IJCSC International Journal of Computer Science & Communication Vol. 1, No. 2, July-December 2010, pp. 175-179

A Novel Architecture of "Mixed Number System MAC Unit" for Digital Signal Processors

Aniruddha Ghosh[1], Satrughna Singha[2] & Amitabha Sinha[3]

[1]Department of Computer Sc. and Engg., Pailan College of Management & Technology, Kolkata, India.
[2]Department of Computer Science and Engineering, JIS College of Engineering, Kalyani, W. B., India.
[3]Director, School of Engineering & Technology, West Bengal University of Technology, Kolkata, India.
Email: g_aniruddha2003@yahoo.co.in[1], satrughna.singha@gmail.com[2], amitabha_sinha@iecee.org[3]

ABSTRACT

Execution of arithmetic operations at a very high speed in real time is the major concern in digital signal processing (DSP) because DSP algorithms are computation intensive. In recent times, Residue Number Systems (RNS) are considered as alternative to binary number system because of their capabilities of performing "carry-free" addition based on proper selection of the modulii set. Double Base Number Systems (DBNS) is another efficient number system. Double Base Number Systems (DBNS) are increasingly becoming attractive for signal processing applications due to their capabilities of handling arithmetic operations, particularly multiplication efficiently. However, the complexity involved in converting binary to DBNS becomes a major bottleneck and the efficiency of performance decreases considerably due to large conversion time. So RNS Adder and DBNS Multiplier can be used to implement multiply & accumulate (MAC) units. MAC units are the key units in Digital Signal Processors. In this paper we have shown how FIR filter can be implemented using the proposed "Mixed Number System MAC units".

Keywords: Residue Number System (RNS), Modulii, Double Base Number Systems (DBNS), Multiply and Accumulate Unit (MAC), DSP Algorithms.

1. INTRODUCTION

DSP algorithms are computationally intensive and frequently used arithmetic operations in most of these algorithms are sum-of-products. For implementing an N tap FIR filter or N point FFT where N is large, time complexity and silicon area complexity for implementing the hardware become very large. Hence, designing high performance adders and multipliers are the major concern for implementing real-time signal processing systems especially for the high frequency signals when the sampling rate is very high. To improve the performance of adders and subtractors, a number of well known schemes have been proposed in some publications [9]. In recent times Residue Number Systems (RNS) [1] are becoming popular due their capabilities of performing carry free addition respectively. This improvement in speed is achieved due to concurrent arithmetic operations on the modulii. Another number system "Double Base Number Systems (DBNS)" [2] [14] have been also reported recently and are becoming attractive for their capabilities of performing multiplication operations efficiently. High quality sound systems, adaptive echo cancellation, solar signal processing, military applications etc where in addition to high speed, high precisions are also required. Keeping this issue in view, in this paper, we propose a new architecture for efficient implementation of MAC units using RNS Adder and DBNS Multiplier. In this paper we have shown how FIR filter can be implemented using the proposed "Mixed Number System MAC units". Performance analysis on simulation results clearly indicates the novelty of the architecture. The architecture was validated on Xilinx Virtex4 FPGA (Xilinx, 2004).

2. DOUBLE BASE NUMBER SYSTEM

2.1. Review of DBNS Theory and Analysis

The Double Base Number System (DBNS) [2][3][5] is a special way of representing integers as a sum of mixed powers of two (2) and three (3) which is known as two integers. This number representation scheme is unusually sparse, which is a good measure for potential applications. In the Double-Base number system, we represent integers in the following form:

$$x = \sum_{i,j} d_{i,j} 2^i 3^j ,$$

where $d_{i,j} = \{0, 1\}$. Clearly, the binary number system is a special case of the above representation. Form the expression it is clear that a given binary number when converted into DBNS system can be represented as a number of (i, j) pairs. These are also referred to as DBNS indices [1][11][13]. Fig.1 depicts a DBNS table where i and j both range from 0 to 3 [14].

The problem of finding the canonic DBNS representation of a given integer is a difficult problem. V. Dimitrov and G. A. Jullien [1] proposed a greedy algorithm [15] which provides the so-called near-canonic double-base number representation (NCDBNR) [8]. Since each iteration finds one index, the number of iterations indicates the number of ones (1s) in the DBNS table which are often referred to as active cells.

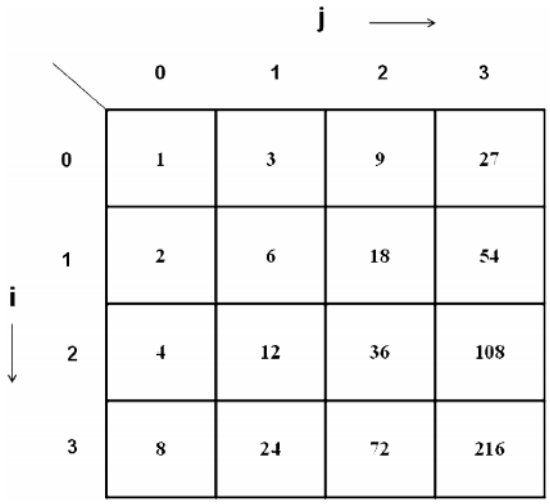


Fig. 1: DBNS Table for *i* and *j* Ranging from 0 to 3

The values given in each box in the DBNS table indicate the weight for the corresponding active cell. The maximum decimal number which is represented by a DBNS system with *m*n* cells can be obtained by adding the weighs of all the *m*n* cells.

2. 2. Review of Residue Number Systems

The residue number system (RNS) is a non-weighted number system [12]. RNS representation [1][12] of a number takes the form of N tuple e.g. $X = (x_1, x_2, x_3,...,x_N)$. Here $x_i = X \ modulo \ m_i$, represents the i-th residue digit, m_i is the i-th modulus and all m_i are mutually prime numbers. Dynamic range M as indicated below represents maximum number of distinct values that can be represented:

$$M = \prod_{i=0}^{N} m_i \text{ and } X < M$$

For signed RNS, any integer in the range of $(-M/2, M/2]$, has a unique RNS N tuple representation where $x_i = (X \ modulo \ m_i)$ if $X > 0$, and $x_i = (M - |X|) \bmod m_i$ if $X < 0$.

2.3. Arithmetic Operations

Suppose, there are two numbers A and B and after performing arithmetic operations [1][12], the generated result is denoted by C.

Now $(A) \ modulo \ (m_1 | m_2 | m_3) = (a_1 | a_2 | a_3)$

$(B) \ modulo \ (m_1 | m_2 | m_3) = (b_1 | b_2 | b_3)$

$(C) \ modulo \ (m_1 | m_2 | m_3) = (c_1 | c_2 | c_3)$

O represents arithmetic operations ie Addition, Subtraction, Multiplication.

Now, $(A \ O \ B) = C$

This implies,

$$((a_1 \ o \ b_1) \ modulo \ m_1 | (a_2 o \ b_2) \ modulo \ m_2 | (a_3 \ o \ b_3) \ modulo \ m_3)$$
$$= (C) \ modulo \ (m_1 | m_2 | m_3)$$
$$= (c_1 | c_2 | c_3)$$

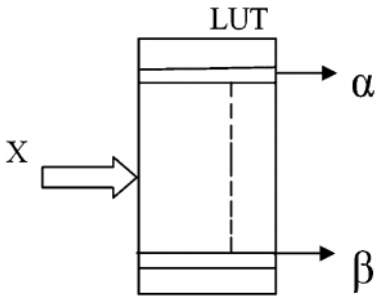
3. ARCHITECTURE OF PROPOSED MAC UNIT

The following modules are required for implementing proposed MAC unit:

1. DBNS Conversion Unit.
2. DBNS Multiplier Unit.
3. RNS Conversion Unit.
4. RNS Adder Unit.
5. RNS to Binary Conversion Unit.

3.1. DBNS Conversion Unit

Integer to DBNS conversion[2] is performed by this unit. For implementing this unit, LUT based approach is one of the efficient techniques. In DBNS, there are two bases, one is 2 and another is 3.



X is an integer $X = 2^{\alpha}.3^{\beta}$

Fig. 2: DBNS Conversion

Suppose, 'X' is an integer. This number can be represented in DBNS. Then,

$X = 2^{\alpha}.3^{\beta}$, where, α, β are two integer.

Here, the values of α and β will be stored in different location of LUT.

3.2. DBNS Multiplier Unit

Suppose, X_1 and X_2 are two integers. In DBNS representation[15],

$X_1 = 2^{\alpha}{}_1 . 3^{\beta}{}_1$ and $X_2 = 2^{\alpha}{}_2 . 3^{\beta}{}_2$.

Now, $X_1.X_2 = \mathbf{Z}$ (say).

Then, $Z = 2^{\alpha}{}_1{}^{+\alpha}{}_2 . 3^{\beta}{}_1{}^{+\beta}{}_2$.

Z_{bin} is binary equivalent of Z.

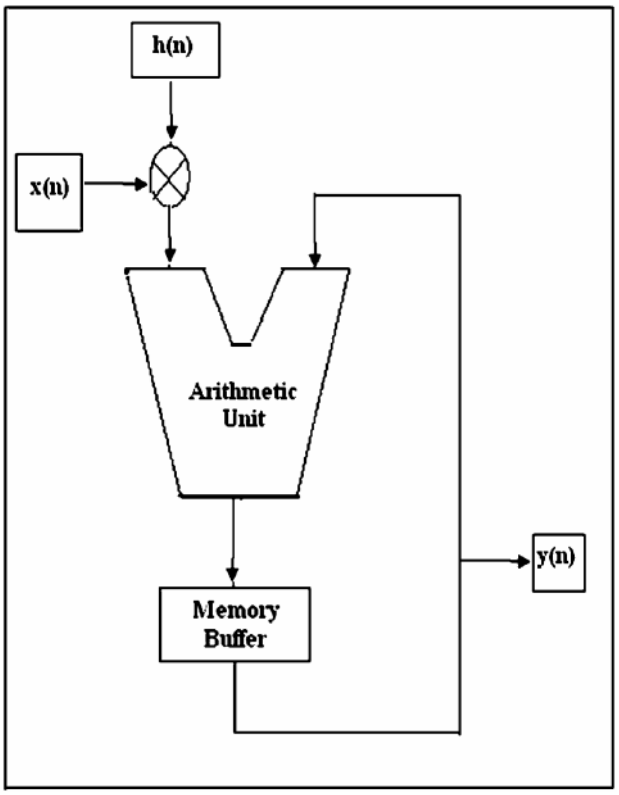


Fig.3: DBNS Multiplier Unit

For implementing DBNS Multiplier unit (shown in Fig.3), Adder, LUT, Shifter (Barrel Shifter) is required. There are two adders. Addition operation over indices of 2 is performed by one adder and Addition operation over indices of 3 is performed by another adder. The binary equivalent of $3^{\beta}{}_1{}^{+\beta}{}_2$ is stored in the LUT. This stored data is passed through a shifter. Instead of using conventional shifter, Barrel Shifter is used because of its capability of performing single cycle shifting operation. How many number of bits has to shift is decided by (α_1+ α_2). Multiplied result can be retrieved from shifter output.

3.3. RNS Adder Unit

RNS Adder (Fig.4) performs addition operation in residue domain. It has some capability of performing "Carry-free" addition operation. Residue Number System [1] performs addition, subtraction, and multiplication as parallel operations.

Suppose $c_i = (a_i + b_i)$ moduli m_i where $0 < a_i, b_i \leq m_i - 1$.

Now, $c_i = a_i + b_i$ if $a_i + b_i < m_i$;

$= a_i + bi - m_i$ if $a_i + b_i \geq m_i$;

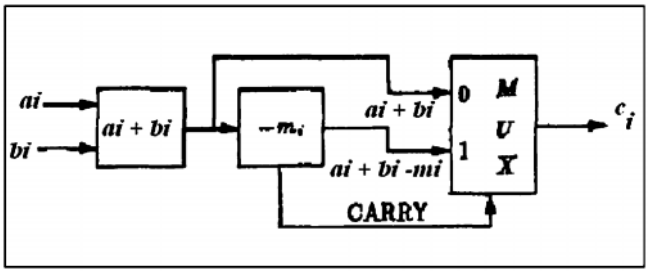


Fig.4: Residue Addition

Here two M bit adders are used. One adder is used to compute $a_i + b_i$ while another computes $a_i + b_i - m_i$. The carry, generated from second adder is used as select line of a multiplexer (Figure 3.4).

3.4. RNS Conversion Unit

This unit is implemented by lookup table. The table size grows exponentially with the number of bits required to represent the maximum moduli. The wire delay does not affect the performance. The prime numbers constitute the best modulii sets [4][16] Arbitrary selection of unbalanced modulii sets leads to inefficient architectures wherein the largest modulus is excessively dominant with respect to both cost and performance. An example of a modulii set with good balance is $\{2^n - 1, 2^n, 2^n + 1\}$.

4. PRINCIPLE OF OPERATION

Multiply-accumulate operation can be performed in single cycle by MAC unit. Suppose, $h(n)$ & $x(n)$ are two input sequence. In conventional MAC[6][7] unit (Fig.5) the inputs are multiplied & added with zero which is initially stored in the memory. The sum is then stored in the memory unit. In the next clock, the next inputs are multiplied and added with previous data stored in the memory.

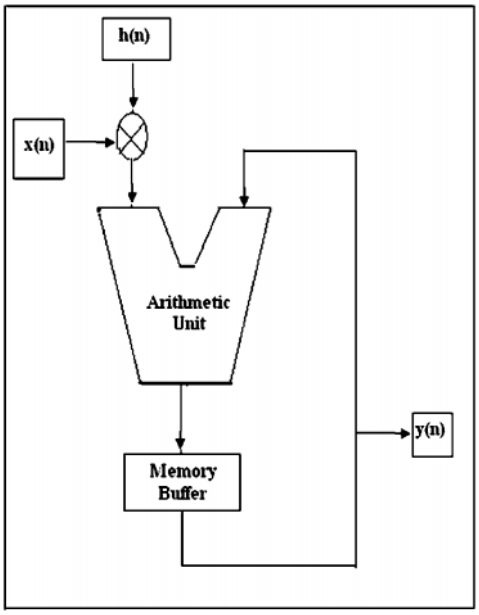


Fig. 5: Conventional MAC Unit

The architecture of the proposed MAC unit is shown in figure 6. In the proposed architecture $h(n)$ and $x(n)$ are the two inputs of DBNS conversion unit. Suppose, after conversion, $h(n) = 2^{\alpha}_{1} . 3^{\beta}_{1}$ and $x(n) = 2^{\alpha}_{2} . 3^{\beta}_{2}$. The indices of 2 and 3 are the inputs of DBNS multiplier unit. After performing multiplication operation in DBNS, generated output is in binary number system (Z_{bin}).

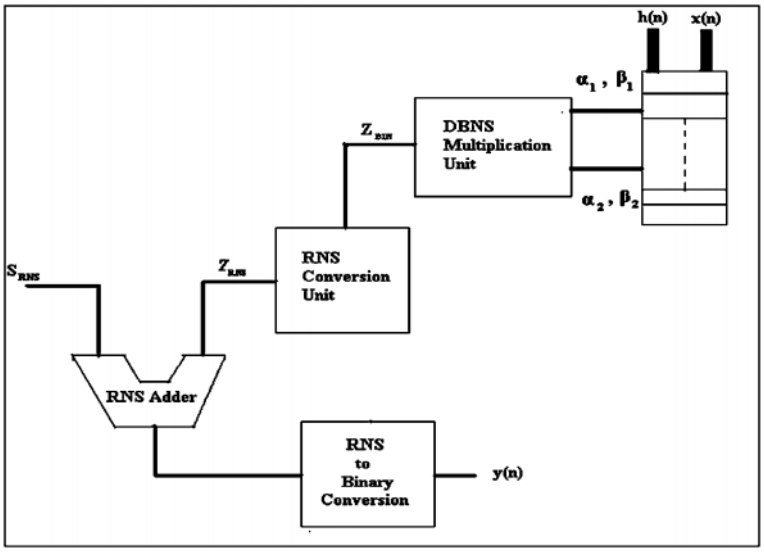


Fig.6: Proposed MAC Unit

This result is passed through binary to RNS conversion unit and Z_{bin} is converted to RNS, Z_{RNS}. This Z_{RNS} is added with previous data, $S_{RNS,}$ (S_{RNS} also in RNS domain) in RNS adder and output is passed through RNS to binary conversion unit to retrieve the final output.

4.1. FIR Filter Algorithm Analysis

If 'M' be the length of FIR Filter[10] and '$x(n)$' be the input sequence and '$y(n)$' be the output sequence. FIR Filters only use the current and past input sample to obtain output value.

Now, $y(n) = \Sigma h(k) * x(n - k)$ for all $k = 0$ to $M - 1$;

$= h(0)x(n) + h(1)x(n - 1) + h(2)x(n - 2) + h(3)\ x\ (n - 3) +\ldots\ldots\ldots + h(M - 1)\ x\ (n - (M - 1))$;

$h(k) \rightarrow$ The filter Coefficients

FIR Filter can be implemented by using systolic array. The structure of systolic array is shown in Fig.7.

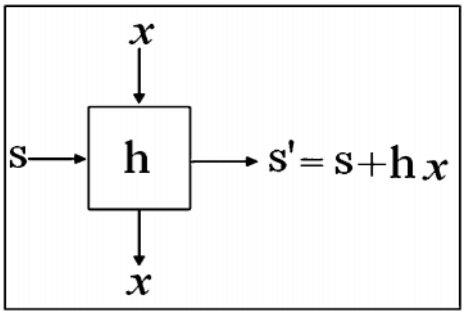


Fig. 7: Systolic Array Cell Structure

Systolic arrays are arrays of data processing units which are connected to a small number of nearest neighbour data processing units in a mesh-like topology. Data processing units perform a sequence of operations on data that flows between them.

The structure of systolic array is similar to MAC unit. Sequential execution on one MAC requires too much time to execute an operation.

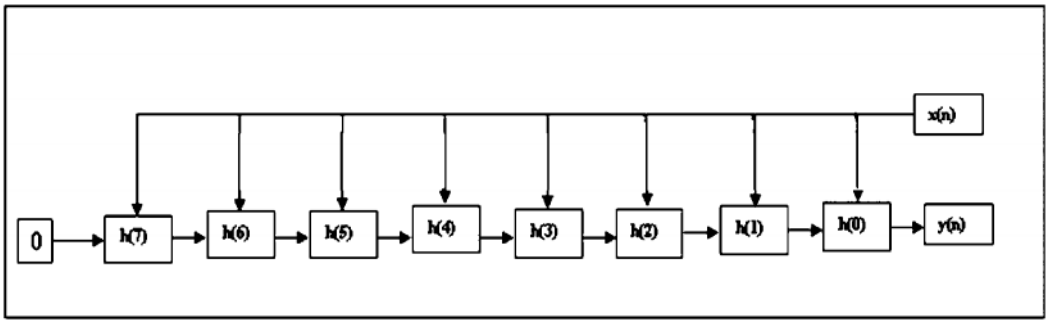


Fig.8: 8-tap FIR Filter Using Systolic Array

In systolic array, parallel execution is possible so execution time of an operation can be minimized. So, instead of using one MAC unit if FIR Filter is implemented using systolic array, it will be much more efficient with respect to execution time. Implementation of FIR Filter, using systolic array is shown in Fig.8. In Fig.8, $x(n)$ is FIR Filter input; $h(7)$, $h(6)$, $h(5)$, $h(4)$……are filter coefficients; $y(n)$ is output.

6. CONCLUSION

In this paper, a new architecture has been proposed for implementing a MAC unit. This proposed MAC unit is implemented using two different number systems. In the proposed MAC unit, addition operation is performed by RNS Adder and multiplication operation is performed by DBNS Adder. Since RNS arithmetic is inherently carry-free [1][14], addition and multiplication operations can be done on a number of small residues concurrently. Hence the need of partial products is eliminated for multiplication operations, thereby increasing the speed to a large extend. DBNS [2] can perform multiplication operations efficiently. The architecture can be validated on Xilinx Virtex IV FPGA and the detailed analysis and studies of different modules of the proposed units can be simulated using Xilinx ISE version 9.1i (Xilinx, 2004). A detailed study on performance improvements on other DSP algorithms[10] like speech processing, high quality sound systems, adaptive echo cancellation, solar signal processing, military applications is required to be done where in addition to high speed, high precisions are also required.

REFERENCES

[1] Chao-Lin Chiang and J. Lennart, "Residue Arithmatic and VLSI", *IEEE Internal Conference on Computer Design: VLSI in Computers (ICCCD'83)*, New York, Oct. 31 - Nov 3, 1983.

[2] V. S. Dimitrov, G. A. Jullien, W. C. Miller, "Theory and Applications of the Double-Base Number System", *IEEE Trans. Computers*, **48**, 10, pp.1098-1106, 1999.

[3] V. S. Dimitrov, S. Sadeghi-Emamchaie, G. A. Jullien, W. C. Miller, "A Near Canonic Double-Based Number System (DBNS) with Applications in Digital Signal Processing", *Proceedings SPIE Conference on Advanced Signal Processing*, August, 1996.

[4] Eep Setiaarif, Pepe Siy, "A New Modulii Set Selection Technique To Improve Sign Detection And Number Comparison In Residue Number System (RNS)", *NAFIPS 2005 - 2005 Annual Meeting of the North American Fuzzy Information Processing Society, IEEE Press*, New York, pages 766 - 768, 2005.

[5] J. Eskritt, R. Muscedere, G. A. Jullien, V. S. Dimitrov, W. C. Miller, "A 2-Digit DBNS Filter Architecture", *Proceedings SiPS Workshop (Lafayette, L A)*, October, 2000.

[6] J. P. Hayes, "Computer Organization", (3rd Edition), McGraw-Hill, 1998.

[7] K. Hwang, and F. A. Briggs, "Computer Architecure and Parallel Processing", (International Edition), McGraw-Hill, 1985.

[8] G. A Jullien, V. S. Dimitrov, B. Li, W. C Miller, A. Lee, M. Ahmadi, "A Hybrid DBNS Processor for DSP Computation", *Proceedings International Symposium on Circuits and Systems*, 1999.

[9] B. G. Lee, "A New Algorithm to Compute the Discrete Cosine Transforms", *IEEE Trans on Acoustics, Speech and Signal Processing*, **ASSP-32**, pp.1243- 1245, December, 1984.

[10] S. K. Mitra, "Digital Signal Processing", TATA McGRAW-Hill Edition, 2001.

[11] R. Muscedere, V. S. Dimitrov, G. A. Jullien, W. C. Miller, M. Ahmadi, "On Efficient Techniques for Difficult Operations in One and Two-digit DBNS Index Calculus", *Proceedings 34th Asilomar Conference on Signals, Systems and Computers*, November, 2000.

[12] F. J. Taylor, "Residue Arithmetic: A Tutorial with Examples", *IEEE Trans. on Computer*, pp. 50-62, May 1984.

[13] R. Tessier and W. Burleson, "Reconfigurable Computing for Digital Signal Processing: A Survey", *Journal of VLSI Signal Processing*, **28**, No.7-27, pp 7-27, 2001.

[14] Satrughna Singha and Amitabha Sinha, "Survey of Various Number Systems and Their Applications", *International Journal of Computer Science and Communication*, **1**, Number. 1, pp. 73-76, Serials Publications, Kurukshetra University, Haryana, India, January 2010.

[15] A. Sinha, P. Sinha, K. Newton, K. Mukherjee, "Multi based Number Systems for Performance Enhancement of Digital Signal Processors", Filed for U.S. Patent. (U.S. Pat. Appl. No. 11/488, 138), Published in U.S. Patent Documents Serial no.488138, U.S. Class at Publication 708/620, int'l Class : G06F 7/52 20060101 G06F007/52, 2006.

[16] Wei Wang, M.N.S Swamy, and M.O. Ahmad, "Modulii Selection in RNS for Efficient VLSI Implementation", *IEEE Press*, New York, Pages. IV-512 ~ 515, May, 2003.