A NEW APPROACH TO DESIGN AN ARITHMETIC LOGIC UNIT BASED ON ANCIENT VEDIC MATHEMATICS

K.BHARATHA BABU[1], REEBA KORAH[2], A.SWARNALATHA[3]

[1]Department of ECE, Research Scholar, Anna University, Chennai, Tamil Nadu, India,

[2]Alliance College of Engineering and Design, Alliance University, Bangalore-562106, Karnataka, India

[3]St. Joseph's College of Engineering, Chennai-600119, Tamil Nadu, India

[1]kbharathababu@gmail.com, [2]reeba26in@gmail.com,[3]swarnalatha7@gmail.com

---

***Abstract*—ALU is the cardinal functional unit in digital signal processor and embedded system devices which perform complex arithmetic and logical functions. In this paper we propose an ALU architecture (Vedic coprocessor) which is an integral unit of arithmetic and logical unit such as multiplication, division, square, cube, square root and cube root units. Each and every unit has an architecture based on unique Vedic math sutras. This proposed ALU architecture overcomes the existing drawbacks such as high delay, irregular structure of combinational circuits and high power dissipation. Vedic ALU is designed and simulated in XILINX ISE simulator and implemented using Spartan 3 FPGA. The proposed ALU is equivalent to Vedic coprocessor which increases the efficiency of multiprocessor configuration system design.**

Keywords: **Vedic mathematics, Nikhilam sutra, Urdhva Tiryakbhyam, Yavadunam, Anurupya, ALU, Multiplier.**

I. INTRODUCTION

Always challenges are grown in the direction of handling complex arithmetic and logical functions in DSP Processor and embedded processor. Multiprocessor configuration technique helps in integration of number of processor cores into one chip. Complexity is reduced as main processor works with coprocessor. Coprocessors are the processors that are designed to work on different task like numeric computation, signal processing and graphics. Performance of the ALU greatly depends on the multiplier. Existing multiplier techniques such as redundant complex number system [RCNS], CORDIC and bit serial multiplication suffer from the problem of long latency, large rearranging of pre/post processing and the necessity to have regular structure. To overcome these disadvantages Vedic mathematics is used. Based on various Vedic aphorisms' various arithmetic modules such as multiplication, division square, cube square root, and cube root modules are designed and integrated into Vedic ALU.

II. PROPOSED TECHNIQUE

A.Nikhilam Navata Charanam Dashatah

An Aphorism which simply means: "all from 9 and the last from 10". This aphorism works efficiently for multiplication of numbers, which are nearer to bases of 10, 100, 1000 i.e. increased powers of 10. The procedure of multiplication using the Nikhilam involves minimum mental manual calculations. This in turn will lead to reduced number of steps in computation, hence reducing the space and saving more time for computation. The numbers taken can be either less or more than the base considered. The mathematical derivation of the algorithm is given below.

Consider two n-bit numbers x and y to be multiplied. Then their complements can be represented as[5]

$x_1 = 10^n - x$ and $y_1 = 10^n - y$.

The product of the two numbers can be given as

$p = (x, y)$.

Now a factor $10^{2n} + 10^{n} (x + y)$ is added and subtracted on the right hand side of the product, which is mathematically expressed as shown below.

$p = xy + 10^{2n} + 10^{n} (x + y) - 10^{2n} - 10^{n}(x +$

$y)$ On simplifying we get,

$p = \{10^{n} (x + y) - 10^{2n} + \{10^{2n} - 10^{n} (x + y) +$

$xy\} = 10^{n} \{(x + y) - 10^{n}\} + \{(10^{n} - x) (10^{n} -$

$y)\} = 10^{n} \{x - y_1\} + \{x_1 y_1\}$

$= 10^{n} \{y - x_1\} + \{x_1 y_1\}$

From the above equation we can derive the left-hand side of the product as $\{x - y_1\}$ or $\{y - x_1\}$ and the right hand side as $(x_1.y_1)$.The basic operations involved in the algorithm for a given set of numbers are given below. Consider 88 x 98.

Here the Nearest Base = 100

88	(100 – 88)
98	(100 - 98)

Column 1	column 2
88	12
98	2
86	24
2 Digits	2 Digits

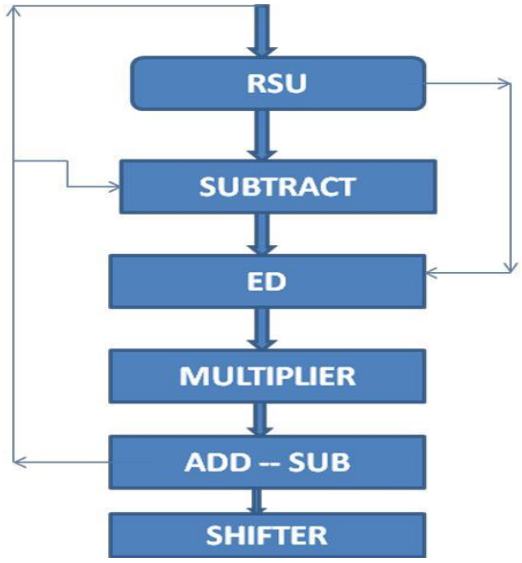

Fig 1: Hardware Flow Diagram for Nikhilam sutra

B. Vedic Squaring

Duplex D property of Urdhva Triyagbhyam is used to calculate the square of the number. In the Duplex, we take twice the product of the outermost pair and so on till no pairs are left [10].When there are odd no's of bits in the original sequence, there is one bit left by itself in the middle and this enters as its square.

Algorithm for 4*4 bit square using Urdhva Triyagbhyam D-Duplex [4].

Y3 Y2 Y1 Y0 Multiplicand

Y3 Y2 Y1 Y0 Multiplier

H G F E D C B A

P7 P6 P5 P4 P3 P2 P1 P0 Product

Parallel Computation

1. D=Y0*Y0=A

2. D=2*Y1*Y0=B

3. D=2*Y2*Y0+Y1*Y1=C

4. D=2*Y3*Y0+2*Y2*Y1=D

5. D=2*Y3*Y1+Y2*Y2=E

6. D=2*Y3*Y2=F

7. D=Y3*Y3=G

Squaring algorithm and the corresponding architecture was implemented with the aid of "Yavadunam Sutra".

Mathematical formulation of Yavadunam Sutra

$Y^2 = 2^n$(Y-2's complement of Y) + (2's complement of Y)^2

C. Vedic Cube

Anurupya sutra of Vedic mathematics is used to find the cube of the number. Aphorism states that "If you start with the cube of the first digit and take the next three numbers (in the top row) in a Geometrical Proportion (in the ratio of the original digits

themselves) then you will find that the 4th figure (on the right end) is just the cube of the second digit".

If a and b are two digits then according to Anurupya Sutra[1],

a^3 a^2b ab^2 b^3

$2a^2b$ $2ab^2$

$a^3 + 3a^2b + 3ab^2 + b^3 = (a + b)^3$

This sutra has been utilized in this work to find the cube of a number. The number M of N bits having its cube to be calculated is divided in two partitions of N/2 bits, say a and b, and then the Anurupya Sutra is applied to find the cube of the number. In the above algebraic explanation of the Anurupya Sutra, we have seen that a3 and b3 are to be calculated in the final computation of $(a+b)^3$.

The intermediate a3 and b3 can be calculated by recursively applying Anurupya sutra. A few illustrations of working of Anurupya sutra are given below.

$(15)^3$ = 1 5 25 125

10 50

3375

D. Vedic Square Root

The method suits for only perfect squares. Finding square root is quiet difficult. The only method known to us till now is to find the factors of a number group the factors taking two at a time and take out one number from a group of two.

Eg: 64 = 2*2*2*2 *2*2

Taking out => 2*2*2=8

Therefore, √64 = 8.

A better and efficient way in solving the problem is use of Vedic maths which is described as follows:-

Squaring of numbers ending in 5 uses a sutra called 'Ekadhikena Purvena' [2]

The Sutra (formula) Ekādhikena Pūrvena means: "By one more than the previous one".

Consider 45^2

$45 = (40 + 5)^2$, It is of the form $(ax+b)^2$ for a = 4, x=10 and b = 5. Answer =a (a+1) / 25

That is, 4 (4+1) / 25 = 4 * 5 / 25 = 2025.

The algorithm for working with perfect squares is given below:

If the number is a perfect square, then we can know what the ending digit of the answer will be by looking at the ending digit of the question.

If the number ends in a:

0 -> then the ending digit is a 0.

1 -> then the ending digit is 1 or 9.

4 -> then the ending digit is 2 or 8.

5 -> then the ending digit is a 5.

6 -> then the ending digit is 4 or 6.

9 -> then the ending digit is 3 or 7.

After finding the last digit (or possibility between two digits) chop off the last two digits and focus on the remaining digits.

Then find out if the answer is in the upper-half of the range (i.e. ends in 5, 6, 7, 8, or 9) or if the answer is in the lower-half of the range (i.e. ends in 0, 1, 2, 3, or 4) by squaring the middle number (the number in the range that ends in 5).

Steps to find the square root of 5329:

We know the last number must be a 3 or 7.

Chopping off the last two numbers we need to focus on the remaining numbers or 53.

We know that 7^2 = 49 and 8^2 = 64 so since 53 is between these two numbers; the answer is between 70 and 80.Squaring 75 we get 5625. Since the original number is lower than this, we know the answer is in the lower-half.

The answer is 73.

E. Cube Root Using Vedic Mathematics

Similar to square root, cube root can be done for specific numbers. Finding cube root of a number is one of the very difficult task one can encounter with. The only method known to us till now is to find the factors of a number, group the factors taking three at a time and take out one number from a group of three[1].

Eg: 1728 = 2*2*2*2*2*2*3*3*3*

Taking out => 2*2*3=12Therefore, $\sqrt[3]{1728} = 12$.

The working method and the name of the sutra is almost the same. Both the square root and cube root can be described as magical method.

The sutra for cube root is 'Vargamula'.

The method for working with perfect cubes is give below [2]:

If the number is a perfect cube, then we can know what the ending digit of the answer will be by looking at the ending digit of the question.

If the number ends in a:

0 -> then the ending digit is a 0

1 -> then the ending digit is 1.

2 -> then the ending digit is 8.

3 -> then the ending digit is 7.

4 -> then the ending digit is 4.

5 -> then the ending digit is 5.

6 -> then the ending digit is 6.

7 -> then the ending digit is 3.

8 -> then the ending digit is 2.

9 -> then the ending digit is 9.

Steps to find the cube root of 9261:

After finding the last digit, chop off the last three digits and focus on the remaining digits.

Ensure that the remaining numbers is less than or equal to the nearest cube of a number.

We know the last number must be a 1.

Chopping off the last three numbers, we need to focus on the remaining numbers or 9.

The nearest cube is '8'; we know that $2^3 = 8$.

So the answer is 21.

F. Vedic Divider

In this paper we report only NND formula to implement the division algorithm and its architecture. "Nikhilam Navatascaramam Dasatah" (NND) is a Sanskrit term indicating "all from 9 and last from 10", formula have been mathematically described for the [10] proposed divider design. Mathematical description of this sutra can be formulated as: Consider two numbers A and B as Dividend and Divisor respectively.

Illustration of NND Sutra

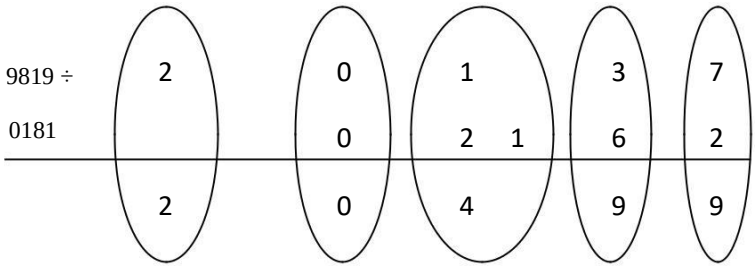

Fig 2: Division using "Nikhilam Navatascaramam Dasatah"

The Chart can be implemented as follows:

Step 1: Assuming Dividend is equal to 20137 and Divisor is equals to 9819. Considering base of operation is equals to 10000. Subtract the divisor from base of operation i.e. equals to 0181.

Step 2: Take the first digit (MSD) of the dividend, put down below the vertical line; here MSD is equal to 2.

Step 3: Multiply the subtraction results with MSD, put down below the dividend. Result is equal to (0181×2=02162). (Here individual digit multiplication has been performed).

Step 4: Perform the addition of the multiplication result with dividend digits. Thus from the above chart our quotient is equal to 2 and remainder is equals to 499. Thus in this division process (by NND formula), we perform only small digit multiplication, without any subtraction and division, quotient and remainder is obtained.

III.HARDWARE IMPLEMENTATION OF ALU

The proposed Arithmetic and logic module has first been split into six smaller modules that is 1.Multiplier 2. Squarer 3.Cube 4.Square root 5.Cube root 6.Division as a whole. These modules have been made using Verilog HDL[3].

Arithmetic Logic Unit can be considered to be the heart of a CPU, as it handles all the mathematical and logical calculations that are needed to be carried out. Again there may be different modules for handling Arithmetic and Logic functions. In this work, an arithmetic unit has been made using Vedic mathematics algorithms and performs Multiplication, Square, Cube, Square root, Cube root, Division operation as well as addition and subtraction. For selection of base, RSU and Exponent determinant have been used. The control signals tell the arithmetic module, when to perform and which operations are provided by the control unit. The individual modules of the Arithmetic unit are Multiplier, Squaring, cubing, square root, cube root and divider blocks. The multiplier block performs multiplication, based on two algorithms namely Urdhva Triyagbhyam and Nikhilam Navatascaramam Dasatach [8].

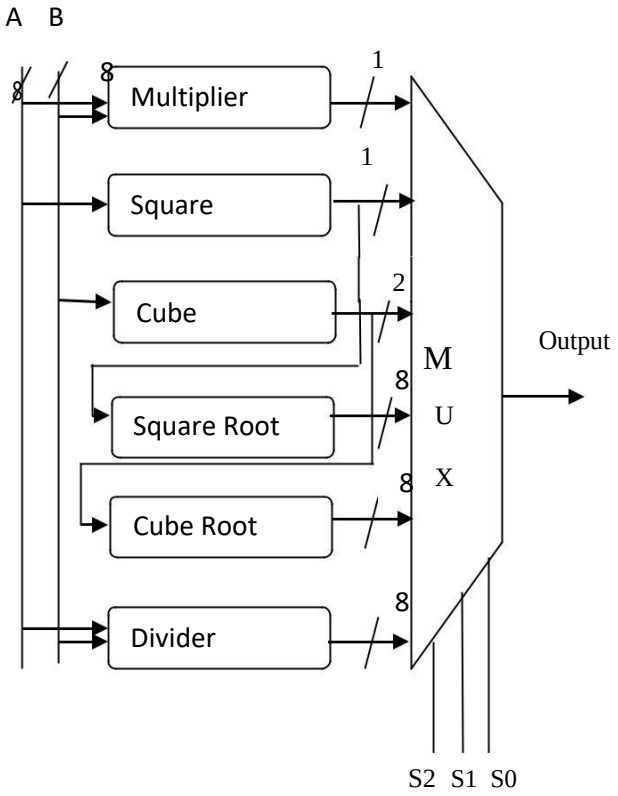

Fig 3: ALU Integration

In Urdhva Tiryagbhyam, multiplication of two 8 bit numbers is performed. Though this sutra holds good for multiplication of binary numbers, the drawback is that when the numbers are large, this algorithm is difficult. To overcome this, Nikhilam sutra is used. Although it is applicable to all cases of multiplication, it is more efficient when numbers involved are large.

The Squarer block is used to perform the squaring operation. When the input is of 8 bit, the output will be 16 bit. The sutra used for this block is Yavadunam. For surplus, Yavadadhikam Tavadahikikritya Vargancha Yojayet and for deficiency, Yavadhunam Tavadahikikritya Vargancha Yojayet is used.

The Cube block is used to perform the cubing operation. When input is of 8 bit, the output will be 32 bit. The algorithm used for this block is Anurupya. Both squaring and cubing block holds good for all numbers.

The Square root block is used to perform the square root operation. When input is of 16 bit, the output will be 8 bit. The algorithm used for this block is Vargamula.

The Cube root block is used to perform the cube root operation. When input is of 32 bit, the output will be 8 bit. Both square root and cube root block holds good for perfect squares and cubes i.e applicable only for specific numbers.

The Divider block is used to perform division operation. The algorithm used for this block is Nikhilam Navatascaramam Dasatah. When the input is of two 8 bit numbers a(dividend) and b(divisor), the output will be single 8 bit number p(quotient).

The Arithmetic module designed in this work, makes use of 6 components, that are, Multiplier, Squarer, Cube, Square root, Cube root, Division. Here the inputs are Data A and Data B, which are 8 bits wide. The arithmetic unit are made using Vedic Mathematics sutras. The control signals which guide the Arithmetic unit to perform a particular operation such as Addition, Subtraction, Multiplication, Square, Cube, Square root, Cube root, Division operation are s0, s1 and s2, which are provided by the control circuit. The operations are performed only when the clock is set to ‘1’.

S2	S1	S0	Operations Performed
0	0	0	Multiplication
0	0	1	Square
0	1	0	Cube
0	1	1	Square root
1	0	0	Cube root
1	0	1	Division
1	1	0	No operation
1	1	1	No operation

Table 1: Operation by mux

A. Hardware Implementation of Multiplier

The architecture can be decomposed into three main subsections, they are:

i) Radix Selection Unit (RSU)

(ii) Exponent Determinant (ED)

(iii) Array multiplier

Radix Selection Unit (RSU)

The RSU is required to select the proper radices corresponding to the input numbers. If the selected radix is nearer to the given number then the multiplication of the residual parts (Z_1*Z_2) can be easier to compute. The Subtractor blocks are required to extract the residual parts (Z_1 and Z_2). The second subsection (ED) is used to extract the power (k_1 and k_2) of the radix and it is followed by a subtractor to calculate the value of (k_1-k_2).The third subsection array multiplier is used to calculate the product (Z_1 *Z_2). The output of the subtractor (k_1-k_2) and Z_2 are fed to the shifter block to calculate the value of Z_2* $2^{k_1-k_2}$.The first adder-subtractor block has been used to calculate the value of $X \pm Z_2 \times 2^{k_1-k_2}$• The output of the first adder-subtractor and the output of the second Exponent Determinant (k_2) are fed to the second shifter block to compute the value of $2^{k_2} * (X \pm Z_2 * 2^{k1-k2})$. The output of the multiplier ($Z_1 \times Z_2$) and the output of the second shifter (2^{k_2}* (X ± Z_2*$2^{k_1-k_2}$))are fed to the second adder/subtractor block to compute the value of ($2^{k_2} * (X \pm Z_2 * 2^{k_1-k_2})) \pm Z_1Z_2$.

Mathematical expression for RSU

Consider an 'n' bit binary number X, and it can be represented [6]

$X = \sum_{i=0}^{n-1} X_i 2^i$ where X_i belongs to {0,1}.Then the values of X must lie in the range $2^{n-1} \leq X < 2^n$.Consider the mean of the range equals to A.

$$A = (2^{n-1}+2^n)/2$$

$$A = 2^{n-2} \times 3$$

If $X > A$ Then radix = 2^n

If $X \leq A$ Then radix = 2^{n-1}

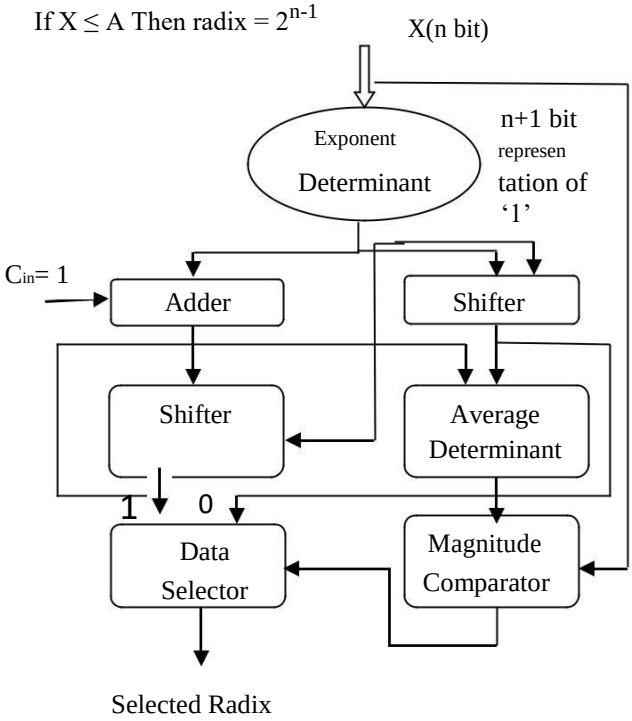

Fig 4: RSU block diagram

N^{th} bit from input X is fed to the ED block. The maximum power of X is extracted at the output which is again fed to shifter and the adder block. The second input to the shifter is the (n+1) bit representation of decimal '1 '.If the maximum power of X from the ED unit is (n-1) then the output of the shifter is$2^{(n-1)}$. The adder unit is needed to increment the value of the maximum power of X by '1'. The second shifter is needed to generate the value of 2^n.Here n is the incremented value taken from the adder block. The Mean Determinant unit is required to compute the mean of ($Z^{n-l} + Z^n$). The Comparator compares the actual input with the mean value of (Z^{n-l} + Z^n). If the input is greater than the mean then 2^n is selected as the required radix. If the input is less than the mean then 2^{n-1} is selected as the radix. The select input to the multiplexer block is taken from the output of the comparator.

Exponent Determinant

The hardware implementation of the exponent determinant is shown in fig 5. The integer part or exponent of the number from the binary fixed point number can be obtained by the maximum power of the radix. For the nonzero input, shifting operation is

executed using parallel in parallel out (PIPO) shift registers. The number of select lines of the PIPO shifter is chosen as per the binary representation of the number $(N-1)_{10}$. 'Shift' pin is assigned in PIPO shifter to check whether the number is to be shifted or not (to initialize the operation 'Shift' pin is initialized to low). A decrementer has been integrated in this architecture to follow the maximum power of the radix. A sequential searching procedure has been implemented here to search the first 'I' starting from the MSB side by using shifting technique.

For an N bit number, the value $(N-I)_{10}$ is fed to the input of decrementer. The decrementer is decremented based on a control signal which is generated by the searched result. If the searched bit is '0' then the control signal becomes low. The decrementer starts decrementing the input value (Here the decrementer is operating in active low logic). The searched bit is used as a controller of the decrementer. When the searched bit is 'I' then the control signal becomes high and the decrementer stops further decrementing and shifter also stops shifting operation [6].

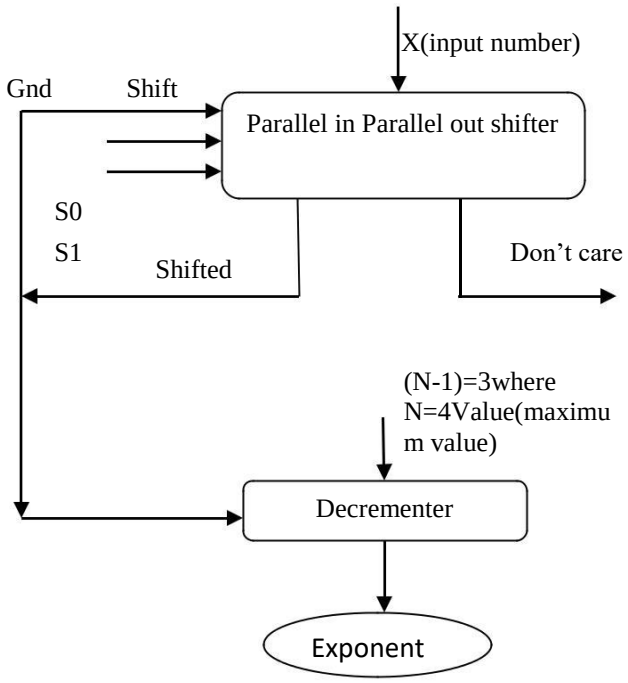

Fig 5: Block diagram of exponent determinant

B. Architecture of Square using Yavadunam

The architecture of squaring algorithm using "Yavadunam Sutra" is shown in fig 6.. The basic building blocks of the architecture are (i) RSU, (ii) Subtractor, (iii) Add-Sub unit and (iv) Duplex squaring architecture.

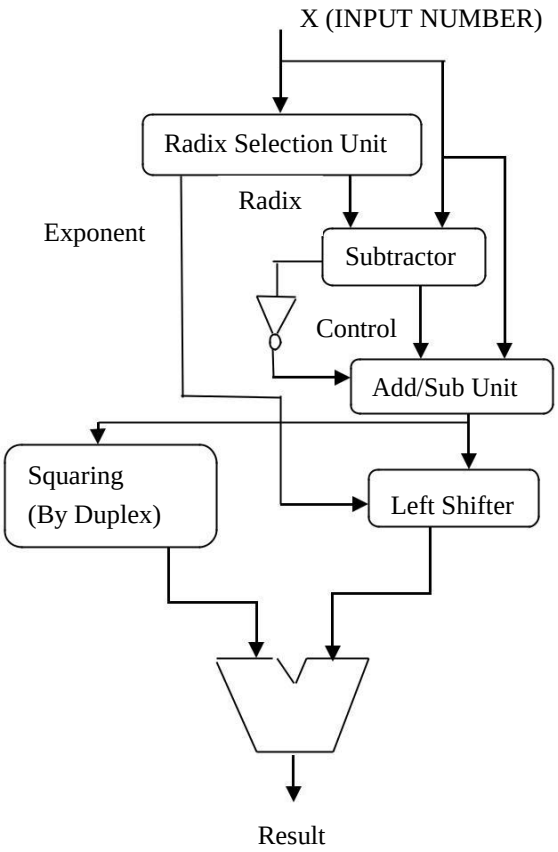

Fig 6: Architecture of squaring algorithm using Yavadunam Sutra

C. Architecture Implementation Divider

The architecture for division using 'Nikhilam' Sutra has been described in Fig. 3. The architecture consists of three major sub-segments :(i) Complement circuitry, (ii) Adder and (iii) Incrementer. The 'n' bit input from divisor is fed to the complement circuitry. Complement methodology that has been used here, is the two's complement method.

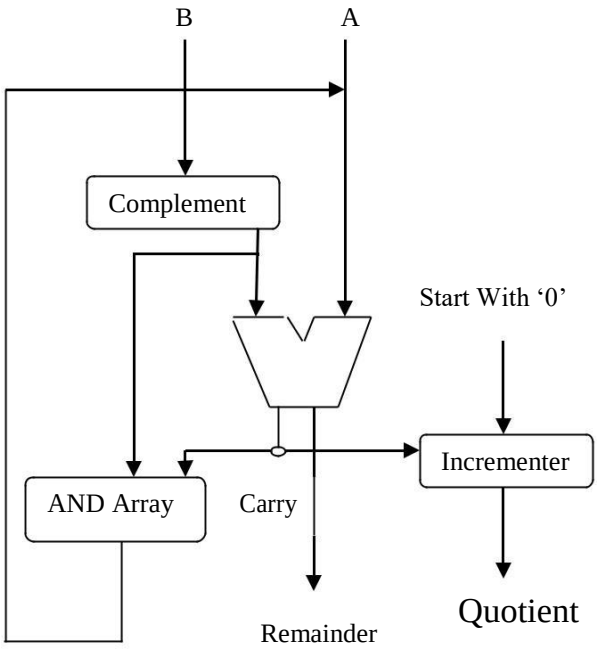

Fig 7: Internal architecture of divider

The result of complement is fed to the adder, the 'Carry' signal generated from the adder is fed to the incrementer as well as AND array. The 'Carry' signal indicates the addition result to be fed to the adder again or not. The incrementer which computes the quotient is controlled by the 'Carry' signal. If the 'Carry' signal is '1', then the output from the adder is again fed to the adder and the operation is repeated again until the result of the incrementer is either n/2 bit or n/2+1 bit. The output from the adder is the actual remainder and the result of the incrementer is the quotient.

In the same way architecture of cube is implemented based on the square architecture. Similarly square root and cube root is also implemented.

IV. PERFORMANCE & DISCUSSION

A. Speed and Delay

Vedic multiplier is faster than conventional multiplier. As the number of bits increases from 8x8 to 16x16, the timing delay is greatly reduced for Vedic multiplier as compared to other multipliers. Vedic multiplier has the greatest advantage as compared to other multipliers over gate delays and regularity of structures. Memory usage of Vedic multiplier is greatly reduced compared to array multiplier

PARAMETERS	2 Bit		4 Bit		8 Bit	
	ARRAY	VEDIC	ARRAY	VEDIC	ARRAY	VEDIC
Speed	4	4	4	4	4	4
Min. I/P arrival	9.3 ns	4.1 ns	12.4 ns	11.7 ns	24.7 ns	21.3 ns
Max. O/P	7.16 ns	7.16 ns	7.16 ns	7.16 ns	7.16 ns	7.16 ns
CPU time	2.66/ 2.80s	2.44/ 2.56 s	2.78/ 2.91s	3.27/ 3.39 s	4.95/ 5.22s	10.38/ 10.69 s
Elapsed time	2.00/ 3.00 s	2.00/ 3.00 s	3.00/ 3.00 s	3.00/ 4.00 s	5.00/ 5.00s	5.00/ 5.00 s
memory usage	141540 KB	14051 6 KB	14154 0 KB	14154 0 KB	24833 2 KB	145916 KB

Table 2.Comparison of Vedic with Array Multiplier

B. LUT Comparison

No of bits	No of 4 i/p LUTs in Conventional Multiplier	No of 4 i/p LUTs in Vedic Squaring Unit
4	32	6
8	186	101
16	880	294

Table 3: Comparison of LUTs

C. Delay

Delay in Vedic squarer for 8 x 8 bit number is 34 ns while the delays in conventional multiplier are 45 ns. Thus the squaring unit using Urdhva shows the highest speed among other squaring techniques. The comparison of the delay is shown below:

No of bits	Delay in Conventional Multiplier	Delay in Vedic Squaring Unit(ns)
4	8.154	4.993
8	45.718	34.947
16	56.657	43.392

Table 4: Comparison of delay

D.Power

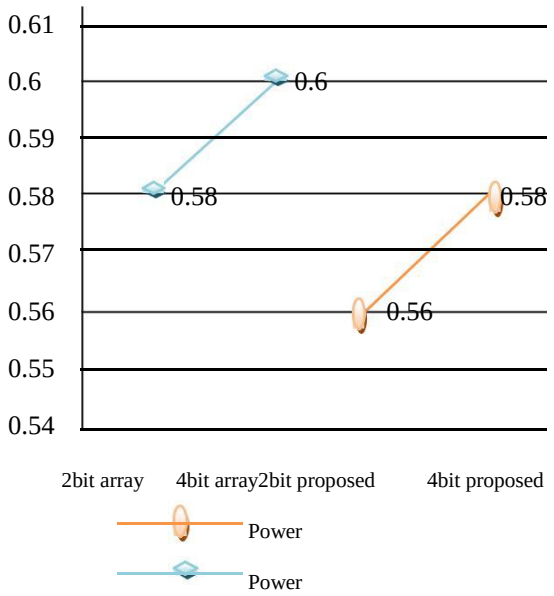

Fig 8: Power comparison graph between array and proposed multiplier

Power consumed by the Vedic multiplier is 0.058W whereas power consumed by array multiplier is 0.060W.

E.Synthesis Report

Device Utilization Summary:

Multiplication -Urdhva Tiryakbhyam

Selected Device	:	3s400tq144-4		
Number of Slices	:	99 out of	3584	2%
Number of 4 input LUTs:	:	174 out of	7168	2%
Number of IOs	:	33		
Number of bonded IOBs	:	32 out of	97	32%

Multiplication –Nikhilam Navatascaramam Dasatah

Selected Device	:	3s400tq144-4		
Number of Slices	:	271 out of	3584	7%
Number of 4 input LUTs:	:	477 out of	7168	6%
Number of IOs	:	33		
Number of bonded IOBs	:	32 out of	97	32%
IOB Flip Flops	:	16		
Number of MULT18X18s:	:	8 out of	16	50%

Square-Yavadunam

Selected Device	:	3s400tq144-4		
Number of Slices	:	65 out of	3584	1%
Number of 4 input LUTs:	:	119 out of	7168	1%
Number of IOs	:	49		
Number of bonded IOBs	:	48 out of	97	49%
Number of MULT18X18s:	:	3 out of	16	18%

Cube-Anurupya

Selected Device	:	3s400tq144-4		
Number of Slices	:	4145 out of	3584	115%
Number of 4 input LUTs:	:	7546 out of	7168	105%
Number of IOs	:	33		
Number of bonded IOBs	:	32 out of	97	32%
Number of MULT18X18s:	:	12 out of	16	75%

V. CONCLUSION

The proposed ALU is proved to be efficient than conventional ALU in terms of area of consumption and delay. Each proposed module uses different Sutras. The delay in squarer block is 34ns which is less than the delay caused by the conventional techniques. The memory usage of Vedic multiplier is 145916 KB which is smaller than 248332 KB of Array multiplier. From RTL schematic we can observe that regular structure is obtained and also it makes routing so easy. Integrated ALU leads to Vedic coprocessor which increases the efficiency of multiprocessor configuration system design.

The future scope of this work is to check whether Vedic maths is applicable for other operations like trigonometric functions, logarithmic function and floating point functions etc., & to calculate delay, power when the bit size is increased

REFERENCES

[1]. Swami Bharati Krishna Tirthaji , Vedic Mathematics. Delhi: Motilal Banarsidass Publishers, 1965.

[2]. Vedic Mathematics [Online]. Available: www.hinduism.co.za/vedic.htm. Accessed November7.

[3]. M. Ramalatha, K. Deena Dayalan, P. Dharani, "High Speed Energy Efficient ALU Design using Vedic Multiplication Techniques" ACTEA, IEEE pp 600-603

[4]. Kabiraj Sethi, Rutuparna Panda"An Improved squaring circuit for Binary Numbers"International Journal of Advanced Computer Science and Application" Vol3.No2, 2012

[5]. P.Saha, A.Banerjee, A.Dandapat, P.Bhattacharyya"Vedic Mathematics Based 32 Bit Multiplier Design for High Speed

[6]. Low Power Processors"International Journal on Smart Sensing and Intelligent Systems Vol.4, NO.2, JUNE2011

[7]. P. Sreenivasa Rao, Mr.C.Md.Aslam "Design Of Complex Multiplier with High Speed ASIC using Vedic Mathematics" International Journal Of Engineering Research And Technology (Ijert) Issn: 2278-0181 Vol.1 Issue 6, August-2012

[8]. S.S.Kerur, Prakash marchi, Jayashree CN, Harish M Kittur and Girish V.A. "Implementation Of Vedic Multiplier For Digital Signal Processing" International Conference On Vlsi, Communication And Instrumentation (ICVCI),2011 Proceeding Published By International Journal Of Computer Application(IJCA)

[9]. Mr.Abhishekgupta, Mr.Utsavmalviya, Prof.Vinod Kapse"Novel Approach to Design High Speed ALU Based On Ancient Vedic Multiplication Technique" International Journal Of Modern Engineering Research Vol.2 Issue.4, July-August 2012 Pp-2695-2698

[10]. M.E.Paramasivam, Dr.R.S.Sabeenian,"An Efficient Bit Reduction Binary Multiplication Algorithm using Vedic Methods" 2010 IEEE 2nd International Advance Computing Conference.

[11]. Prabir Saha*, Arindam Banerjee**, Partha Bhattacharyya*, Anup Dandapat," Vedic Divider: Novel Architecture (ASIC) for High Speed VLSI Applications"2011Internaional symposium on electronic system design.

[12]. Bathija, R.K., Meena, R.S., Sarkar, S., Sahu, Rajesh. : "Low Power High speed 16X16 bit Multiplier using Vedic Mathematics," International Journal of Computer Applications(IJCA), Vol. 59 -Number 6, December2012

[13]. Eganathan Sriskandarajah, "Secrets of Ancient Maths: Vedic Mathematics", Journal of Indic Studies Foundation, California, pages 15 and 16

[14]. Gupta, A., Malviya, U, Kapse, V.: "Design of Speed, Energy,andpower efficient Reversible logic based ALU for digital processors," IEEE Proc. NUiCONE, Ahmedabad, 6-8 Dec 2012, pp. 1-6.

[15]. Hanumantharaju,M.C., Jayalaxmi, H., Renuka R.K., and Ravishankar, M.: "A High-Speed Block Convolution Using Ancient Indian Vedic Mathematics, " IEEE International Conference on Computational Intelligence and Multimedia Applications,Sivakasi, Tamil Nadu, 13-15 Dec 2007, pp.169-173.

[16]. Huddar, S.R., Rupanagudi, S.R., M., Mohan, S.: "Novel high-speedVedic mathematics multiplier using compressors," IEEE International multiConference, 2013, pp.465-469

[17]. Madhu Latha B, B. Nageswar Rao, "Design and Implementation of High-Speed 8-Bit Vedic Multiplier on FPGA" International Journal of Advanced Research in Electrical, Electronics and instrumentation engineering, Vol. 3, Issue 8, August 2014.

[18]. Murali A, G Vijaya Padma, T Saritha, "An Optimized Implementation of Vedic Multiplier Using Barrel Shifter in FPGA Technology", Journal of Innovative Engineering 2014, 2(2).

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

[20]. Tiwari, H.D., Gankhuyag, G., Kim, M.,and Cho, B.: "Multiplier design based on ancient Indian Vedic Mathematics," IEEE Proc. International SoC Design Conference, ISOCC, Busan, 2008, pp. II-65 - II-68.

[21]. Toni J.Billore, D.R.Rotake, "FPGA implementation of high speed 8-bit Vedic Multiplier using Fastadders" Journal of VLSI and Signal Processing Volume 4, Issue 3, Ver. II (May- Jun. 2014), PP 54-59 e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197

[22]. Wallace, C.S., "A suggestion for a fast multiplier," IEEE Trans. Elec. Comput.,vol. EC- 13, no. 1, pp. 14–17, Feb. 1964.