RESIDUE ADDER BASED HIGH SPEED 8-BIT VEDIC MULTIPLIER

Ayush Tayal[1], Neeta Pandey[2], Rajeshwari Pandey[3]
[1, 2, 3] Department of Electronics & Communication Engineering, DTU, Delhi (India)

ABSTRACT

This paper aims at incorporating residue adder in the design of multiplier based on Nikhilam Sutra of Vedic mathematics. The residue adder employs residue number system (RNS). The performance is the proposition is compared with a Vedic multiplier employing carry save adder. The multiplier is coded in Very High Speed Integrated Circuits (VHSIC) Hardware Description Language (VHDL) while the simulation and synthesis is done using the Xilinx ISE Design Suite 14.2 software.

Keywords: Nikhilam Sutra, Residue Adder, Residue Number System (RNS), Vedic Mathematics, VHDL

I. INTRODUCTION

The use of multipliers is inevitable in any of the embedded system applications and they find great use in digital signal processing and high speed general purpose processor design. So, an important task is to design a multiplier which is efficient enough to help a designer design the final product with good performance parameters. VLSI deals with these methodologies and basically, there are three important parameters – power dissipation, chip area and delay. Due to the evolution and rapidly increasing use of the battery-based mobile embedded technology, design emphasis is now laid on optimizing the conventional delays and power dissipation while still maintaining high performance. To achieve this, different architectures and designs are explored. Some of the architectures of multipliers include array multiplier and Booth multiplier [1]. Booth multiplier is a way to perform multiplication of binary numbers in signed 2's complement notation. A modified Booth architecture has been proposed in [2] with improved performance. But the quest to find an optimized solution brought in the use of the ancient Vedic sutras. There are 16 different sutras in Vedic mathematics out of which two of them are used to design efficient multipliers. The performance of Urdhva Tiryagbhyam sutra of Vedic and the Booth multiplier architectures are compared in [3] in terms of the area, delay and power. In [4], a simple and basic Vedic multiplier is simulated while in others [5-8], the two sutras, Urdhva Tiryagbhyam Sutra and Nikhilam Sutra, are employed to design various 4-bit and 8-bit multipliers. Gunakasamuchyah Sutra is investigated in [9] for 4-bit Vedic multiplier implementation. The effectiveness of Vedic multipliers is demonstrated for convolution [10] and AES algorithm [11] which are used for digital signal processing and cryptography applications respectively. The adder is an integral part of all multipliers and influences the delay. The efficiency of the Vedic multiplier has also pronounced dependence on adder. Ripple Carry adders are used in [5-7] while [8] use Carry save adder in its architecture. Carry save adders have an obvious advantage of delaying the addition of the carry of different stages to next stages, thereby optimizing the delay. In [12-13], Pradhan and Panda have also worked on multipliers based on the Nikhilam Sutra and Urdhva Tiryagbhyam Sutra using the carry save adder. It was found that the speed of 16-bit multiplier based on Nikhilam Sutra is better than Urdhva Tiryagbhyam Sutra when magnitude of both operands are more than half of their maximum values. Using this result, they have designed a Vedic multiplier based on Nikhilam Sutra [13] using the Carry Save adder. The speed of the multiplier can be further improved by residue adder which is based on Residue number system (RNS) [14]. This system removes the carry propagation and provides an efficient way to perform calculations faster. Vedic multipliers have been designed based on CSA but to the best of author's knowledge, residue adders have not been investigated in Vedic multipliers. In this paper, residue adder based 8-bit Vedic multiplier design is put forward . The performance is compared with Vedic multiplier having a carry save adder. The paper has been formulated in six sections: Section II and III present the concepts of Vedic mathematics and Residue Number System (RNS). The proposed multiplier has been presented inSection IV. Section V shows the results while the final conclusions are drawn in the last section.

II. VEDIC MATHEMATICS

Vedic Mathematics is the ancient system of mathematics which was rediscovered from the Vedas between 1911 and 1918 by Sri Bharati Krishna Tirthaji Maharaja (1884-1960). His research presented 16 Sutras & 16 Up-sutras from Atharva Veda. These sutras are word formulae which present an easy way to perform mathematical calculations [5]. Nikhilam Navatashcaramam Dashatah Sutra has been used to design the multiplier in this paper. A single line meaning of this sutra is "All from 9 and the last from 10". The basic description of Nikhilam sutra is as follows. The two numbers to be multiplied are complemented by subtracting them from a base. The numbers and their complements are written down vertically side by side. The right hand side of the result is obtained by multiplying the two complements while the left part is calculated by subtracting any of the diagonal numbers i.e. subtracting one's complement from the other number. The example below clearly depicts the above described sutra.

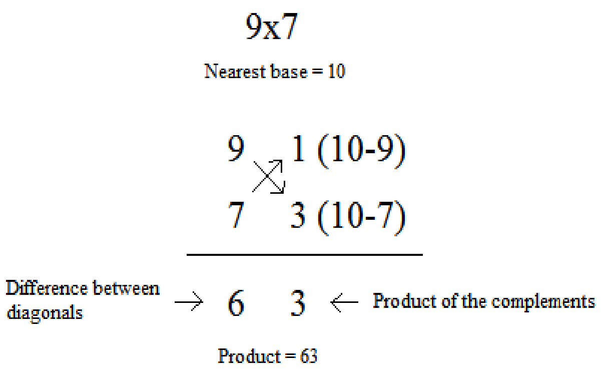


Fig. 1. Example Showing Multiplication Using The Nikhilam Sutra

Here the numbers to be multiplied are 9 and 7 written vertically. The complements of the two numbers 1 and 3 are also written vertically alongside the two numbers. The right part of the result is obtained by multiplying the two complements 1 and 3 which gives 3 while the left portion is obtained by subtracting the diagonal elements to obtain the difference as 6 (9-3 or 7-1). Thus the product is obtained as 63. This method can be extended to large numbers as well.

III. RESIDUE NUMBER SYSTEM

The speed of the arithmetic systems is of primary concern in any application. The conventional representations of numbers include decimal, binary and other weighted systems. The Residue Number System (RNS) represents numbers in a way which removes the carry propagation in an arithmetic unit. This in turn makes it high speed and low power arithmetic. RNS is very good for applications like digital signal processing, computer security, image & speech processing and also in embedded processors, especially those found in portable devices, for which power consumption is the most critical aspect of the design [14]. RNS involves the use of a set of relatively prime integers called moduli. This set can be of the form $\{m_1, m_2,\ldots,m_n\}$. Every number is then represented by n integers which correspond to the remainders with respect to each modulus. These n integers are called residues and are of the form $\{r_1, r_2,\ldots,r_n\}$. So, for any number X, we have,

$$X \bmod m_i = r_i \tag{1}$$

$$\& \quad X = \{r_1, r_2,\ldots,r_n\} \tag{2}$$

The dynamic range of this system is given by,

$$M = \pi m_i \tag{3}$$

All numbers lying in this dynamic range are uniquely represented in this system. Thus, we have a system which ensures parallel independent operations on small number of digits. This significantly speeds up the arithmetic operations especially the addition and the multiplication. Since each modulus is only of a small number of bits, the carries are strictly minimal. This makes it implementable through the use of Look-up Tables (LUTs).

The number in its binary form is first converted into the residual form by choosing a particular set of moduli. This set is chosen on the basis of some rules which uniquely represent the numbers. The largest modulus determines the speed of the operation, so the smallest possible moduli are taken. According to the dynamic range required for the application, the choice of the moduli is made. Here, for 8-bit operations, the range is 256. According to this range, moduli are chosen by taking prime numbers in sequence and including powers of smaller primes before moving to larger primes. The moduli are relatively prime and are represented by the smallest possible number of bits as this decides the speed of the arithmetic. After the set is decided, it is used for the conversion of binary numbers to RNS form. The moduli of powers of 2 ranging from 0 to 7 for 8-bit numbers with respect to the set of moduli chosen are stored in a LUT. The following equality is used for the conversion,

$$<(X_{k-1}X_{k-2}\ldots X_1X_0)>m_i = <<2^{k-1}X_{k-1}>m_i + <2^{k-2}X_{k-2}>m_i + \ldots\ldots. + <2^1X_1>m_i + <2^0X_0>m_i>m_i \tag{4}$$

Corresponding to each modulus, the residues are obtained on the basis of the above equality. The arithmetic operations are performed on these residues which provide a carry free operation. The residue to binary conversion is performed using the Chinese Remainder Theorem (CRT). Digits of the residue number system (RNS) have position weightings s_i which are given by

$$s_i = M/m_i \tag{5}$$

According to the CRT, the binary form of the number is given by,

$$X = \{\Sigma\, s_i\, [(X_i s_i^{-1}) \bmod m_i]\} \bmod M$$
$$\text{where}(s_i s_i^{-1}) \bmod p_i = 1. \qquad (6)$$

This is also most efficiently done through a LUT as a very small number of entries are required in the table which is equal to the sum of the moduli.

IV. PROPOSED MULTIPLIER

The proposed multiplier uses Nikhilam Sutra of Vedic mathematics along with a residue adder to get a high performance 8-bit Vedic multiplier. The block diagram of the proposed multiplier is shown in Fig. 2.
The two 8-bit numbers, the multiplicand and the multiplier, are both complemented when passed through the complementer block which corresponds to the subtraction of the numbers from the nearest base in the Nikhilam Sutra. The complements which are also of 8 bits each are then multiplied to obtain the right part of the product. The multiplication of the complements yields a 16-bit result. The lower 8 bits form the right part of the product while the higher 8 bits are carried over to the residue adder via a binary to residue converter. The multiplicand and the multiplier are also passed through binary to residue converters. The converters output three numbers in their residual form. These residues are added using a residue adder. The left part is then obtained by passing the output of the residue adder through a residue to binary converter which converts the residues back into their binary form. The complete product is obtained by concatenating the left and the right parts.

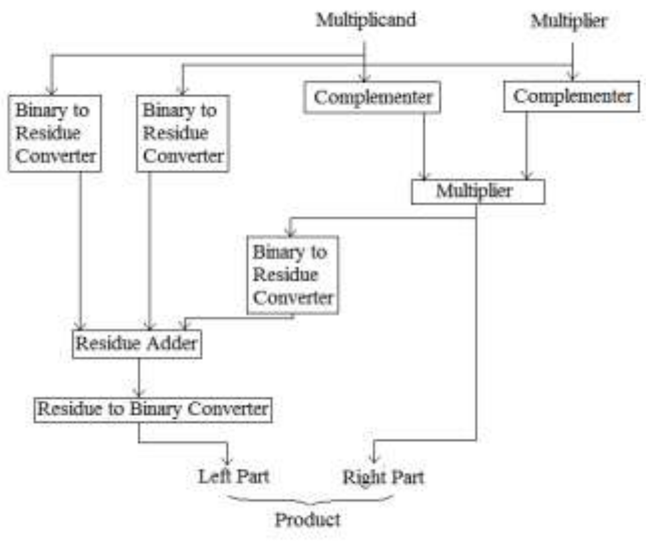

Multiplicand
Multiplier
Binary to Residue Converter
Binary to Residue Converter
Complementer
Complementer
Multiplier
Binary to Residue Converter
Residue Adder
Residue to Binary Converter
Left Part
Right Part
Product

Fig. 2: Block Diagram of the proposed multiplier

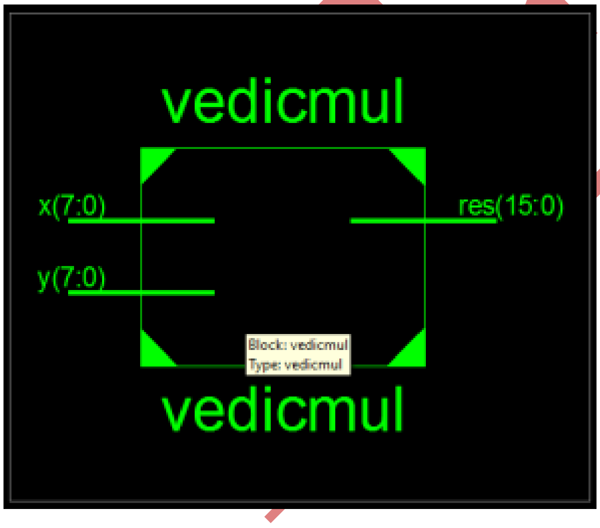
The use of residue adder optimizes the performance parameters to obtain a highly efficient multiplier. The three inputs to the residue adder are of 8 bits and so the maximum value can be 255. Based on this and the concept of RNS, three moduli are chosen to represent the numbers. The three moduli taken here are 5, 11 and 14. The binary to residue converters convert binary numbers into residue numbers using these three moduli. Look-up tables are simulated using VHDL to carry out the conversion. The modulus operation is implemented using Finite State Machine (FSM) concept having three states. For any residue, the modulus is subtracted from the binary number until the remainders are obtained. Three residues then represent one binary number. This gives us in total nine residues, three for each binary number. These are added correspondingly by the residue adder to give the output three residues of the sum. These are then converted back into binary form using the residue to binary converter which is based on CRT. This residue arithmetic enables us to improve the delays by removing the carry propagation due to smaller residues involved in addition rather than large 8-bit numbers.

V. RESULTS

The proposed multiplier using Nikhilam Sutra of Vedic mathematics incorporating a residue adder has been designed and simulated in Very High Speed Integrated Circuits (VHSIC) Hardware Description Language (VHDL) coding using the Xilinx Design Suite 14.2 and synthesis results are drawn for the XC3S50 device and PQ208 package of the Spartan 3 family. Table 1 compares the results with the multiplier using carry save adder proposed by Pradhan and Panda in [13]. The RTL schematic of the design is shown in Fig. 3. The external schematic with I/O's and one of the sub modules inside the complete design. The complete schematic is made up of many such sub modules and other sub modules combined together.

TABLE 1. Comparison of timings of the two multipliers

Device: Spartan 3 XC3S50: PQ208	Vedic multiplier based on CSA proposed by Pradhan & Panda	Proposed multiplier based on Residue Adder
Delay	19.851 ns	15.912 ns



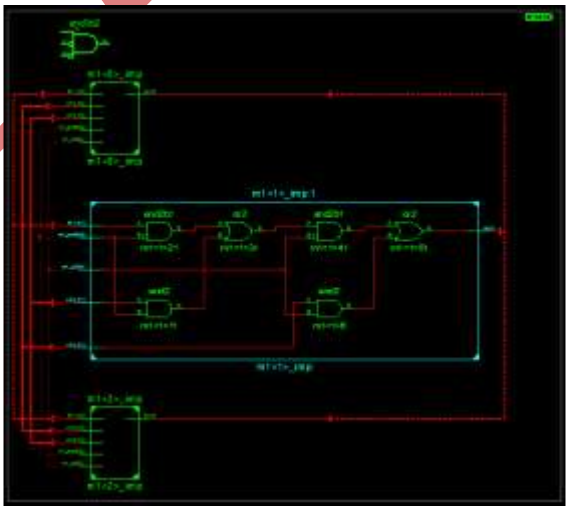


Fig. 3 (A) RTL Schematic Of The Proposed Multiplier (B) Schematic Of One Of The Sub Modules In The Design.

VI. CONCLUSION

In this paper, an efficient Vedic multiplier based on Nikhilam sutra of Vedic mathematics and residue number system based residue adder is proposed. The functionality is verified through VHDL simulations and synthesis on Xilinx Spartan platform. The simulation results show that the proposed RNS based vedic multiplier has significant speed improvement over CSA based realization. The 8-bit multiplier presented here can easily be extended to 16 bits.

REFERENCES

[1] S Laxman, R Prabhu Darshan, M. Shetty, Manjula, C. Sharma, FPGA Implementation of different multiplier architectures, *IJETAE*, ISSN 2250-2459, Vol. 2, Issue 6, June 2012.

[2] M. Chaudhary, M. S. Narula, High Speed Modified Booth's Multiplier for Signed and Unsigned Numbers, *International Journal of Science and Engineering Investigations*, 2, 2013, 106 - 113.

[3] Anju, Performance Comparison of Vedic Multiplier and Booth Multiplier, *International Journal of Engineering and Advanced Technology*, 2, 2013, 336 - 339.

[4] G.Vaithiyanathan, K. Venkatesan, S. Sivaramakrishnan, S. Siva, S. Jayakumar, Simulation and Implementation of Vedic Multiplier using VHDL Code, *International Journal of Scientific & Engineering Research*, 4(1), 2013, 1-5.

[5] M. Poornima, S. K. Patil, Shivukumar , K. P. Shridhar , H. Sanjay, Implementation of Multiplier using Vedic Algorithm, *International Journal of Innovative Technology and Exploring Engineering*, 2 (6), 2013, 219 – 223.

[6] P. Sharma, Design of 4x4 bit Vedic Multiplier using EDA Tool, *International Journal of Computer Applications*, 48, 2012, 32-35.

[7] P. Verma, K. K. Mehta, Implementation of an Efficient Multiplier based on Vedic Mathematics Using EDA Tool, *International Journal of Engineering and Advanced Technology*, 1, 2012, 75-79.

[8] S. Karthik, P. Udayabhanu, FPGA Implementation of High Speed Vedic Multipliers, *International Journal of Engineering Research & Technology (IJERT)*, 1, 2012.

[9] Kavita, U. Goyal, FPGA Implementation of Vedic multiplier, *IJARET*, 4, 2013, 150-158.

[10] A. Haveliya, FPGA implementation of a Vedic convolution algorithm, *IJERA*, 2, 2012, 678-684.

[11] Ambika R , C S Mala , S K Pushpa, FPGA implementation of AES using Vedic Mathematics, *International Journal of Innovative Research in Science & Engineering*, 1(2), 2013, 1-8.

[12] M. Pradhan, R. Panda, S. K. Sahu, Speed Comparison of 16x16 Vedic Multipliers, *International Journal of Computer Applications* (0975 – 8887), 21(6), 2011, 16-19.

[13] M. Pradhan, R. Panda, High speed multiplier using Nikhilam Sutra algorithm of Vedic mathematics, *International Journal of Electronics*, vol. 101, pp. 300 – 307, 2014.

[14] *Residue Numbers and the Limits of Fast Arithmetic*, www.stanford.edu_class_ee486_doc_chap2.