Low Power Realization of Residue Number System based FIR Filters

M. N. Mahesh, Mahesh Mehendale
Texas Instruments (INDIA) Ltd.
Golf View Homes, Wind Tunnel Road,
Bangalore - 560017, INDIA
maheshmn,mhm@india.ti.com

Abstract

In this paper, we present algorithmic and architectural transforms for low power realization of Residue Number System(RNS) based FIR filters. These transforms have been systematically derived so as to achieve power reduction by voltage scaling, switched capacitance reduction and reduction in signal activity. We show how some of the existing techniques can be suitably adopted to RNS based implementations and also propose new techniques that exploit the specific properties of RNS based computation. We present results to show the effectiveness of our techniques. The results for modulo-5 and modulo-7 indicate that using just two of these techniques(coefficient encoding and coefficient ordering), power reduction of upto 33% can be achieved.

1. Introduction

With recent trend towards portable computing and wireless communication systems, power dissipation has become an important design consideration. These systems require high speed computation, complex functionalities, real-time processing capabilities. Residue Number System(RNS) has been proposed in the literature for high speed implementation of DSP algorithms [1, 2]. RNS achieves this by breaking an operation (such as addition, multiplication etc.) into smaller operations that can be executed in parallel. In this paper, we present algorithmic and architectural transforms for low power realization of RNS based Finite Implulse Response(FIR) filters.

FIR filters are one of the most common building blocks in most of the Digital Signal Processing applications. FIR filtering is achieved by convolving the most recent N input data samples and the desired unit impulse response of the filter as shown in equation 1.

$$Y(n) = \sum_{i=0}^{N-1} A[i] * X[n-i] \quad (1)$$

The filter coefficient values($A[i]$) and the number of filter taps(N) are selected so as to satisfy the desired filter response. Many algorithmic and architectural extensions have been presented in the literature for low-power implementation of FIR filters [4, 5]. However, these techniques cannot be directly applied to the RNS based FIR filters because of the difference in implementation style. In this paper, we show how the existing techniques can be adopted for RNS based implementations and also present RNS specific techniques for low power realization of FIR filters.

The paper is organized as follows. We start with a brief introduction to RNS along with the implementation of RNS based FIR filters in section 2. We identify the sources of power dissipation and suggest different techniques for low power realization of RNS based FIR filters in section 3. The results of coefficient ordering and coefficient encoding techniques on the implementation of RNS based FIR filters are presented in section 4. Finally, we conclude in section 5 with a brief discussion on the future scope of our work.

2. Residue Number System

In RNS, an integer is represented as a set of residues with respect to a set of integers called the Moduli. Let $(m_1, m_2, m_3, ..., m_n)$ be a set of relatively prime integers called the Moduli set. An integer X can be represented as $(X = (X_1, X_2, X_3, ..., X_n))$ where $X_i = (X)\ modulo\ m_i \quad for\ i = 1, 2, .., n.$ we use notation X_i to represent $|X|_{m_i}$ the residue of X w.r.t m_i. Given the moduli set, the dynamic range(M) is given by the LCM of all the moduli [6].

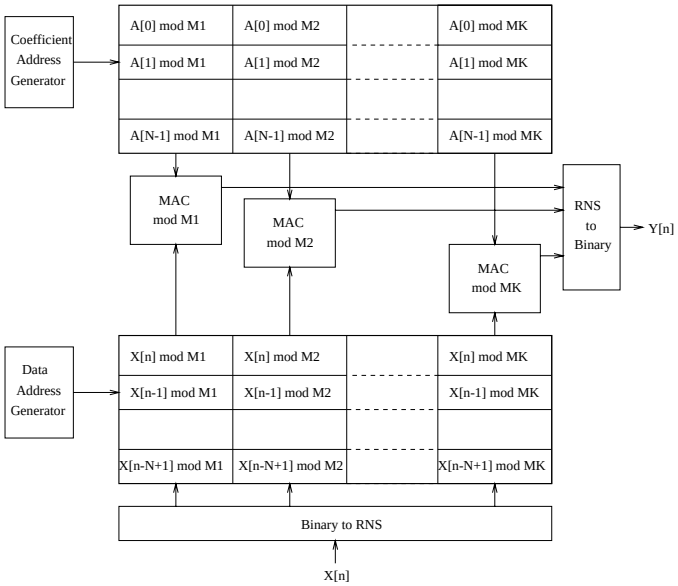

Figure 1. RNS Based FIR Filter Implementation

Let X, Y and Z have the residue representations $X = (X_1, X_2, X_3, ..., X_n)$, $Y = (Y_1, Y_2, Y_3, ..., Y_n)$ and $Z = (Z_1, Z_2, Z_3, ..., Z_n)$ respectively and $Z = (X\ op\ Y)$ where *op* is any operation in addition, multiplication or subtraction. Thus we have in RNS, $Z_i = |X_i\ op\ Y_i|_{m_i} \quad for\ i = 1, 2, .., n$.

The implementation of RNS based FIR filter is shown in Figure 1. As can be noted, FIR filtering is achieved in RNS domain by using multiple modulo m_i FIR filter blocks. The implementation is generic and assumes K moduli (M1 to MK) selected so as to meet the desired filter precision requirements. The FIR filtering is performed as a series of modulo MAC operations across each moduli M1 to MK. Figure 2 shows an implementation of a modulo MAC unit. The modulo multiplication and modulo addition is typically implemented as a look up table for small moduli. Figure 2 also shows the look up table based implementation of a modulo 3 multiplier. We now present the transformations for low power realization of RNS based FIR filter implementations.

3. Transformations for Low Power Realization

The sources of power dissipation in CMOS circuits can be classified as dynamic power, short circuit power and leakage power [7]. Dynamic power dissipation is the major source of power dissipation given by equation 2.

$$P_{dynamic} = C_{switch}.V^2.f \qquad (2)$$

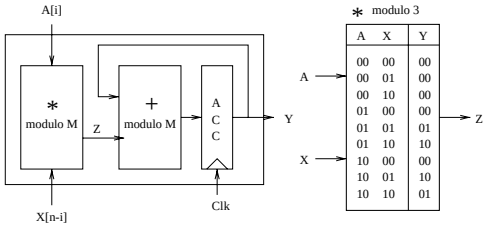

Figure 2. Modulo MAC using look up tables

where V is the supply voltage, f is the operating frequency, C_{switch} is the switching capacitance given by the product of the physical capacitance being charged/discharged and the corresponding switching probability. While power optimization can be done at different levels of abstraction [5], we focus on algorithmic and architectural level. We now identify the sources of power dissipation and propose transforms to minimize the power dissipation while maintaining the throughput.

Transform 1: Data Movement by Moving the Pointer to the Data : At the end of every output computation, the data in the data memory needs to be shifted so as to read-in the new data sample (X[n+1]) in place of X[n], the data value X[n] in turn replaces the data value X[n-1]. The power dissipation due to this data movement can be minimized by modifying the data address register to act as a circular address generator. For implementing this, a counter can be used which resets to location 0 after counting upto (N-1). The new data sample is then read-in at this location and the computation of the next output is resumed.

Transform 2: Data Flow Restructuring - Parallel Processing: The RNS implementation shown in figure 1 can be modified so as to read, for each of the moduli, two data-coefficient pairs during every cycle. The RNS structure can be modified, as shown in figure 3, so as to have two modulo MAC units per modulus. With such a parallel processing of degree two, an N tap filter can be computed in N/2 cycles. If the same throughput is to be maintained, the clock can be slowed down appropriately and the supply voltage lowered resulting in power savings.

Transform 3: Gray Coding of Coefficient Address Bus: The power dissipation in the coefficient address register and the coefficient memory can be reduced by implementing the coefficient address register as a gray counter. Since the coefficients are accessed sequentially, the gray counter can minimize the toggling in the address bus of the coefficient and data mem-

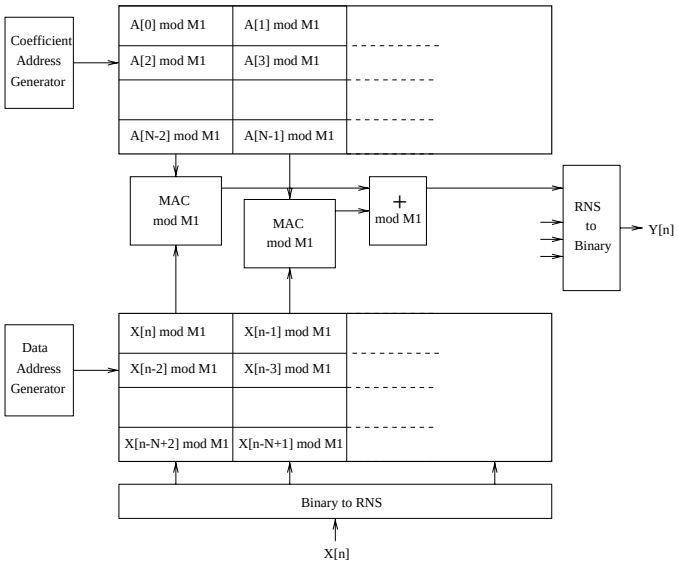

Figure 3. RNS Based Implementation of FIR Filters with Parallel Processing Transformation

ory. It has been observed that by using gray coding, as much as 50% reduction in the number of togglings can be achieved on the address bus.

Transform 4: Coefficient Encoding: The modulo multiplier of the modulo MAC unit is typically implemented as a look up table. The look up table shown in figure 2 for modulo 3 multiplier assumes that the coefficient residues are binary coded (i.e. residue 0 as 00, residue 1 as 01 and residue 2 as 10). However, since the coefficient residues are fed to the modulo MAC units only, they need not necessarily be binary coded. Secondly, since there is a separate MAC unit for each modulus, the coefficient residue coding can be different across moduli. For example, residue 0 for modulus 5 may be coded as 010 and for modulus 7 may be coded 100. This flexibility in the coefficient coding can be exploited to minimize the switching activity in the coefficient memory output busses that feed the modulo MAC units thereby reduce the power dissipation.

We now formulate the coefficient residue coding problem. For a given moduli M_i, we construct a fully connected graph with the nodes of the graph representing the residues. We then assign weights to the edges (E[i,j]) of the graph to indicate the number of times the corresponding residues (i and j) appear sequentially in the coefficient memory. The residues can then be coded so as to minimize the following cost function:

$$CF = \sum_{i} \sum_{j} HD[i,j] \cdot W[i,j] \qquad (3)$$

where $HD[i,j]$ is the Hamming distance between the coding of residues i and j, and $W[i,j]$ is the edge weight. It can be noted that the cost function CF is similar to that used for FSM state encoding. We use the stochastic evolution based optimization strategy [3] to perform the coefficient residue coding.

Transform 5: Coefficient Scaling: Scaling the output of an FIR filter preserves the filter characteristics in terms of passband ripple and stopband attenuation, and results in the magnitude gain equal to the scale factor. For a scale factor K, from equation 4 we get

$$K \cdot Y[n] = K \cdot \sum_{i=0}^{N-1} A[i] \cdot X[n-i] = \sum_{i=0}^{N-1} (K \cdot A[i]) \cdot X[n-i] \qquad (4)$$

Thus scaling the output of an FIR filter translates into scaling the coefficients by the same scale factor. Since the residue values of the scaled coefficients are different than the residue values of the original coefficients, for a given set of filter coefficients, an optimum scale factor within the specified range (e.g. $\pm$3db) can be found such that the total Hamming distance between residue values of successive coefficients, across all moduli, is minimized. Such a transformation can thus reduce the power dissipation in the coefficient memory and the modulo MAC units.

Transform 6: Coefficient Ordering: Since modulo addition is both commutative and associative, the order in which the coefficient-data pairs are fed to the modulo MAC units can be changed without impacting the filter functionality. The residues in the coefficient memory can be re-ordered so as to minimize the total Hamming distance between successive values of the coefficient residues. Such a minimization reduces the switching activity in the coefficient memory output busses and hence reduces power dissipation in the coefficient memory and the modulo MAC units.

Transform 7: Coefficient Optimization: Coefficient optimization to minimize the hamming distance between the successive coefficients has been presented in [8]. This algorithm can be adapted to minimize the total hamming distance between residues of successive coefficient values across all moduli. Coefficients can thus be optimized so as to minimize the switching activity in the coefficient memory output busses and hence reduce power dissipation in the coefficient memory and the modulo MAC units.

4. Results

The results presented in this section show the improvement in power dissipation for two of the transformations, the coefficient encoding and the coefficient

Table 1. Current Through Vdd in Modulo-7 MAC Implementation

Capacitance	Conventional(14)		Coeff_encod(12)		Coeff_ordered(4)	
	I_MULT	I_TOTAL	I_MULT	I_TOTAL	I_MULT	I_TOTAL
10 fF	30.78	151.26	18.67	130.49	13.50	105.87
30 fF	26.92	153.12	18.58	129.13	12.78	105.73
50 fF	26.02	149.75	18.99	132.74	11.91	104.38
70 fF	24.74	151.00	20.03	136.36	11.81	105.78

Table 2. Current Through Vdd in Modulo-5 MAC Implementation

Capacitance	Conventional(14)		Coeff_encod(6)		Coeff_ordered(4)	
	I_MULT	I_TOTAL	I_MULT	I_TOTAL	I_MULT	I_TOTAL
10 fF	14.75	89.46	4.76	72.25	5.62	60.20
30 fF	14.16	92.24	4.49	72.37	5.59	60.58
50 fF	13.38	93.84	4.37	73.32	4.81	59.33
70 fF	12.33	92.82	4.51	74.21	5.03	61.47

I_MULT -- Current through Vdd for multiplier block
I_TOTAL -- Current through Vdd for (modulo MAC + coeff. memory)
All currents expressed in uA.
* -- No. of togglings on the coeffcient data bus for each output of FIR filter.

ordering techniques. These techniques have been implemented on modulo m_i MAC structure shown in figure 2 along with a coefficient memory feeding the modulo multiplier. The data is assumed to be available to the modulo multiplier. The coefficient memory, modulo multiplier and the modulo adder have been implemented as PLAs and modeled in SPICE. The PLAs are minimized for area using espresso [9]. In case of coefficient ordering technique, the data values need to be ordered in the same fashion as coefficients. This can be done by using a simple switching matrix. We assume that the power dissipation due to this switching matrix is negligible and hence not considered in the results.

To evaluate the efficiency of the techniques, we consider the implementation of modulo-5 and modulo-7 MAC structures using the coefficients of a 16-tap low pass filter.The power dissipation values along with the number of togglings which account for power reduction are shown in Tables 1 and 2 for different techniques. The implementation has been done for varying capacitances on the coefficient memory data bus and the power dissipation is expressed in terms of the current flowing through Vdd. As can be seen from the results, as much as 33% reduction in total power dissipation has been achieved using these techniques. It is to be noted that in coefficient encoding technique, the area of the modulo multiplier block is dependent on the residue encodings. This has an effect on the capacitative loading of the bus lines thereby affecting the power dissipation.

5. Conclusion and Future work

In this paper, we have presented algorithmic and architectural transforms for low power realization of RNS based FIR filters. It is to be noted that while most of the transforms are generic, the coefficient encoding technique is RNS specific optimization. The coefficient encoding and coefficient ordering techniques result in upto 33% reduction in power dissipation. As pointed out earlier, the encoding of coefficients effects the area of the modulo multiplier which accounts for the capacitative loading of the bus lines. So, we need to come up with an encoding which optimally reduces both modulo multiplier area and also the togglings on the coefficient memory data bus.

An alternative form of modulo m_i FIR filter structure has been proposed in the literature for higher order FIR filters [10]. Because of a different structure, some of the transforms discussed in this paper are not applicable. The algorithm in the coefficient optimization technique can be modified so that number of unique residues across the moduli set is reduced, thereby reducing the number of modulo multipliers needed for a specific modulo. This directly translates to reduction in power dissipation because of the elimination of some computational elements.

References

[1] W. K. Jenkins, "A highly efficient residue combinatorial architecture for digital filters," Proc. of IEEE, Vol. 66, pp. 700-702, June 1978.

[2] W. K. Jenkins and B. Leon, "The use of Residue Number System in the design of finite impulse response digital filters," IEEE Trans. on Circuits and Systems, Vol: CAS-24, No.4, pp: 191-201, Apr. 1977.

[3] Mahesh Mehendale, B. Mitra, "An Integrated Approach to State Assignment and Sequential Element Selection for FSM Synthesis", 7th International Conference on VLSI Design, 1994, pp 369-372

[4] M. Mehendale, S. D. Sherlekar, G. Venkatesh, "Algorithmic and Architectural Transformations for Low Power Realization of FIR Filters", Intl. Conference on VLSI Design, VLSI Design'98, pp 12-17.

[5] Jan Rabaey and M. Pedram, *Low Power Design Methodologies* Kluwer Academic Publishers, 1995.

[6] N. S. Szabo and R. I. Tanaka, *Residue arithmetic and its Applications to computer technology.* New York: McGraw-Hill,1967.

[7] A. P. Chandrakasan and R. W. Brodersen, "Minimizing Power Consumption in Digital CMOS Circuits", Proceedings of the IEEE, April 1995, pp 498-523.

[8] M. Mehendale, S. D. Sherlekar, G. Venkatesh, "Coefficient Optimization for Low Power Realization of FIR Filters", IEEE workshop on VLSI Signal Proc., 1995

[9] G.De Micheli, *Synthesis and optimization of digital circuits.* McGraw-Hill, 1994.

[10] M. A. Soderstrand and Kamal Al-Marayati, " VLSI implementation of very-high-order FIR filters", Proc. of ISCAS, pp: 1436-1439, 1995.