DESIGN AND IMPLEMENTATION OF FLOATING POINT MULTIPLIER AND SQUARE ROOT OF A NUMBER BY VEDIC MATHEMATICS USING VHDL

1P. S. K. ROHIT VARMA, 2PRATHIK S.M., 3R. ROHIT, 4ANANDILKAL

IV year, B.E. Electronics and Communication department, S. D. M. college of Engineering and Technology Dharwad, Karnataka India
Email: varmarohit75@gmail.com, a.r.rohit9999@gmail.com

Abstract: Vedic Mathematics deals mainly with various Vedic mathematical formulae and their application for carrying out tedious and cumbersome arithmetical operations. The sutras (aphorisms) apply to and cover each and every part of each and every branch of mathematics. The paper explains about UrdhvaTiryaka and vargamula sutras. UrdhvaTiryaka sutra deals with the multiplication of two numbers and vargamula sutra deals with the square root of a number. These Sutras are used for multiplication of two 16-bit precision floating point numbers and square root of 16-bit number respectively. The results are observed on the Xilinx ISE simulator using VHDL codes.

Keywords: Vedic Mathematics, Urdhvatiryaka And Vargamula Sutras, Floating Point Multiplication, Square Root.

I. INTRODUCTION

The word ‘Veda’ has this derivational meaning, i.e. the fountain-head and illimitable store-house of knowledge. The work Vedic Mathematics or ‘Sixteen simple Mathematical formula from the Vedas’, was written by His Holiness Jagadguru Sankaracharya Sri Bharathi Krsna Tirthaji Maharaja of Govardhana Matha, Puri (1884-1960). The Vedas are ancient scriptural texts of India written in Sanskrit. They deal with many subjects but the texts are very difficult to understand. Tirthaji made great efforts to dig out the system of mathematics from these texts and came up with sixteen sutras and about thirteen sub-sutras. It forms a class by itself not pragmatically conceived and worked out as in the case of other scientific works, but is the result of the intuitional visualization of fundamental mathematical truths and principles during the course of eight years of highly concentrated mental endeavour.

Vedic Mathematics deals mainly with various vedic Mathematical formulae and their applications for carrying out tedious and cumbersome arithmetic operation. The methods which are used to solve the operations are very unconventional. The sutras, provide easily remembered word-formulae for solving problems in arithmetic, algebra, geometry and their various applications. The methods are very fast and effective. One of the important sutras in Vedic Maths is UrdhvaTiryaka Sutra. This sutra helps to perform multiplication operation. This method is also called as ‘VERTICAL AND CROSS-WISE METHOD’. Floating point multiplication will be carried out using this sutra.

Step 1: First we multiply the unit digit of both numbers, we retain only one digit and remaining digits are forwarded as carry, which gives the right most part of the answer.

Step 2: We then get the sum of products of the ten digit of 1st number to the unit digit of 2nd number and ten digit of 2nd number to the unit digit of 1st number, along with the preceding carry(if any), which gives the middle part of the answer.

Step 3: Finally multiply the left hand-most digit of the multiplicand vertically by the left hand-most digit of multiplier to get their product, along with the preceding carry and set it down as the left hand-most part of the answer.

The proof for the above explained sutra is explained by multiplication of (ax+b) by (cx+d), where x is 10 and the product is acx²+x(ad+bc)+bd.

Example:

1. To multiply 12 by 13

The numbers are less than 100 so they can be expressed as linear equation. 12 = 1x+2
13 = 1x+3

By Sutra: [(1*1)*100]+[(1*3)+(1*2)] 10+(2*3) = 156

By Algebra: Product after multiplying both expressions is x²+5x+6. Substitute x as 10 and the result is 100+50+6 = 156. This multiplication can be done even for binary numbers as shown below.

i) Algorithm for UrdhvaTiryaka (Vertical and cross-wise method):

Design And Implementation Of Floating Point Multiplier And Square Root Of A Number By Vedic Mathematics Using VHDL
ii) Floating point representation
In 32-bit single-precision floating-point representation:
- The most significant bit is the sign bit (S), with 0 for negative numbers and 1 for positive numbers.
- The following 8 bits represent exponent (E).
- The remaining 23 bits represent fraction (F).

Due to some limitations, the fraction part has been reduced to 7 bits thus reducing the entire number of bits at the output to 16 bits.

The bit representation will be converted into decimal form using the formula below:
For \(1 \leq E \leq 254\)
\[
N = (-1)^S \times 1.F \times 2^{(E - 127)}
\]
These numbers are in the so-called normalized form. The sign-bit represents the sign of the number. Fractional part (1.F) are normalized with an implicit leading 1. The exponent is bias (or in excess) of 127, so as to represent both positive and negative exponent. The range of exponent is -126 to +127.
The weight of the fractional part reduces as \(2^{-x}\).

Suppose that 16-bit floating-point representation pattern is

\[
101111110 1000000
\]
Sign bit \(S = 1 \Rightarrow\) negative number
\(E = 0111 1110B = 126D\) (in normalized form)
Fraction is 1000000B (with an implicit leading 1)
\(N = -1 \times (1+0.5) \times 2^6(-1) = -0.75\)
In decimal we are performing the following addition: \( 131 + 129 - 127 = 133 = 10000101 \)

Step 2. Multiply the significands with sign bit as MSB. If sign is 0 then number is positive or else negative.

Step 3. Normalize the product shifting (right) if necessary.

The result will be 0 10000110 00110000000000000000 which is 155.2.

B) Square root of a 16-bit number:

Before going to the algorithm, the concept of duplex should be understood.

- Duplex of a single digit number (say as a) will be \( a^2 \).
- Duplex of a two digit number, (say ab) will be 2ab.
- Duplex of a three number (say as abc) will be 2ac+b^2 and so on.

The algorithm for the square root of a number is:
1. The given number is first arranged in two-digit groups from right to left: and a single digit if any left over at the left-hand-end is counted as a single group by itself.
2. The number of digits in the square root will be the same as the number if digit-groups in the given number itself including a single digit if any such there be. Thus, 16 counts as one group, 144 as two groups and 1024 also as two.
3. Once we divided them into groups, the leftmost digit's square root is taken and its decimal part is truncated (In example for 6, square root will be taken as 2).
4. Multiply the square root with 2, which will be acting as divisor (2 multiplied by 2 will be 4, which is the divisor).
5. Now subtract 6 from the duplex of 2. We get 2 and concatenating it with the next number 5 in 65025 and divide it by the divisor (25/4 \( \Rightarrow \) q=6 r=1). The quotient will be the next digit of the output.
6. The remainder is appended with 0 (from 65025) and we get 10. Now subtract 10 from the duplex 6. As it gives us a negative result, we move back to 5 and reduce the quotient to 5 and increase the remainder to 5(1+divisor).
8. Repeat from Step 5 until all the digits in the number are done. Then the resultant is the output which contains both decimal and fraction part.

Example:

\[
\begin{array}{cccc}
6 & 5 & 0 & 2 \\
(4) & & & \\
2 & 5 & 5 & 2
\end{array}
\]

The square root of 65025 is 255.00

III. RESULTS

Figure 5 shows the results of multiplication of two floating point numbers 0.75 and 2.625. 
0.75 is represented as 0 01111110 1000000000000000000000
2.625 is represented as 0 10000000 010100000000000000000
Result obtained is 0 01111111 11110000000000000000000000000000
Which is 1.96875

Fig. 5. Floating point multiplication of two numbers using Xilinx ISE simulator.

The combinational delay from the input and the output is only 32ns.

Figure 6 shows the result of the square root of a 16-bit number 1556.

Fig. 6. Square root of a 16-bit number in VHDL using Xilinx ISE simulator.

The result is 39.5 where 39 is showed in one register and 5 is showed in another.

REFERENCES

[1]. IEEE paper on 'Low power ALU design using Ancient Mathematics' by Anvesh Kumar and Ashish Raman.
[5]. 'Digital electronics and design with VHDL' by volnei A. Pedroni