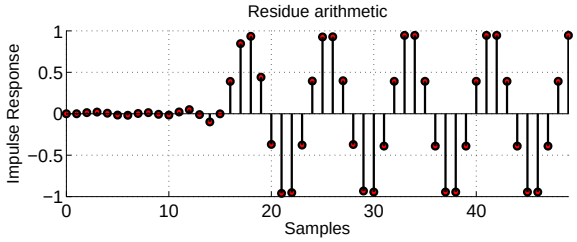

Fig. 9. Steady state response of the RNS lowpass filter.

FIR filter of length $N+1$ is fully "warmed up" after N samples of input; that is, for input starting at time $k=0$, by time $n=N$, all internal state delays of the filter contain delayed input samples instead of their initial zeros. When the input signal is a unit step $u(n)$ times sinusoid (or, by superposition, any linear combination of sinusoids), we may say that the filter output reaches steady state at time $n=N$.

In general, complete response of our filter is given by the superposition of its zero-state response and, initial-condition response. Zero-state response simply means the response of the filter to an input signal when the initial state of the filter is zeroed to begin with. The initial-condition response is of course the response of the filter to its own initial state, with the input signal being zero.

Note that, both the phase and group delay of a linear-phase filter are equal to

$N/2$ samples of plain delay at every frequency. Since FIR filter of length $N+1$ implements N samples of delay, the value $N/2$ is exactly half of the total filter delay.

As it was expected, steady state response of linear phase filter (10) and steady state response on the Figure 9 are similar. The quantization error of steady state response resulting from the coefficients quantization to 10 bits is shown in Figure 10. It can be seen that error is less than 4×10^{-3}.

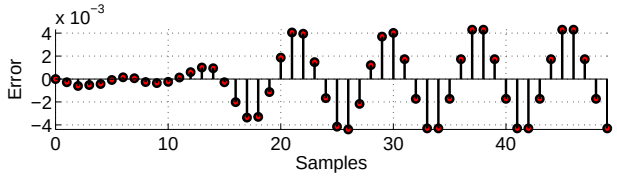

Fig. 10. Quantization error for steady state response resulting from filter coefficients rounding to 10 bits including sign.

To decrease error coefficients must be quantized to more than 10-bit precision. In this case set of modulo must be larger, for example `RNS=[127, 128, 129]`. Coefficient of subfilter for this modulo set are stored in 7-, 7-, and 8-bit word.

6 Conclusion

An residue number system finite impulse response filter architecture is presented in this paper. The RNS coding technique is attractive for FIR filters which requires only multiplication and addition because these operations are very fast in an RNS. Since the RNS implementation, in its fundamental form, produces filter outputs with full precision (no roundoff error), it is particulary attractive for real time filtering and image data, both of which coarsely quantized to minimize processing and storage requirements.

An RNS design proposed for the 31th order lowpass FIR filter can be based on standard TTL IC packages.

Acknowledgment

The author is grateful to Professor Vidosav Stojanović for comments and suggestions that improved the presentation in the paper.

References

[1] M. Soderstrand, M. A. W. Jenkis, G. Jullien, and F. Taylor, Eds., *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. Piscataway, NJ: IEEE Press, 1986.

[2] M. N. Mahesh and M. Mehndale, "Low power realization of residue number system based FIR filters," in *Thirteent International Conference on VLSI Design*, 2000, pp. 30–33.

[3] W. Freking and K. Parhi, "Low-power FIR digital filters using residue arithmetic," in *Conf. Record 31st Asil. Conf. Signals, Syst. and Comput. (ACSSC 1997)*, vol. 1, Pacific Grove, CA USA, 1997, pp. 739–743.

[4] A. D'Amora *et al.*, "Reducing power dissipation in complex digital filters by using the quadratic residue number system," in *Conf. Record 34th Asil. Conf. Signals, Syst. Comput. (ACSSC 2000)*, vol. 2, Pacific Grove, CA USA, 2000, pp. 879–883.

[5] G. Cardarilli *et al.*, "Low-power implementation of polyphase filters in quadratic residue number system," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS 2004)*, vol. 2, Vancouver, BC, Canada, 2004, pp. 725–728.

[6] W. K. Jenkins and B. Leon, "The use of residue number systems in the design of finite impulse response digital filters," *IEEE Trans. on Circuits and Systems*, vol. CAS-24, no. 4, pp. 191–201, Apr. 1997.

[7] Y. Wang, M. Aboulhamid, and H. Shen, "Adder based residue to binary numbers converters for $(2^n-1, 2^n, 2^n+1)$," *IEEE Trans. Signal Processing*, vol. 50, no. 7, pp. 1772–1779, 2002.

[8] S. J. Piestrak, "A high-speed ealization of a residue to binary number system converter," *IEEE Trans. on Circuits and System II: Analog and Digital Signal Processing*, vol. 42, no. 10, pp. 661–663, Oct. 1995.

[9] D. Gallaher, F. E. Petry, and P. Sirnivasan, "The digit parallel method for fast RNS to weighted number system conversion for specifiv moduli $(2^k-1, 2^k, 2^k+1)$," *IEEE Trans. on Circuits and System II: Analog and Digital Signal Processing*, vol. 44, no. 1, pp. 53–57, Jan. 1997.

[10] A. Hiasat and H. S. Abdel-Aty-Zohdy, "Residue-to-binary arithmetic converter for moduli set $(2^k, 2^k-1, 2^{k-1}-1)$," *IEEE Trans. Circuits and System*, vol. 45, pp. 204–208, 1998.

[11] W. Wang, M. N. S. Swamy, M. O. Ahmad, and Y. Wang, "A. high-speed residue-to-binary converter and a scheme of its VSLI implementation," *IEEE Trans. Circuits and System II: Analog and Digital Signal Processing*, vol. 47, pp. 1576–1581, 2000.

[12] P. V. A. Mohan, "Rns-to-binary converter for a new three moduli set $(2^n, 2^{2n}-1, 2^{2n}+1)$," *IEEE Trans. Circuits and System II: Analog and Digital Signal Processing*, vol. 54, no. 9, pp. 775–779, 2007.

[13] F. Pourbigharaz and H. M. Yassine, "A signed-digit architecture for residue to binary transformation," *IEEE Trans. Comput.*, vol. 46, pp. 1146–1150, 1997.

[14] A. Hariri, K. Navi, and R. Rastegar, "Range moduli set with efficient reverse converter," *International Elsevier Journal of Computers and Mathematics with Applications*, 2007, doi:10.1016/j.camwa.2007.04.028.

[15] A. S. Molahosseini and K. Navi, "New arithmetic residue to binary converters," *Int. Journal of Compter Science and Engineering Systems*, vol. 1, no. 4, pp. 291–295, Oct. 2007.

[16] K. A. Rosen, *Elementary Number Theory*. Addison-Wesley, 2000.

[17] A. Omondi and B. Prekumar, *Residue number system, Theory and implementation*. London: Imperial College Press, 2007.

[18] D. E. Knuth, *The Art of Computer Programming*, 2nd ed. Addison-Wesley Published Company, 1981, vol. 2, Seminumerical Algorithm.

[19] J. D. Lipson, *Elements of Algebraic Computing*. Addison-Wesley Published Company, 1981.

[20] N. Szabo and R. I. Tanaka, *Residue Arithmetic and its Application to Computer Technology*. New York: McGraw-Hill, 1967.

[21] G. Dimauro, S. Impedovo, and G. Pirlo, "A new techniques for fast number comparison in residue number system," *IEEE Trans. on Computer*, vol. 42, no. 5, pp. 608–612, May 1993.

[22] T. Tomczak, "Fast sign detection for rns $(2^n-1, 2^n, 2^n+1)$," *TEEE Trans. on Circuits and Systems-I: Regular papers*, vol. 55, no. 6, pp. 1502–1511, July 2008.

[23] M. Lu and J. Chiang, "A novel division algorithm for the residue number system," *IEEE Trans. on Computer*, vol. 41, no. 8, Aug. 1992.

[24] V. K. Ingle and J. K. Proakis, *Digital Signal Procesing Using MATLAB®* , ser. BookWare Companion Series. New York, Madrid: Brooks/Cole Publishing Company, 2000.