IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 1, January 2015.
www.ijiset.com
ISSN 2348 – 7968

Design of Floating Point Multiplier Using Vedic Mathematics

[1]Mr. S.S.Mohanasundaram, A.Nirmal kumar, T.Arul prakash

[1]Assistant Professor,
Dept. of Electronics and Communication Engineering, P.A. College of Engineering and Tech

ABSTRACT

A conventional way of performing multiplication between two 8 bit numbers can be outstand by adopting Vedic mathematics. Vedic mathematics is the ancient system of mathematics which has a unique technique of calculations based on sixteen sutras. Vedic mathematics which use certain prescribed sutras like Urudhva-Tiryagbhyam, Chalana-Kalanabhyam, Paraavartya Yojayet is analysed to be a convinent and efficient method because it reduces the processing delay and saves the time. Eventhough it increases design complexity, it is traded off by providing decreased computational area, functionality and thence power consumption. On such, a single precision floating point multiplication using Vedic mathematics is coded using VHDL and implemented in SPARTAN 3E FPGA using Xilinx ISE 9.1.

Keywords

Vedic Mathematics, Urudhva-Tiryagbhyam algorithm, Chalana algorithm, Paraavartya algorithm, Floating point multiplier, VHDL.

1. INTRODUCTION

1.1 Vedic Mathematics

Vedic Mathematics-a gift given to this world by the ancient sages of India. A system which is far simpler and more enjoyable than Modern mathematics. Vedic Mathematics refers to a technique of calculations based on a set of 16 sutras or aphorisms as algorithm. In which every individual can develop their own methods to do the calculations when compared with the modern mathematics and can develop their knowledge.

Vedic formula

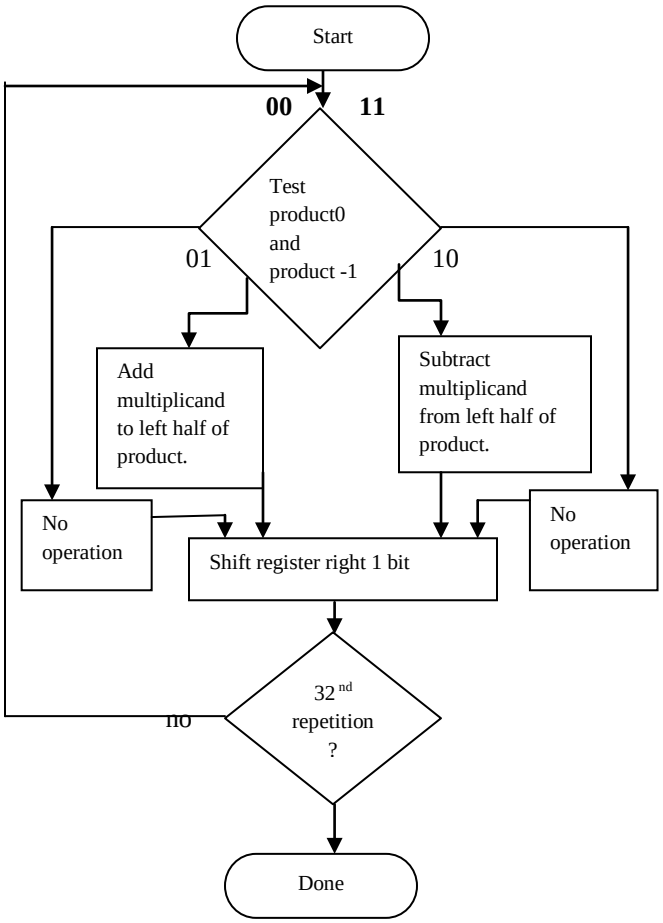
- (Anurupye) Shunyamanyat - If one is in ratio, the other is zero.
- Chalana – Kalanabhyam - Differences and similarities.
- Ekhadhikina Purvena - By one more than the previous one
- Ekanyunena Purvena - By one less than the previous one.
- Gunakasamuchyah -The factor of the sum is equal to the sum of the factors.
- Gunitasamuchyah - The Product of the sum is equal to the sum of the product.
- Nikhikam Navatashcaramam Dashatah - All from 9 and the last from 10.
- Paraavartya Yojayet - Transpose and adjust.
- Puranapuranabhyam – By the completion or non completion .
- Sankalana-Vyavakalanabhyam - By addition and by subtraction.
- Shesanyankena Charamena - The remainders by the last digit.
- Shunyam Saamyasamuccaye - When the sum is the same that sum is zero.
- Sopaantyadvayamantyam - The ultimate and twice.
- Urudhva- Tiryagbhyam - Vertically and Crosswise
- Vyashtisamansith - Part and Whole.
- Yaavadunam - Whatever the extent of its deficiency.

2. EXISTING DESIGN

Booth's algorithm is one of the multiplication algorithms, which treats both positive and negative number uniformly. It is a powerful algorithm for signed number multiplication which generates 2n-bit product and treats both positive and negative number uniformly. We can reduce the number of operations required for multiplication by representing multiplier as a difference between two

ISSN 2348 – 7968

numbers. Booth multiplier for type-2 normal basis of GF(2^m) saves approximately 10% space complexity as compared to related parallel multipliers.



Booth's algorithm uses an extra bit on the right of the least significant bit in the product register. This bit starts out as 0 because:

1) if bit 0 is a 1, we want to subtract the multiplicand from the product register
2) if bit 0 is a 0, we do not want to perform any operations

When the shift in step 2 in the diagram is performed, the rightmost (least significant) bit in the product register is placed into this extra bit.Why does Booth's algorithm work for negative numbers? To see why, we need to rewrite a multiplication into the steps that Booth's algorithm operates on. Booth's algorithm looks at adjacent pairs of bits. Suppose that a_i is bit i of the multiplier and b_i is bit i of the multiplicand. The following table shows what Booth's algorithm would do:

0010_{two}
x 0111_{two}

\+ 0010 multiplicand shifted by 0 bits left
\+ 0010 multiplicand shifted by 1 bit left
\+ 0010 multiplicand shifted by 2 bits left
\+ 0000

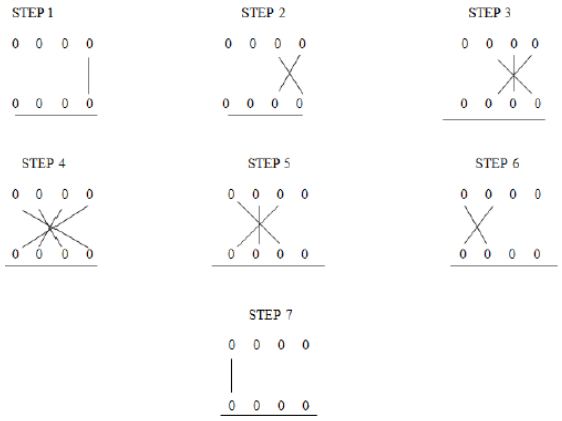
00001110_{two}

We can rewrite $2^i - 2^{i-j}$ as:

$$2^i - 2^{i-j} = 2^{i-j} \times (2^j - 1)$$
$$= 2^{i-j} \times (2^{j-1} + 2^{j-2} + \ldots + 2^0)$$
$$= 2^{i-1} + 2^{i-2} + \ldots + 2^{i-j}$$

3. PROPOSED DESIGN

a) URUDHVA-TIRYAGBHYAM SUTRA



The proposed Vedic multiplier is based on the "Urudhva-Tiryagbhyam". These sutras have been traditionally used for the multiplication of two numbers in the decimal number system. It is a general multiplication formula applicable to all cases of multiplication; it literally means "Vertically and crosswise". The multiplier based on this sutra has the advantage that as the number of bit increases, gate delay and area increases very slowly as compared to the other conventional multipliers. To illustrate the multiplication algorithm, consider the multiplication of two binary numbers a3a2a1a0 and b3b2b1b0. The result of this multiplication would be more than 4 bits and is expressed as r3r2r1r0.

ISSN 2348 – 7968

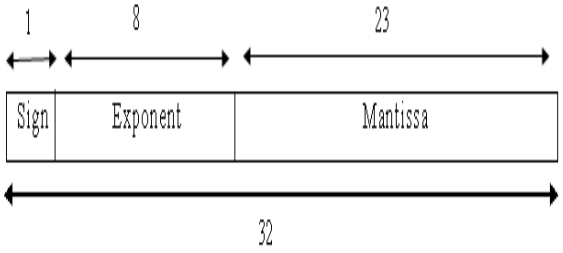
Thus the following expressions are obtained:
r0=a0b0; (1)
c1r1=a1b0+a0b1; (2)
c2r2=c1+a2b0+a1b1 + a0b2; (3)
c3r3=c2+a3b0+a2b1 + a1b2 + a0b3; (4)
c4r4=c3+a3b1+a2b2 + a1b3; (5)
c5r5=c4+a3b2+a2b3; (6)
c6r6=c5+a3b3 (7)

b) FLOATING POINT MULTIPLIER ARITHMETIC

The IEEE (Institute of Electrical and Electronics Engineers) has produced a Standard to define floating-point representation and arithmetic. The standard brought out by the IEEE come to be known as IEEE 754. The IEEE 754 Standard for Floating-Point Arithmetic is the most widely-used standard for floating-point computation, and is followed by many hardware (CPU and FPU) and software implementations. Many computer languages allow or require that some or all arithmetic be carried out using IEEE 754 formats and operations. The standard specifies:

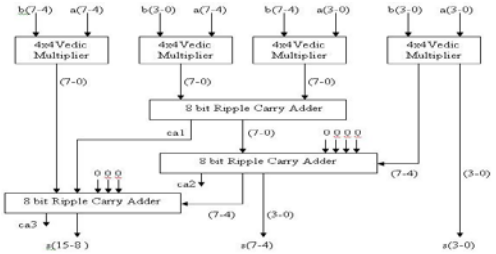
- Basic and extended floating-point number formats
- Add, subtract, multiply, divide, square root, remainder, and compare operations
- Conversions between integer and floating-point formats
- Conversions between different floating-point formats
- Conversions between basic format floating-point numbers and decimal strings
- Floating-point exceptions and their handling, including non- numbers

There are basically two binary floating-point formats. These formats are the Single Precision and Double precision. In this project Single Precision
format is used,
The single precision number has 3 main fields that is sign field, exponent field and mantissa field as shown in Fig . Thus a total of 32-bits are required for single precision number representation. To achieve a bias, $2_{n-1} - 1$ is added to the actual exponent in order to obtain the stored exponent. This equals to 127 for an eight-bit exponent of the single-precision format. The addition of bias allows the use of an exponent in the range from -127 to +128 corresponding to a range of 0-255 for single precision number. The single-precision format offers a range from 2^{-127} to 2^{+127}.
Sign: 1-bit wide and used to denote the sign of the number i.e., 0 indicate positive number and 1 represent negative number.
Exponent: 8-bit wide and signed exponent in excess -127 representations.
Mantissa: 23-bit wide and fractional component.

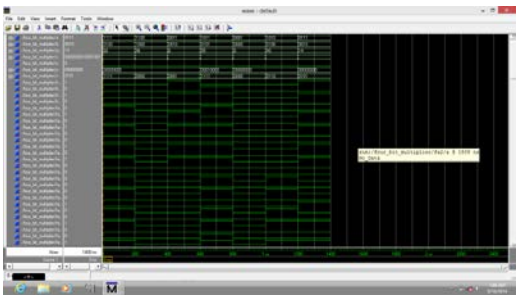


c) PROPOSED MULTIPLIER ARCHITECTURE

The 8x8 bit Vedic multiplier module as shown in the block diagram in Fig. It can be easily implemented by using four 4x4 bit Vedic multiplier modules as discussed in the previous section. Let's analyze 8x8 multiplications, say A= A7 A6 A5 A4 A3 A2 A1 A0 and B= B7 B6 B5B4 B3 B2 B1B0. The output line for the multiplication result will be of 16 bits as – S15
S14 S13 S12 S11 S10 S9 S8 S7 S6S5S4 S3 S2 S1 S0.
Let's divide A and B into two parts, say the 8 bit multiplicand A can be decomposed into pair of 4 bits AH-AL. Similarly multiplicand B can be decomposed
into BH-BL. The 16 bit product can be written as
P = A x B = (AH-AL) x (BH-BL) = AH x BH + (AH x BL + AL x BH) + AL x BL.



4. RESULT



ISSN 2348 – 7968

5. FUTURE WORK

This method is to be implemented in Xilinx software to measure the performance analysis. Vedic mathematics can be implemented in the FFT algorithm for reducing the number of computations.

6. REFERENCES

1) Anvesh Kumar, Ashish raman,"Low Power, High Speed ALU Design by Vedic Mathematics "publish in National conference organized by NIT, hamirpur, [2009].
2) Aniruddha Kanhe, Shishir Kumar Das and Ankit Kumar Singh, "Design and Implementation of Low Power Multiplier Using Vedic Multiplication Technique", (IJCSC) International Journal of Computer Science and Communication Vol. 3, No. 1, pp. 131-132, January-June[2012].
3) Prabha S., Kasliwal, B.P. Patil and D.K. Gautam,"Performance Evaluation of Squaring Operation by Vedic Mathematics", IETE Journal of Research, vol.57, Issue 1, Jan-Feb [2011].
4) Pushpalata Verma, K. K. Mehta, "Implementation of an Efficient Multiplier based on Vedic mathematics Using EDA Tool"International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-1, Issue-5, June [2012].
5) Poornima M, Shivaraj Kumar Patil, Shivukumar , Shridhar K P , Sanjay H "Implementation of Multiplier using Vedic Algorithm "International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-2, Issue-6, May [2013] Note that the journal title, volume number and issue number are set in italics.
6) S. S. Kerur, Prakash Narchi, Jayashree C N, Harish M Kittur, Girish V A, "Implementation of Vedic Multiplier for Digital Signal Processing," International Journal of Computer Applications (IJCA) [2011].
7) Anna Jain, Baisakhy Dash, Ajit Kumar Pande, Muchharla, "FPGA Design of a Fast 32-bit Floating Point Multiplier Unit", Proc of 2012 International Conference on Devices, Circuits and Systems(ICDCS), 15-16 March [2012], pp 545 – 547.
8) Kusuma R, Kavitha V," Performance Analysis of 48-bit Prefix Tree Adders," Proc of 2013 ICECE, 24th April [2013], pp 17 - 21.
9) BHAGYASHREE HARDIYA et al "IMPLEMENTATION OF FLOATING POINT MULTIPLIER USING VHDL" Technology and Engineering (BEST: IJMITE) Vol. 1, Issue 3, Dec [2013], 199-204 © BEST Journals.