iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** I **Month of publication:** January 2022

DOI: https://doi.org/10.22214/ijraset.2022.39742

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue I Jan 2022- Available at www.ijraset.com

Design and Implementation of Floating-Point Addition and Floating-Point Multiplication

Nagireddy Kavya
PG Student, Department of Electronics and Communication Engineering, VLSI and Embedded Systems, University College of Engineering Kakinada (Autonomous) - 533003, Andhra Pradesh, India

Abstract: ***In this paper, we present the design and implementation of Floating point addition and Floating point Multiplication. There are many multipliers in existence in which Floating point Multiplication and Floating point addition offers a high precision and more accuracy for the data representation of the image. This project is designed and simulated on Xilinx ISE 14.7 version software using verilog. Simulation results show area reduction and delay reduction as compared to the conventional method.***

Keywords: FIR Filter, Floating point Addition, Floating point Multiplication, Carry Look Ahead Adder

I. INTRODUCTION

In computing, floating-point arithmetic (FP) is arithmetic using formulaic representation of real numbers as an approximation to support a trade-off between range and precision. For this reason, floating-point computation is often used in systems with very small and very large real numbers that require fast processing times. In general, a floating-point number is represented approximately with a fixed number of significant digits (the significand) and scaled using an exponent in some fixed base; the base for the scaling is normally two, ten, or sixteen. A number that can be represented exactly is of the following form:

$$\text{Siginificand} \times \text{base}^{\text{exponent}}$$

Here significand is an integer, base is an integer greater than or equal to two, and exponent is also an integer. For example:

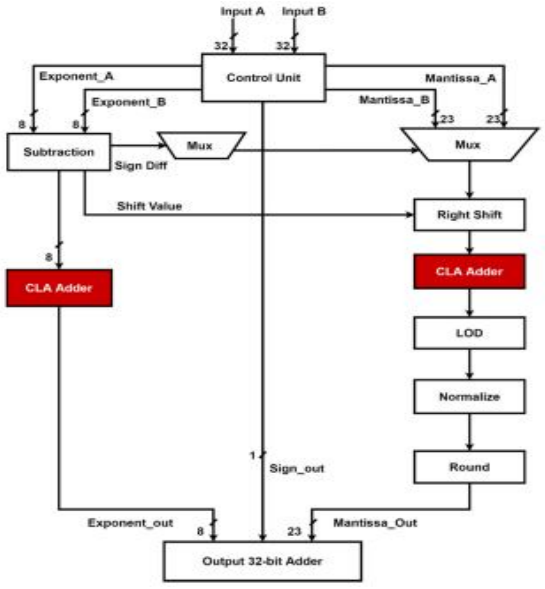
$$1.2345 = 12345 \times 10^{-4}$$

II. FLOATING POINT ADDITION

Contrasted with a fixed point addition , a floating point addition is more complicated and equipment consuming. This is on the grounds that type field is absent if there should be an occurrence of fixed point arithmetic. A floating point addition of two numbers and can be communicated as

Sa.Ma.2Ea + Sb.Mb.2Eb =S.2Eb(Ma+Mb*)

A. Flow Chart



B. Procedure

Floating point addition algorithm

$$X3=X1+X2$$

$$X3 = (M1 \times 2E1) + (M2 \times 2E2)$$

1) X1 and X2 must be added if the exponents are the equivalent i.e E1=E2.

2) We accept that X1 has the bigger absolute value of the 2 numbers. Absolute value of X1 must be equal to o X2,otherwise swap those values .

$$Abs(X1)>Abs(X2).$$

3) Initial value of the exponent must to be the bigger of the 2 numbers, since we know exponent of X1 will be bigger,hence Initial exponent result E3 = E1.

4) Calculate the difference of those exponents i.e.Exp_diff=(E1-E2).

5) After calculating the exponent difference left shift the decimal point of mantissa (M2) .now we having both exponents of X1 and X2 are same.

6) Depending on the sign bit S1 and S2 now add the mantissas.

if both signs are equal then add mantissa (S1==S2)

if not equal then subtract (S1!=S2)

7) Normalize the resultant mantissa (M3) if necessary. (1.m3 arrangement) and the initial exponent result E3=E1 should be changed by the normalization of mantissa.

8) if (E3>Emax) then overflow occurred ,if (E3<Emin) then underflow occurred, then output is to be zero

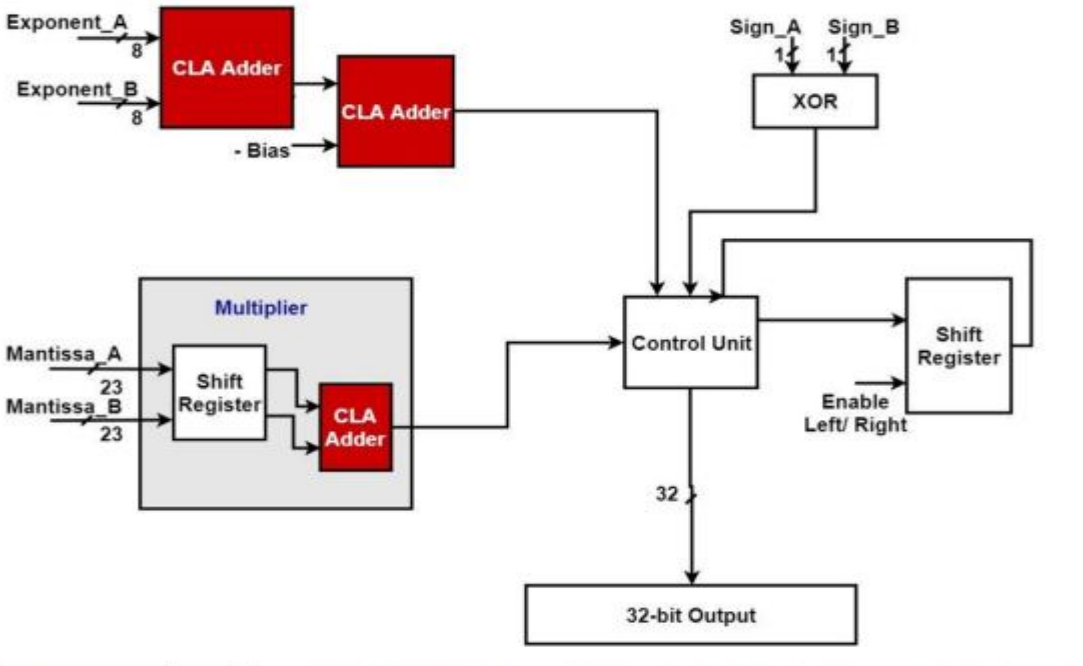
9) Nan's are not upheld.

III.FLOATING POINT MULTIPLICATION

Floating point multiplication is comparatively easy than the floating point addition algorithm but off course consumes more hardware than fixed point multiplier circuit. Major hardware block is the multiplier which is same as fixed point multiplier. This multiplier is used to multiply the mantissas of the two numbers. A floating point multiplication between two numbers a and b can be expressed as

$$S_b \cdot M_b \, 2^{E_b} \times S_a \cdot M_a \, 2^{E_a} = S_a \oplus S_b \cdot M\,b \times Ma \cdot 2^{(Eb+Ea)-bias}$$

A. Flow Chart



B. Procedure

Explanation of floating point algorithm has explained

Result X3 = X1 * X2

$= (-1)^{s1} (M1 \times 2^{E1}) * (-1)^{s2} (M2 \times 2^{E2})$

S1, S2 => Sign bits of X1 & X2.

E1, E2: =>Exponent bits of X1 & X2.

M1, M2 =>Mantissa bits of X1 & X2.

1) Check in the event that one/the two operands = 0 or infinity. Set the output to 0 or infinty. for example exponent = all 0 or all 1

2) S1 is XOR with the S2 where S1 is sign bit multiplicand and S2 is the sign bit of multiplier.

3) The mantissa of M1 and M2 are multiplicated where.M1 is the multiplier and M2 is multiplicand and the output is placed in resultant field of mantissa.

=M1 * M2

4) The exponents of the M1 is (E1) and M2 is (E2) bits are added and the base value is subtracted from the output. That result is placed in the exponential field of the output block.

=E1+E2-bias

5) Now normalize the sum, By shifting right and increment the exponent or shifting left and decrement the exponent.

6) Now check the underflow/overflow, if it is underflow set output is zero and if it is overflow output is infinity.

7) If (E1 + E2 -bias) >= to Emax then, at that point, set the product to infinity.

8) If E1 + E2 - bias) is lesser than/equivalent to Emin then, at that point, set output to zero.

IV. RESULTS

This project is implemented on Xilinx software 14.7 version using veilog language, Spartan 5 Vertex device family, XC6SLX100T Device and the package used is FGG900 with a speed grade of "-3".

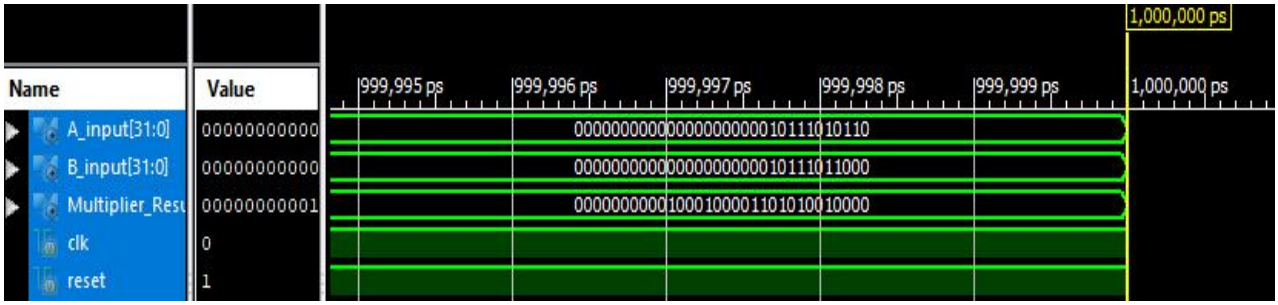


Fig 1.Test bench results for floating point multiplication

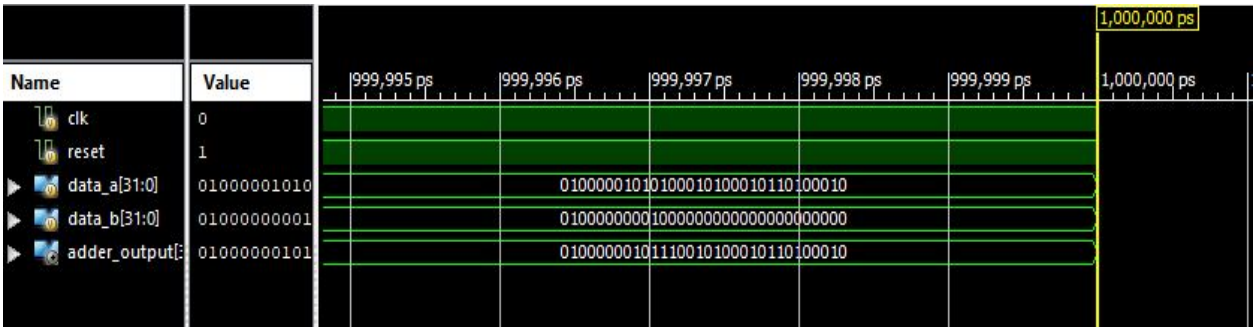


Fig2. Test bench results for floating point addition

V. CONCLUSION

In this brief, by the design and implementation of Floating point addition and Multiplication , it has been proved that the speed of multiplication addition can be increased and also it requires less area compared to the conventional method. here the Carry Look Ahead Adder is used to increase the speed. Thus this proposed method can be used for less area and high-speed digital image processing applications.

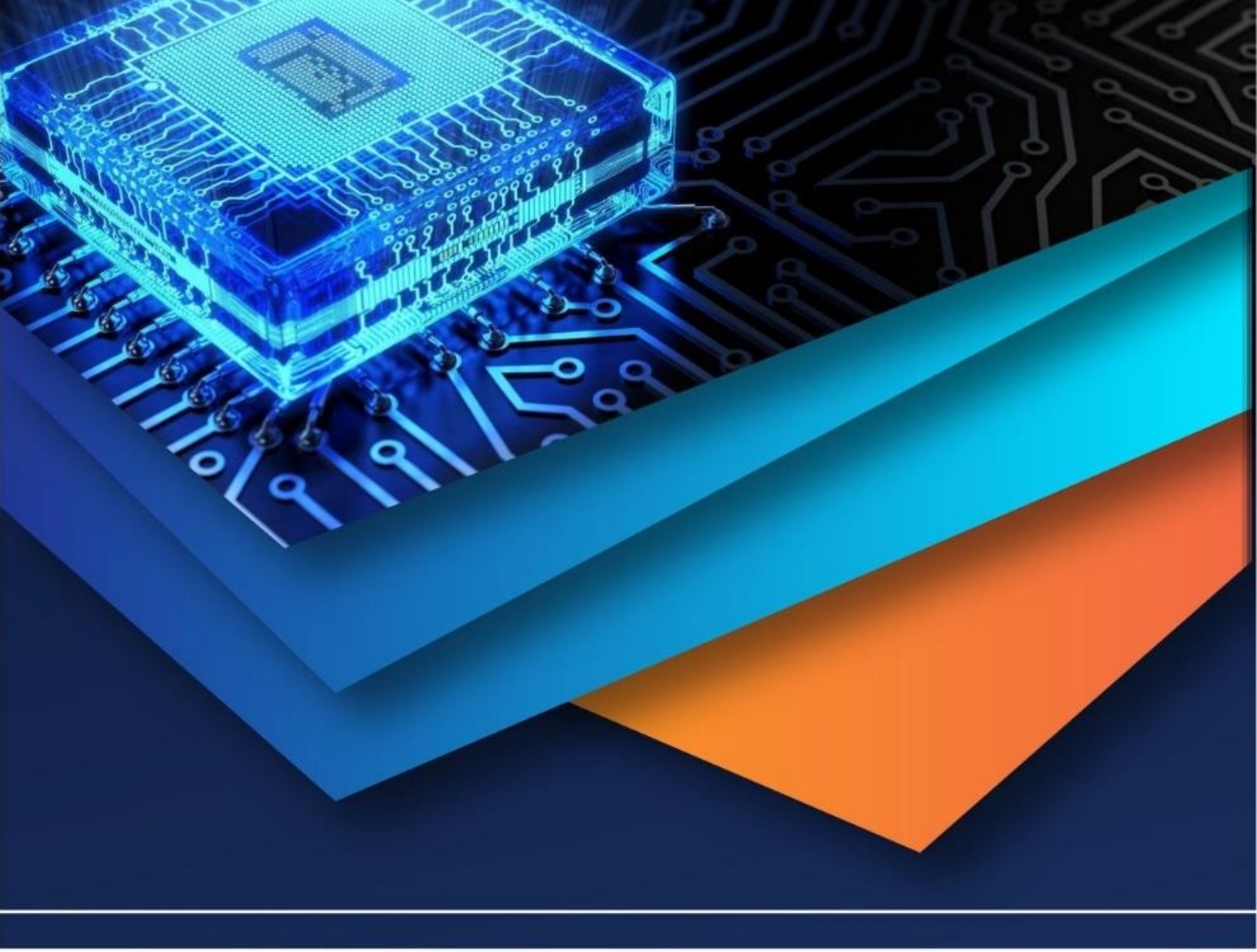
REFERENCES

[1] Grover N, Soni M. Design of FPGA based 32-bit floating point arithmetic unit and verification of its VHDL code using matlab. Modern Education and Computer Science Press; 2014. p. 1–14. [23] Iqbal F. Wavelet transform based image compression on FPGA. Master of Science Thesis, Florida State University; 2004. diginole.lib.fsu.edu

[2] Dhobale R, Chaturvedi S. Implementation of 32 bit binary floating point adder using IEEE-754 single precision format. IOSR J VLSI Signal Process (IOSR-JVSP) 2015;5(1):50–3.

[3] Dahiya S, Kumar R. Performance Analysis of different bit carry look ahead adder using VHDL environment. Int J Eng Sci Innov Technol (IJESIT) 2013;2:80–8.

[4] Hassan H, Ismail S. CLA based Floating-point adder suitable for chaotic generators on FPGA. In: 30th international conference on microelectronics (ICM); 2018.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY