A conference on smart sensors and systems was organized by IEEE, Banglore section, titled "IC-SSS 2015". It emphasized on building and integrating various systems to develop rather more versatile system that is easily adaptable in society. Students of fifth semester, electrical department had worked hard to get their paper presented at the conference, held from December 21st - 23rd, 2015 at MSRIT, Banglore.

Vedic and Conventional Methods of $N \times N$ Binary Multiplication with Hardware Implementation

Pranav Patel, *Ashish Shandilya*, *Nisarg Brahmbhatt*, *Kshitij Raval*, *Dipankar Deb*

Department of Electrical Engineering
Institute of Infrastructure Technology, Research and Management
Ahmedabad, Gujarat, India, 380026.

***Abstract*— A comparative study of the resources involved in the Multiplication of two N-Bit binary numbers is performed using a Vedic Multiplication and the Modern Binary Multiplication techniques. A generic Vedic method for $N \times N$ Binary Multiplication is presented. From first principles, using logic gates, an $N \times N$ Binary Multiplication is performed using both the methods for different number of bits, using Transistor Transistor Logic (TTL) gates. It is found that the propagation delay encountered while using the Vedic Multiplication technique is lesser, more so when the size of multiplication is more. Additionally, *Nikhilam Sutra* is presented for Multiplication of large binary numbers close to a chosen base and comments are made as to how such Multiplication are effectively reduced to Multiplication of smaller numbers.**

Keywords: Vedic Mathematics, Logic Gates, Integrated Circuits, Propagation delay.

I. Introduction

Researchers interested in very accurate sensing of diverse signals are increasingly relying on faster computations which are at the very core of such sensing. One of the ways in which this objective is being explored today is through what is known as Vedic Mathematics, an interesting, simple but innovative of fast computations that was first presented by Krisna Tirthaji Swami in 1965.[1]

Speed and efficiency of computation is often an important factor in effective usage of digital sensors. Digital Multipliers are at the core of all digital signal processors (DSPs) and largely determine the speed of the DSPs and digital sensors. Due to the importance of digital multipliers in DSP, it has always been an active area of research and a number of interesting multiplication algorithms have been reported in the literature.[2],[5]

A number of multiplier architectures also have been proposed based on these algorithms that include parallel, serial and serial-parallel multipliers. In an array multiplier, multiplication is based on shift and add. The partial product is generated by the bitwise multiplication of the multiplicand with one multiplier bit. The partial product are shifted according to their bit orders and then added with carry propagate adder. Array multiplier is easy to design due to its regular structure.[3] However, as the width of the multiplier increases, the delay becomes more.

This paper presents a Vedic multiplication technique called *Urdhva Tiryakbhyam Sutra* which is equally applicable to all cases of multiplication. Attempts have been made in the literature to apply this general multiplication formula to binary arithmetic.[4],[5] This paper also presents a new multiplication algorithm which circumvents the need of large multipliers by reducing the multiplication of large numbers to that of smaller numbers. This reduces the propagation delay associated with the conventional large multipliers considerably. The framework of the proposed algorithm is based on the *Nikhilam Sutra* of Vedic Mathematics.

In this paper, the partial products of the multiplier are added by using fast carry adders. Hardware implementation indicates that the proposed multiplier with fast carry adders can achieve improvement in propagation delay when compared to conventional multipliers. *Urdhva Tiryakbhyam Sutra* is first applied to the binary number system and is used to develop a digital multiplier architecture. This is demonstrated to be similar to the popular array multiplier architecture. *Nikhilam Sutra* is more efficient in the multiplication of two large numbers as it reduces it to that of two comparatively smaller numbers.[7] The basic framework of the proposed algorithm is taken from the *Nikhilam Sutra* of Vedic Mathematics and is further optimized to take full advantage of the bit-reduction in multiplication.

In Section II, we present a typical conventional multiplication method as implemented in actual hardware with gates and fast carry full adder ICs. In Section III, we discuss the *Urdhva Tiryakbhyam* (Vertical and Crosswise) method of multiplication using Vedic Mathematics in detail. We describe the generic $n \times n$ bit multiplication by *Urdhva Tiryakbhyam Sutra* and an equation is developed which provides the individual digits. In Section IV, Propagation delay in the case of different sizes of binary multiplication is investigated from first principles (from gates and ICs) for two different methods of multiplication. Section V presents the *Nikhilam Sutra* for binary numbers and demonstrates how it effectively reduces multiplication of large numbers into multiplication of smaller numbers. This Section also compares *Nikhilam Sutra* with another conventional multiplication method for large numbers called Karatsuba algorithm and finds it to be faster than the latter.[10]

II. A Conventional Multiplication Method

Multiplication of binary numbers is performed in several different ways. A typical method is wherein the multiplicand of n bits is multiplied by each bit of the multiplier

of n bits, starting from the least significant bit.[8] Each such multiplication forms a partial product, the successive partial products are shifted one position to the left, and the final product of a maximum of $2n$ bits is obtained from the sum of these partial products.

For two 4-Bit binary multiplication, the multiplicand bits are $b_3\ b_2\ b_1\ b_0$, and the multiplier bits are $a_3\ a_2\ a_1\ a_0$, and the product is given by $c_7\ c_6\ \ldots\ c_0$. These outputs can be obtained using 16 AND gates (IC 7408) and 3 Full adders (IC 7483).

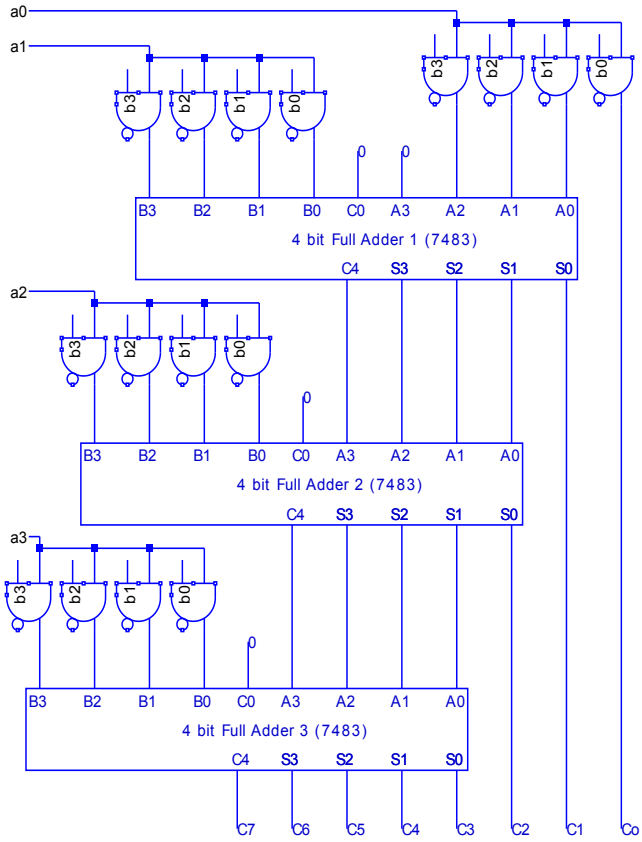

Fig. 1. 4×4 Binary Multiplication by Conventional Multiplier Method.

There are many other conventional techniques with minor differences between them which have been reported to have slightly better propagation delays.

III. $n \times n$ Vedic Multiplication

With the *Urdhva Tiryagbhyam Sutra* (Vertical and Crosswise Algorithm), multiplication can be performed in a single line. On the contrary, in the Conventional (shift and add) method, n partial products have to be added to get the result. The reduction in the number of steps, reduces the computation time and increases the speed of the multiplier.

As is well-known now, a $n \times n$ Vedic Multiplication uses the same number of bitwise multiplications but lesser number of additions. Consider two n bit binary numbers:

$$b_{n-1} \ldots \ b_2\ \ b_1\ \ b_0 \tag{1}$$

$$\times\ \ a_{n-1} \ldots \ a_2\ \ a_1\ \ a_0. \tag{2}$$

The result of this binary multiplication can be expressed as a binary number of length $2n$:

$$C_{2n-1}\ \ C_{2n-2} \ldots C_n\ \ C_{n-1} \ldots C_1\ \ C_0, \tag{3}$$

where $C_i, i = 0, 1, \ldots, 2n-1$, are individual binary digits which can be expressed as

$$\begin{aligned}
C_0 &= a_0 b_0 \\
C_1 &= a_0 b_1 + a_1 b_0 = \sum_{i=0}^{1} a_i b_{1-i} \\
C_2 &= c_2 + a_0 b_2 + a_1 b_1 + a_2 b_0 = c_2 + \sum_{i=0}^{2} a_i b_{2-i} \\
&\vdots \\
C_{n-1} &= c_{n-1} + a_0 b_{n-1} + a_1 b_{n-2} \ldots + a_{n-1} b_0 \\
&= c_{n-1} + \sum_{i=0}^{n-1} a_i b_{n-1-i} \\
C_n &= c_n + a_1 b_{n-1} + a_2 b_{n-2} \ldots + a_{n-1} b_1 \\
&= c_n + \sum_{i=0}^{n-2} a_{i+1} b_{n-1-i} \\
C_{n+1} &= c_{n+1} + a_2 b_{n-1} + a_3 b_{n-2} \ldots + a_{n-1} b_2 \\
&= c_{n+1} + \sum_{i=0}^{n-3} a_{i+2} b_{n-1-i} \\
C_{2n-3} &= c_{2n-3} + a_{n-1} b_{n-2} + a_{n-2} b_{n-1} \\
&= c_{2n-3} + \sum_{i=0}^{1} a_{i+n-2} b_{n-1-i} \\
C_{2n-2} &= c_{2n-2} + a_{n-1} b_{n-1} \\
C_{2n-1} &= c_{2n-1}
\end{aligned} \tag{4}$$

where $c_i, i = 2, 3, \ldots, 2n-1$ represent the terms that are carried over to the ith binary place after the addition has been completed in the previous binary place. Note that C_0 and C_1 terms do not encounter any carry terms due to the very nature of binary addition and C_{2n-1} consists of only the carry term from the previous addition. It is easy to note that $c_2 = a_0 a_1 b_0 b_1$.

For a given binary multiplication of each n bits, that is, $n \times n$ multiplication, the terms of the result of binary multiplication, that is, C_j are expressed as

$$C_j = \begin{cases} \sum_{i=0}^{j} a_i b_{1-i} & \text{if } j = 0, 1 \\ c_j + \sum_{i=0}^{j} a_i\ b_{j-i} & \text{if } j = 2, \ldots, n-1 \\ c_j + \sum_{i=0}^{2n-j-2} a_{i+j-n+1} b_{n-1-i} & \text{if } j = n, \ldots, 2n-2 \\ c_j, & \text{if } j = 2n-1. \end{cases} \tag{5}$$

For scalability of implementation of large multiplications, we propose to use multiple 4×4 bit multipliers to implement a 8×8 multiplication using the Vedic method and also the conventional method.

Without loss of generality, let us consider the case of binary multiplication wherein the multiplicand is $b_1\ b_0$ and multiplier is $a_1\ a_0$. Using the *Urdhva Tiryagbhyam* technique of Vedic mathematics, the result would have 4

terms:

$$\begin{aligned} p_0 &= a_0b_0 \\ p_1 &= a_0b_1 + b_0a_1 \\ p_2 &= a_1b_1 + \text{Carry from } p_1 \\ p_3 &= \text{Carry from } p_2. \end{aligned} \tag{6}$$

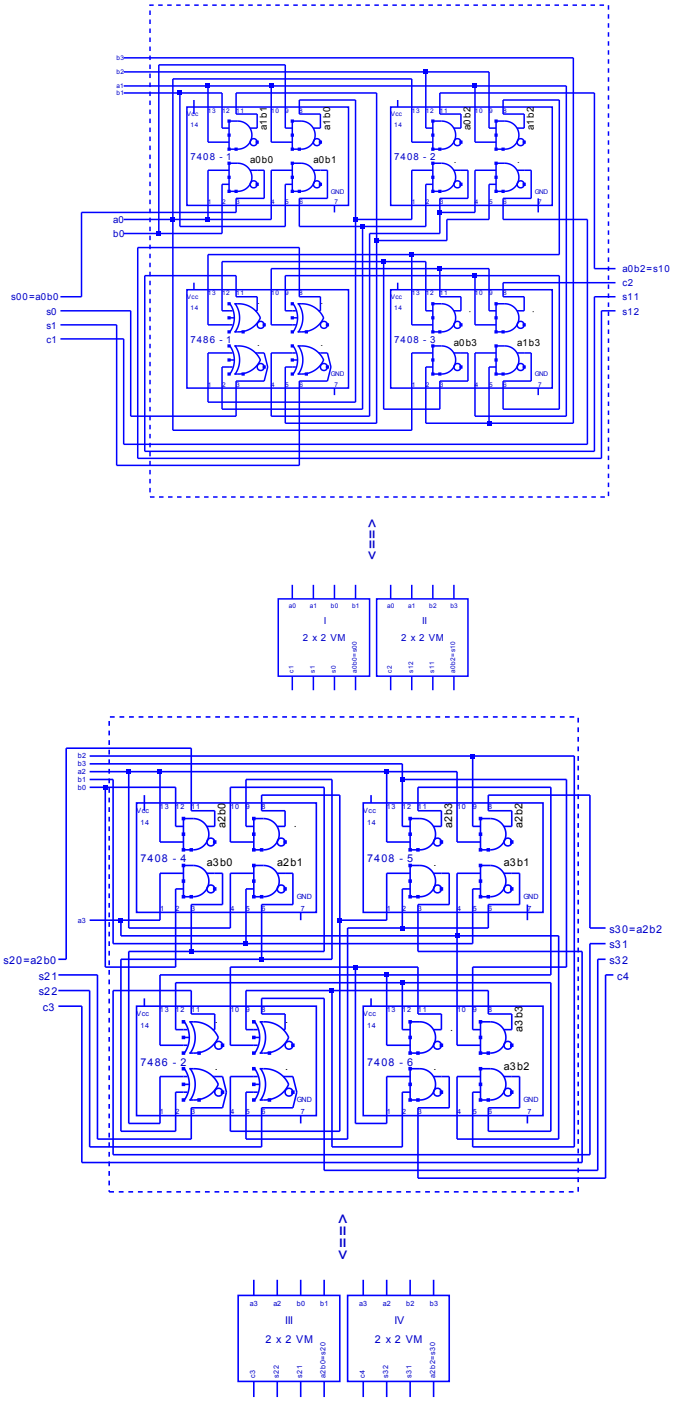

Fig. 2. Four 2×2 Multiplication Blocks with *Urdhva Tiryagbhyam Sutra*.

In the case of 4×4 Vedic multiplication, the above outputs can be obtained using 4 AND gates for $a_0b_0, a_0b_1, a_1b_0, a_1b_1$ and two half adders (2 XOR gates and 2 AND gates), that is, a total of six AND gates (IC 7408) and two XOR gates (IC 7486). Since IC 7408 contains 4 AND gates and IC 7486 has 4 XOR gates, we can implement two such 2-Bit binary multiplication blocks using four XOR gates (1 IC 7486) and twelve AND gates (3 IC 7408) each. We have shown the connections explicitly with respect to the Pin diagrams of the ICs that have been used in this multiplication.

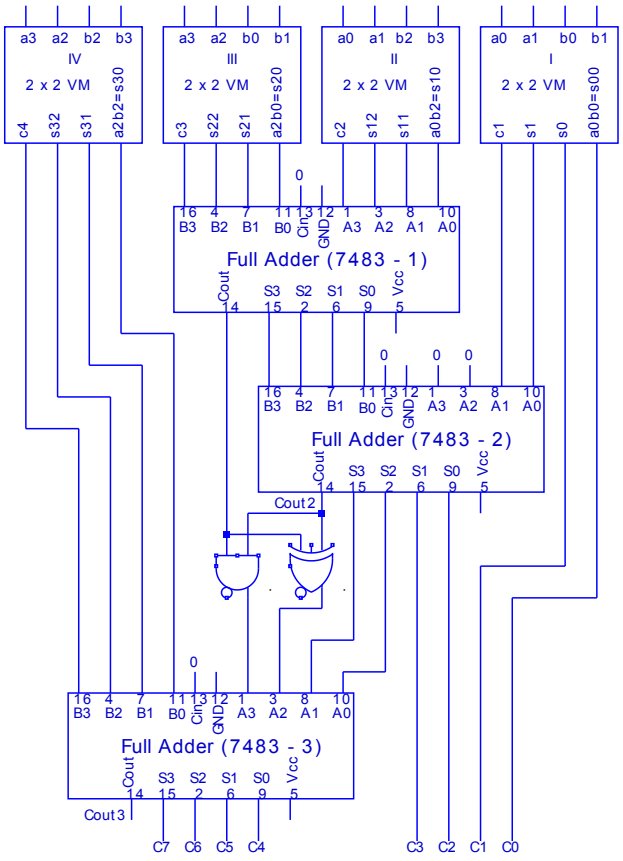

Fig. 3. 4×4 Binary Multiplication by *Urdhva Tiryagbhyam*.

IV. Propagation Delay in Multiplication by Conventional and Vedic Methods

Propagation delay, subsequently represented as t_{pd} (Table 1), is the time required for a digital signal to travel from the input(s) of a logic gate to the output. It is typically measured in nanoseconds. It is the average of the time for the input going high to low or vice-versa and the time taken by output to go from high to low or vice-versa. The propagation delay for an integrated circuit (IC) logic gate may differ for each of the inputs. If all other factors are held constant, the average propagation delay in a logic gate IC increases as the complexity of the internal circuitry increases. Propagation delay has a direct effect on the speed at which a digital device, such as a computer, can operate.

We ascertain that as the number of bits increase the benefit of the described Vedic method becomes increasingly more evident. Note that we have used all logic gates with TTL logic and all ICs are from the same manufacturer (Texas Instruments) so that the comparison presented henceforth is a fair one. It has been found that the Vedic method performs progressively better for products when the number of bits involved in the numbers is higher because of the inherent efficient method of parallel computation.

In Table 1, we compare the propagation delays for the two methods: Conventional Method (Method I) and *Urdhva Tiryagbhyam Sutra* (Method II).

TABLE I

PROPAGATION DELAYS IN DIFFERENT METHODS

Number of Bits	t_{pd} (Method I) (nsecs)	t_{pd} (Method II) (nsecs)
4	98	121
8	373	294
16	1523	617

In the construction of 4-Bit conventional multiplier, we need four AND ICs (7408) and three 4-Bit full adder ICs (7483).[11],[12] The three full adders (25 ns each) operate sequentially and the four AND gate ICs can compute in parallel to each other (23 ns). Thus it will take a total of ($25 \times 3 + 23 = 98$ ns) as shown in Table 1.

For 4 bit Vedic multiplication, we need seven AND ICs (7408), three 4-Bit full adder ICs (7483) and three XOR ICs (7486).[13] The three full adders (25 ns each) operate sequentially and the six AND gate ICs can compute in parallel to each other (23 ns). Additionally, another 23 ns is required for the implementation of half adder in Figure 3. Thus it will take a total of $25 \times 3 + 23 + 25 = 121$ ns.

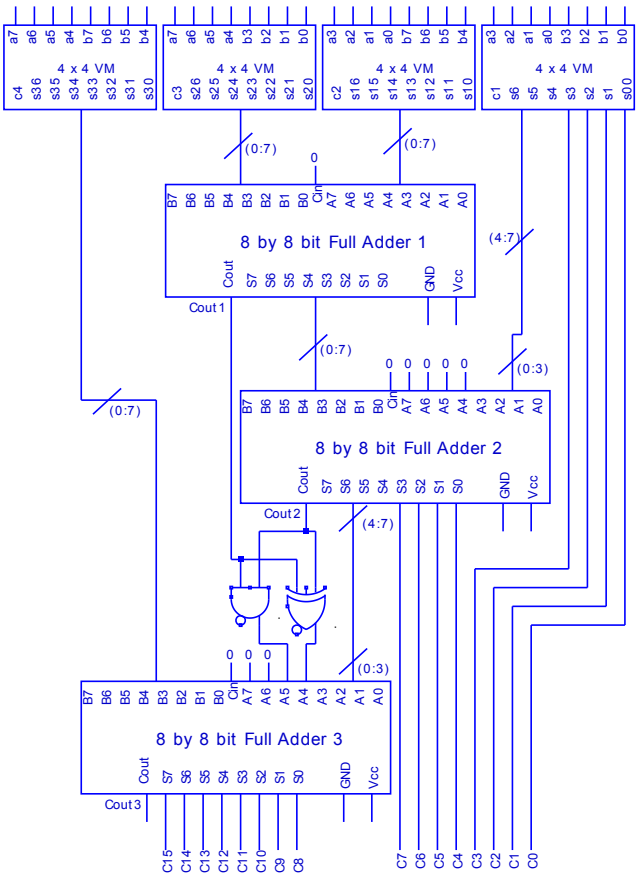

Fig. 4. 8×8 Binary Multiplication (*Urdhva Tiryagbhyam*).

For 8-Bit Vedic multiplication, we need four 4×4 Vedic block (121 ns) and three 8-Bit full adders which can be implemented using two 4-Bit full adder ICs (7483) working sequentially ($3 \times (2 \times 25)$ ns). As already explained in 4-Bit Vedic multiplication, a similar half adder (23 ns) is also required here. Therefore, the total propagation delay = $121 + 3 \times (2 \times 25) + 23 = 294$ ns.

In the implementation of 8-Bit conventional multiplier, we need sixteen AND gate ICs (7408) working in parallel (23 ns), and seven 8-Bit full adders (each implemented using two 4-Bit full adder ICs) working sequentially ($7 \times (2 \times 25)$ ns). Thus, the total propagation delay = $7 \times (2 \times 25) + 23 = 373$ ns.

Similarly, for 16-Bit conventional multiplication, we need sixty-four AND ICs (7408) and fifteen 16-Bit full adders (implemented using four 4-Bit full adder ICs). The total propagation delay = $15 \times (4 \times 25) + 23 = 1523$ ns. For construction of 16-Bit Vedic multiplication, we need four 8×8 Vedic block, three 16-Bit full adders and a half-adder, resulting in a total propagation delay of $3 \times (4 \times 25) + 23 + 294 = 617$ ns.

It is possible to have much faster response times by using logic gates from different families. However, the idea is only to present the progressive benefit of the Vedic methods as the size of multiplication increases. This benefit is expected to remain when improved IC fabrication techniques are used and propagation delays are reduced even further.

In order to be able to determine the viability of the Vedic Method when implemented in the form of an IC, we propose to use the unitary method to determine the speed of computation. We built a Vedic Multiplier Circuit and Conventional Multiplier Circuit for 4 bits using AND Gates and Full adders. We determine the typical (or average) computation time for these circuits as shown in the table above. Additionally, noting that, for example, an IC SN74284 for 4×4 Multiplier from Texas Instruments requires 40 nsecs computation time. A proposed Vedic 8×8 Multiplier IC, if made of the same material, would typically require:

$$t_{pd} = \frac{294 \times 40}{98} = 120 \text{ nsecs.}$$

The computation time would reduce significantly if we build an IC for multiplication with progressively higher number of bits, as indicated by the propagation delays pertaining to the multiplications mentioned in Table I.

V. NIKHILAM SUTRA (FOR LARGE NUMBERS)

The *Nikhilam Sutra* applicable to all multiplications, is more efficient and less complex in the case of large numbers. This method involves multiplying the complements of the large numbers from its nearest base instead of the number themselves.

Although originally intended for decimal numbers, this *Sutra* has since been extended to Binary multiplications. Larger the size of the original numbers, lower are the complements and thus lesser is the complexity of multiplication.

We illustrate *Nikhilam Sutra* by considering the multiplication of two binary numbers (1101×1110) where the chosen base is (1 0 0 0 0) which is nearest to and greater than both these numbers. We write the multiplier and the multiplicand in two rows followed by the differences of each of them from the base (1 0 0 0 0). There are two

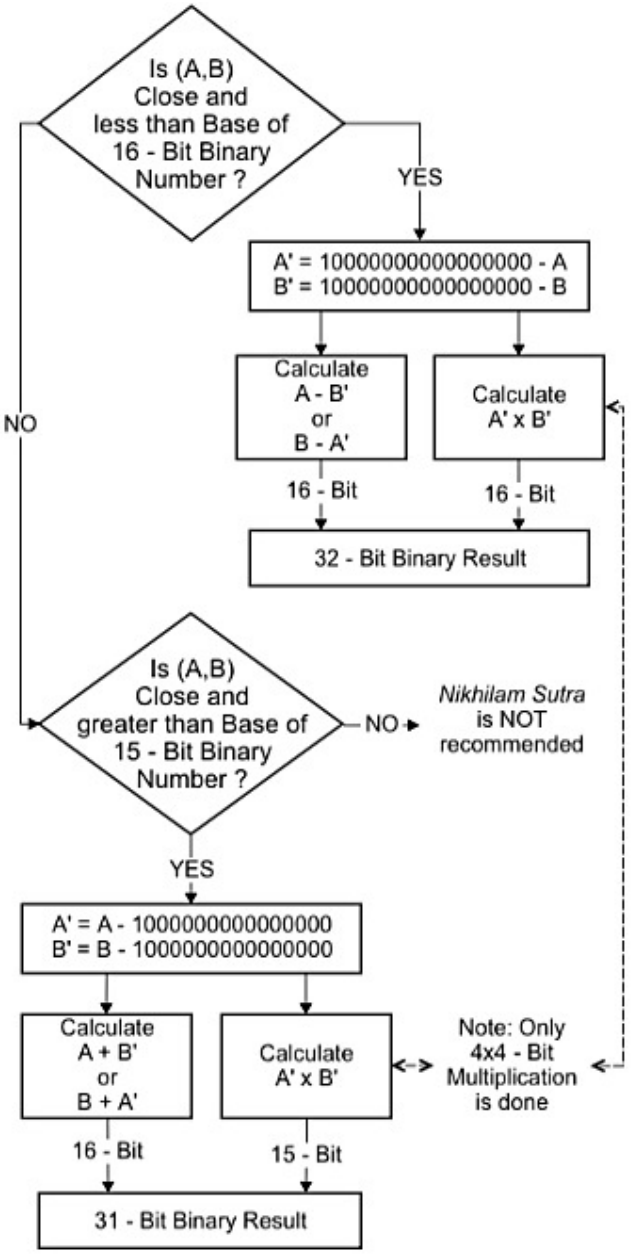

Fig. 5. 16-Bit Binary Multiplication by *Nikhilam Sutra.*

columns of numbers: the first column consists of numbers to be multiplied and the second column consists of their compliments.

$$1\,1\,0\,1 \qquad (1\,0\,0\,0\,0 - 1\,1\,0\,1) = 0\,0\,1\,1$$
$$1\,1\,1\,0 \qquad (1\,0\,0\,0\,0 - 1\,1\,1\,0) = 0\,0\,1\,0$$

Subtracting $0\,0\,1\,0$ from $1\,1\,0\,1$ or $0\,0\,1\,1$ from $1\,1\,1\,0$, we get the four most significant bits $1\,0\,1\,1$ and $0\,1\,1\,0$ are the four least significant bits of the result and is obtained by multiplying the "small numbers" $0\,0\,1\,1$ and $0\,0\,1\,0$. Therefore the result of the multiplication operation is

$$1\,0\,1\,1\,0\,1\,1\,0.$$

Note that in case of multiplication of 4-Bit binary numbers, this method is most effective when both the numbers are greater than $1\,0\,0\,0$.

In general, the product m obtained through *Nikhilam Sutra* for two n-Bit binary numbers a and b with compliments $a' = 2^n - a$ and $b' = 2^n - b$ respectively, is given by

$$\begin{aligned} m = ab &= (2^n - a')(2^n - b') \\ &= 2^n(2^n - b' - a') + a'b' \\ &= 2^n(b - a') + a'b' = 2^n(a - b') + a'b'. \end{aligned} \quad (7)$$

Note that (8) clearly demonstrates why the product of binary numbers is equal to product of their complements in the four least significant bits of the result and the *cross-difference* $(a - b' = b - a')$ makes up the most significant four bits of the result because of the existence of the exponent 2^n as the coefficient.

Alternatively, if the binary numbers a and b are close to but greater than the base 2^{n-1} such that their compliments $a' = a - 2^{n-1}$ and $b' = b - 2^{n-1}$, we have

$$\begin{aligned} m = ab &= \left(2^{n-1} + a'\right)\left(2^{n-1} + b'\right) \\ &= 2^{n-1}\left(2^{n-1} + b' + a'\right) + a'b' \\ &= 2^n(b + a') + a'b' = 2^n(a + b') + a'b'. \end{aligned} \quad (8)$$

As illustrated in Fig. 5, for 16-Bit binary numbers, *Nikhilam Sutra* is most effective for multiplication of numbers very close to the base. Application of *Nikhilam Sutra* is very efficient for 16-Bit binary numbers for the range 65520 to 65535 when it is close to base 2^{16}, and for the range 32768-32783 when the number is close to 2^{15}.

If numbers in the range 65520 to 65535 are multiplied by *Nikhilam Sutra*, the multiplication effectively reduces to a multiplication of 4-Bit numbers since the first 12 bits are all 1's in both the numbers. The second term in (7) equals multiplication of four bit numbers. With *Nikhilam Sutra*, the closer the numbers are to the base, the simpler is the product of the compliments and as a consequence the *cross-difference* is simply a subtraction operation. Hence, when both the multiplicand and the multiplier are closer to their respective base, the *Nikhilam Sutra* is seen to be faster than the *Urdhva Tiryakbhyam Sutra*.

In order to evaluate the effectiveness of the *Nikhilam Sutra*, it is compared to a conventional multiplication method for larger numbers called the Karatsuba algorithm: a formula that allows computation of the product of two large numbers and using three multiplications of smaller number, each with about half as many digits, some additions and digit shifts.[10]

For multiplication of two binary numbers k_1 and k_2, the numbers are first split as follows:

$$\begin{aligned} k_1 &= k_{1a}\,2^s + k_{1b}, \\ k_2 &= k_{2a}\,2^s + k_{2b}, \end{aligned} \quad (9)$$

where $k_{1b}, k_{2b} < 2^s$ and s is typically a positive integer starting from 2, such that

$$\begin{aligned} k_1 \times k_2 &= K_2 2^{2s} + K_1 2^s + K_0, \\ K_2 &= k_{1a}k_{2a}, \\ K_1 &= (k_{1a} + k_{1b})(k_{2a} + k_{2b}) - K_2 - K_0, \\ K_0 &= k_{2a}k_{2b}. \end{aligned} \quad (10)$$

Karatsuba algorithm for $k_1 \times k_2$ involves three multiplications along with a few additional additions.

We take two 16-Bit binary numbers for comparing the *Nikhilam Sutra* and Karatsuba Algorithm, in the range 32768 and 32782 respectively. Since the numbers are close to 2^{15} and greater than base, we need to calculate A' and B' for *Nikhilam Sutra* as suggested in Figure 5. We note that A' and B' are the last four bits from both the numbers which are 1010 and 1100 respectively. The product is calculated as shown in flowchart as 10000000000101100000000001111000 whose decimal equivalent is 1074462840. With Karatsuba Algorithm, for the same 16-Bit binary numbers, we choose k_{1a}, k_{1b}, k_{2a} and k_{2b} as 1, 1010, 1 and 1100 respectively, taking 2^{15} as base.

Nikhilam Sutra incorporates one 4-Bit full adder IC (7483) for calculating $A + B'$ or $B + A'$ and a 4-Bit multiplier Block (either by Conventional or Vedic method) for multiplication $A' \times B'$. We can infer that both multiplication and addition can be done in parallel. While in the Karatsuba Algorithm, calculation for the product consists of 3 steps which include two additions of 4-bit binary numbers $k_{1a} + k_{1b}$ and $k_{2a} + k_{2b}$ using 4-bit full adder IC (7483). Secondly, two 4-Bit multiplications $(k_{1a} + k_{1b})(k_{2a} + k_{2b})$ and $k_{1b}k_{2b}$ are required which can be evaluated using 4-bit Vedic or Conventional multiplier blocks. Finally, two subtractions are needed to evaluate K_1. K_2 is 1 in our case as k_{1a} and k_{2a} are chosen to be 1. Thus, *Nikhilam Sutra* is faster than Karatsuba algorithm when two n-Bit numbers are to be multiplied and are close to the base, that is, 2^{n-1}.

VI. Concluding Remarks

Two distinct Vedic Multiplication techniques have been studied from first principles and implemented on breadboard to compare the benefit obtained in terms of reduced propagation delays with respect to a conventional multiplication method; especially for larger size of binary multiplication. It is also evident that the *Nikhilam* technique is very efficient for multiplications where both the multiplicand and the multiplier are closer to their respective base. Propagation delay in multiplication when used with sensors, will cause faster sensing and thus prevent erroneous signal transmission and interpretation. Applications requiring the usage of fast sensors wherein fast computations and reduced propagation delays through the use of Vedic Mathematics leading to sensor reliability, would be an important area for further work.

References

[1] Jagadguru Swami Sri Bharati Krisna Tirthaji Maharaja, *Vedic Mathematics: Sixteen Simple Mathematical Formulae from the Veda*, Delhi, 1965.

[2] H. Thapliyal, "Vedic Mathematics for Faster Mental Calculations and High Speed VLSI Arithmetic,"*Invited talk at IEEE Computer Society Student Chapter*, University of South Florida, Tampa, FL, Nov 14, 2008.

[3] N. H. E Weste, D. Harris, A. Banerjee, *CMOS VLSI Design, A Circuits and Systems Perspective*, Third Edition, Pearson Education, PP. 327-328, 2006.

[4] G. G. Kumar and V. Charishma, "Design of High Speed Vedic Multiplier using Vedic Mathematics Techniques," *International Journal of Scientific and Research Publications,* Vol. 2(3), March 2012.

[5] A.P. Nicholas, K.R Williams, J. Pickles, "Application of *Urdhava Sutra*," Spiritual Study Group, Roorkee, India, 1984.

[6] P. Mehta and D. Gawali, "Conventional versus Vedic mathematical method for Hardware implementation of a multiplier," *IEEE International Conference on Advances in Computing, Control, & Telecommunication Technologies,* 2009.

[7] H. S. Dhilon and A. Mitra, "A Reduced-Bit Multiplication Algorithm For Digital Arithmetic," *International Journal of Computational and Mathematical Sciences,* Waset, Spring, 2008.

[8] M. M. Mano and M. D. Ciletti, *Digital Design With an Introduction to the Verilog HDL*, Fifth Edition, Pearson India, 2013.

[9] D. Zuras, "On squaring and multiplying large integers," *Proceedings of International Symposium on Computer Arithmetic,*" IEEE Computer Society, pp. 260-271, 1993.

[10] A. Karatsuba and Y. Ofman, "Multiplication of Many-Digital Numbers by Automatic Computers," *Proceedings of the USSR Academy of Sciences 145: 293-294. Translation in the academic journal Physics-Doklady*, 7 (1963), pp. 595-596.

[11] Texas Instruments, "Quadruple 2-Input Positive-AND Gates," SN74S08 datasheet, 1988.

[12] Texas Instruments, "4-Bit Binary Full Adders with Fast Carry," SN7483A datasheet, 1988.

[13] Texas Instruments, "Quadruple 2-Input Exclusive-OR Gates," SN74LS86A datasheet, 1988.