iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 7 **Issue:** V **Month of publication:** May 2019

DOI: https://doi.org/10.22214/ijraset.2019.5400

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: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

Performance Analysis of Floating Point Multiplier Designs

Hemalatha K N[1], Shashikala M R[2], Seema Bhanu K I[3], Shwetha S M[4], Sundari G[5]
[1, 2, 3]*Assistant Professor, ECE Dept),* [4, 5]*Dr.Ambedkar Institute of Technology, Mallathalli, Bangalore-56*
(An Autonomous Institute aided by the Government of Karnataka, Affiliated to VTU- Belagavi)

Abstract: ***Floating point multiplier is the most typical illustration these days for real numbers on computers or laptops. This paper describes about a single precision floating point multiplier for better area, delay and the timing performance. The main object of the paper is to reduce the area, the delay and to increase the speed of execution by using shift and add and array multiplier for multiplying two floating point numbers. The implementations tradeoffs are area, speed, delay and accuracy. Floating point multiplier handles overflow and underflow cases. For high accuracy of the results normalization is additionally applied. By the use of pipelining process this multiplier is very good at speed and accuracy compared with previous multiplier. This pipelined architecture is described in Verilog code. The Timing performance is measured with Xilinx Timing Analyzer and Timing performance is compared with the normal multipliers.***
Keywords: floating point multiplier, single precision, multipliers, adders, Verilog code.

I. INTRODUCTION

Because of the quality of the algorithms, floating point operations are very hard to implement on FPGA. The computations for floating point operations involve massive dynamic range, however the resources required for this operations is high compared with the integer operations. We have unsigned/unsigned multiplier for multiplication of binary numbers, for multiplication of floating point numbers floating point multiplier is used. There is a separate algorithm for this multiplication. Floating point numbers are represented in binary format; the IEEE 754 standard represents two floating point formats, Binary interchange and Decimal interchange format. Multiplying floating point numbers is a requirement for DSP applications involving large dynamic range.[9]

A. Floating Point Multiplier

This multiplier factor is mainly used to multiply two floating point numbers. Separate algorithm is essential for multiplication of those numbers. Here multiplication operation is easy than addition this is especially true if we are employing 32-bit format.

B. Floating Point Format

One of the way to represent real numbers in binary is floating point format. There are different formats for the IEEE 754 standard. Binary interchange format and decimal interchange format. In the multiplication of floating point numbers involves a massive dynamic range which is useful in DSP applications. This paper concentrates solely on single precision normalized binary interchange format. Thefigure 1.1shows the single precision binary format representation; consisting of 1 bit sign(S),an 8 bit exponent€, and a 23 bit fraction (M or Mantissa).[9]

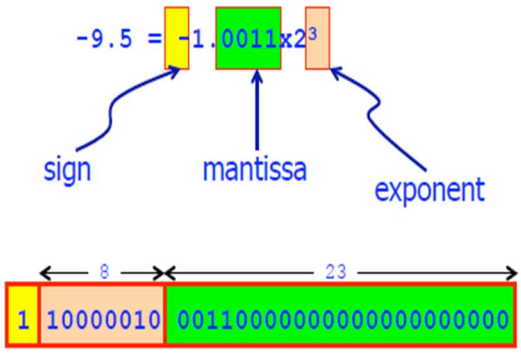


Fig1.1single precision format

The term floating point actual refers the radix point (decimal point ,or more commonly in computers, binary point) can "float", that is ,it can be placed anywhere relative to the significant digits of the number. Thisposition is indicated individual within the representation and floating point number representation will therefore be thought of as a computer realization of scientific notation. Over the years, a spread of floating point numbers representation have been employed in computers. However,since the Nineteen Nineties, the most unremarkably encountered representation is that outlined by the IEEE 754 standard.[9]

The advantage of floating point representation over fixed point and number representation is that it can support a way wider vary of values. Forexample, a hard and fast fixed point representation that has seven decimal digits with 2 decimal places will represent the numbers 12345.67,123.45,1.23 and so on, whereas a floating point representation (such as the IEEE754 decimal32 format) with seven decimal digits could in addition represent 1.234567,123456.7,0.00001234567,12345670000000000,and so on. The floating point format desires slightly additional storage (to encrypt the position of the radix point), therefore stored within in the same area, floating point numbers achieve their bigger range at the expense of precision.[9]

II. FLOATING POINT MULTIPLICATION ALGORTHIM.

Floating point multiplication are used in many digital circuits and signal processing computations, floating point multiplication design involves overflow and underflow.in this normalized floating point number have the form of Z=(-1S)*2(E-Bias)*(1.M).[9]
To multiply two floating point numbers it involves following steps:

A. Multiplying the significant; i.e. (1.M1*1.M2).
B. Placing the decimal point in the result.
C. Adding the exponents; i.e. (E1+E2-Bias).
D. Obtaining the sign; i.e. S1xor S2.
E. Normalizing the result; i.e. obtaining one at the MSB of the results "significant".
F. Rounding the result to suit within the available bits.

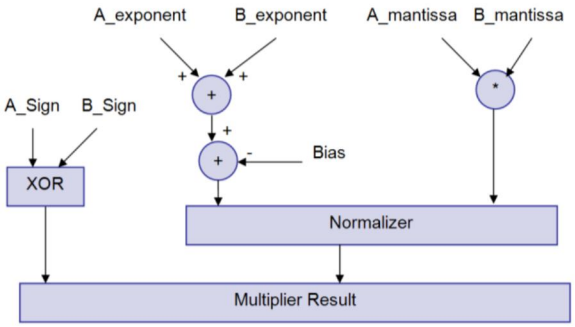


Fig 2.1 process diagram

III. ADDERS

A. Ripplle Carry Adder

A ripple carry adder is a logic circuit within which the carry-out of every full adder is that thecarry in of the succeeding next most vital full adder.it is called a ripple carry adder because each carry bit gets rippled into the next stage. In a ripple carry adder the sum and carry out bits of any half adder stage is not valid until the carry in of that stage happens. Propagation delays within the logic circuitry is that the reason behind this. Propagation delay is time advance between the applying of an input and occurrence of the corresponding output. Consider a NOT gate, once the input is "0"and the output will be "1" and vice versa. The time taken for the NOT gate's output to become "0" when the applying of logic "1" to the NOT gate's input is that the propagation delay here.Similarly the carry propagation delay is that the time moves on between the applying of the carry in signal and also the occurrence of the carry out (cout) signal. Circuit diagram of 4-bit ripple carry adder is shown below.[10]

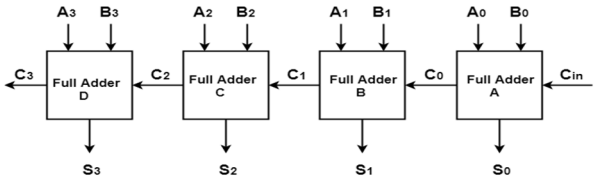


Fig 3.1 ripple carry adder

Sum out S0 AND carry out Cout of the full adder 1 is valid only after the propagation delay of full adder 1.in the same way, sum out S3 of the full adder four is valid solely when the joint propagation delays of full adder one to full adder four[6]. In simple words, the final results of the ripple carry adder is valid only after the joint propagation delays of all full adder circuits inside it.

B. Carry Lookahead Adder

A carry look ahead adder is a fast parallel adder because it reduces the propagation delay by additional advanced hardware, hence it's costlier. In this design, the carry logic over fixed groups of bits of the adder is reduced to two-level logic, which is nothing but a transformation of the ripple carry design [8]. This method makes use of logic gates so as to the lower the order bits of the augend and addend to see whether a higher order carry is to be generate or not.

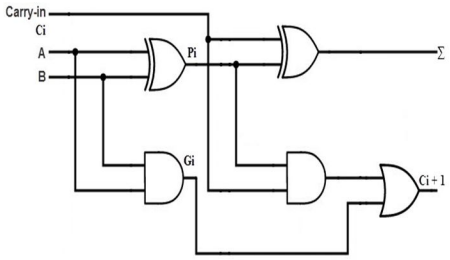


Fig 3.2 carry look ahead adder

For an n-bit carry look ahead adder to evaluate all the carry bits, we require:
Number of AND gate = n(n+1)/2.
Number of OR gate = n

C. Carry Save Adder

A carry-save adder is a digital adder, utilize in computer microarchitecture to compute the sum of three or more n-bit numbers in binary. It differs from different digital adders in this it outputs two numbers of the constant dimensions as the inputs, one which is a sequence of partial sum bits and another which is a sequence of carry bits[6].

Carry save adder is mainly used to calculate partial products that are generated by integer multiplier .by using carry save adder it is possible to reduce delay[11].

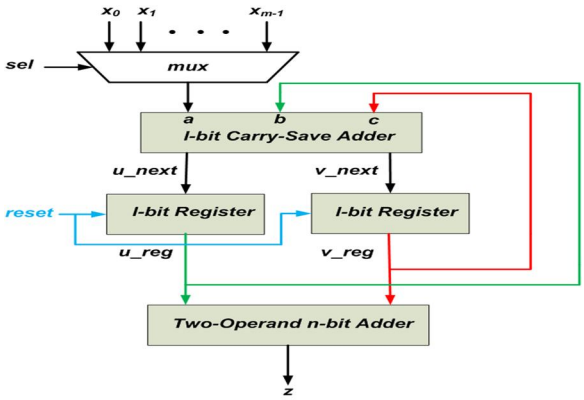


Fig 3.3 carry save adder

D. Kogge Stone Adder

The kogge stone adder may be a parallel prefix kind carry look ahead adder.The kogge stone adder takes a lot of space to implement the Kogge stone adder, however as a lower fan-out at everystage, that increase performance typical CMOS method nodes. However, wiring congestion often aproblem for a kogge stone adder. The lynch-Swartz lander design is smaller, as lower fan out, and does not suffer from wiring congestion; however to be used the process node must support Manchester carry chain implementations. The general problem of optimizing parallel prefix adders is identical to the variable block size, multilevel, carry-skip adder optimization problem, a solution of which is found in Thomas Lynch's thesis of 1996.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

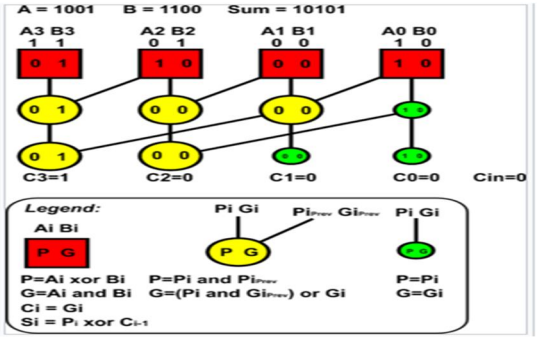


Fig 3.4 kogge stone adder

An example of a four bit kogge stone adder is shown in diagram. Each vertical stage produces a “propagate” and a “generate” bit, as shown. The culminating generate bits are produced in the last stage, and these bits are XOR’d with initial propagate after the input to produce the sum bits.e.g,the first sum bit is calculated by Xoring the propagate in the farthest-right red box (a “1”) with the carry-in (a “0”),producing a “1”.the second bit is calculated by Xoring the propagate in second box from the right (a”0”) with C0(a “0”),producing a “0”[1].

IV. MULTIPLIERS

A. Array Multiplier

Array multiplier is an layout of a combinational multiplier. The two’s complement multiplication is regenerate to an identical parallel array addition problem in which each partial product bit is 1` that the AND of a multiplier bit and a multiplicand bit, and the signs of all the partial product bits are positive. in array multiplier, consider two binary A and B,of m and n bits. There are mn summands that are produced in parallel by a sets of mn AND gates*n multiplier requires n(n-2) full adders half adders and n2 AND gates. Also, in array multiplier worst case delay would be (2n+1) td array multiplier factor provide optimum number of components required, however delay for this multiplier factor is larger. It also require large number of gates because of which area is also increased; due to this array multiplier is less economical thus, it is a fast multiplier but hardware complexity is high[7].

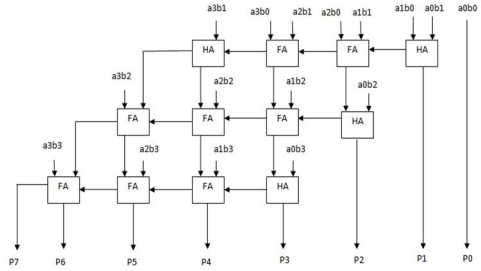


Fig 4.1 array multiplier

B. Shift And Add Multiplier

Shift and add multiplication is analogous to the multiplication performed by paper and pencil. This technique adds the multiplicand X to itself Y times, where Y denotes the multiplier .to multiply two numbers by paper and pencil, the algorithm is to take the number of the multiplier one at a time from right to left, multiplying the numberby one digit of the number and inserting the intermediate product within the acceptable positions to the left of the earlier results. Theoriginal algorithm shifts the multiplicand left with zeros inserted in the new positions, therefore the least significant bits of the product cannot change after they are formed[2]. Instead of shifting the multiplicand left, we are able to shift the product to the right. Therefore the multiplicand is fastened relative to the product, and since we tend to area unit adding only n bits, the adder needs to be only n bits wide. Only the left half of the 2n-bit product register is changed throughout the addition. Another observation is that the product register has associate empty space with the dimensions capable that of the number. As the empty space within the product register disappears, so do the bits of the number.In consequence, the ultimate version of the multiplier factor combines the product (A register) with the multiplier (Q register). The A register is just n bits wide, and therefore the product is formed within the A and Q register. Below figure shows the new version of the circuit[3].

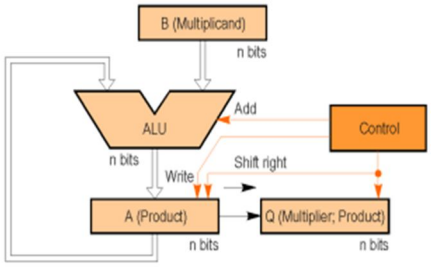


Fig 4.2 shift and add multiplier

C. Vedic Multiplier

Multipliers are based on Vedic Mathematics. Vedic Mathematics is an ancient mathematics concept developed by Sri Bharati Krishna Tirthaji between 1911 & 1918. It includes sixteen sutras, dealing with arithmetic, algebra, geometry, trigonometry and calculus. Vedic Multiplier discussed in this context is based on UrdhvaTiryakbhyam Sutra (literally- "Vertically and advantage that as the number of bits increases, the gate delay and area increases slowly. Therefore it is time, space and area efficient. The 2 u 2 Vedic Multiplier is implemented using four AND gates and two half adders. By using this sutra, the larger number (N u N) is broken down into smaller numbers (N/2 u N/2) and these smaller numbers are again broken into still smaller numbers (N/4 u N/4) till we reach multiplicand of size 2 u 2, thus simplifying the whole multiplication process. Hence 4 u 4 Vedic Multiplier is implemented using four 2 u 2 Vedic Multipliers and three 4-bit adders. Also an 8 u 8 Vedic Multiplier is implemented using four 4 u 4 Vedic Multipliers and three 8-bit adders [4]. Thus an N u N Vedic Multiplier is implemented using four (N/2 u N/2) Vedic Multipliers and three N-bit adders. A 32 u 32 VM is shown in Fig.

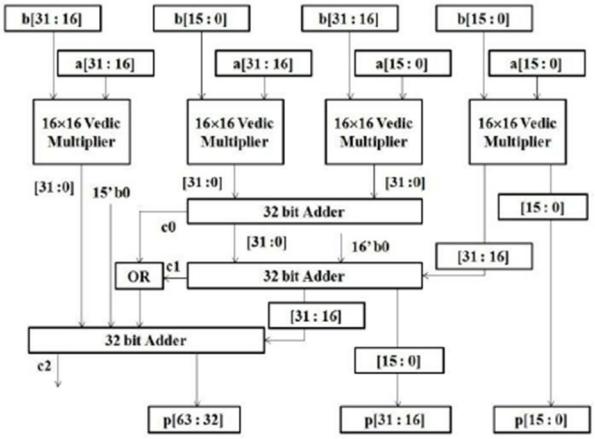


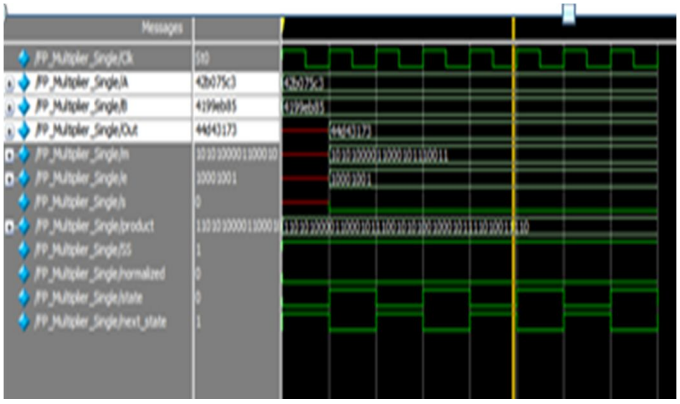
Fig 4.3 vedic multiplier

V. COMPARITION RESULT

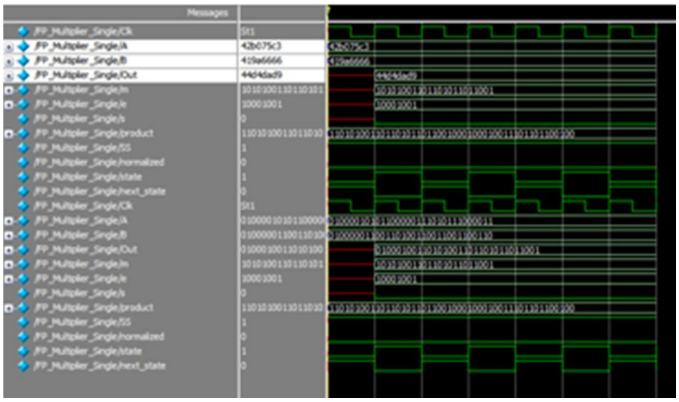
Method Name	Area in Number of LUT			Delay		
Spartan 3 XC 3S 4000-4FG1156	LUT	Slices	Gates	Delay	Gate or Logic Delay	Path or Route Delay
Array & Kogee Stone Adder	85	45	16795	18.803ns	14.845ns	3.958ns
S&A with Kogee Stone Adder	1088	551	9768	80.133 ns	52.416ns	27.717ns
Vedic Multiplier with Kogee Stone Adder	2175	1147	13674	92.053ns	36.388ns	55.665ns

VI. SIMULATION RESULT

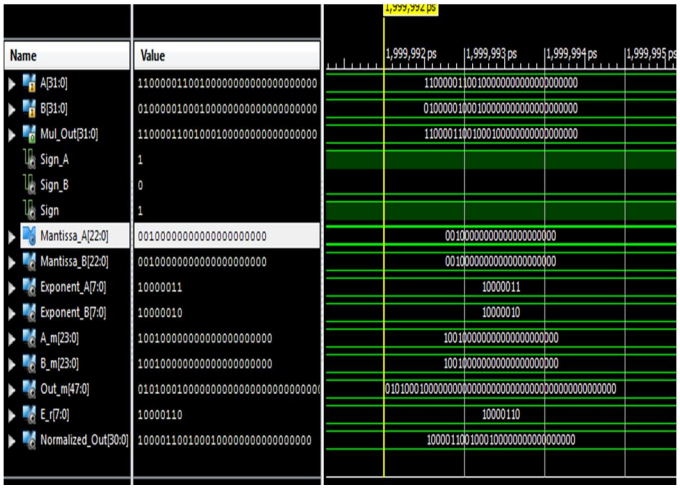
A. *Shift And Add Multiplier With Kogge Stone Adder*



B. *Vedic Multiplier With Kogge Stone Adder*



C. *Array Multiplier With Kogge Stone Adder*

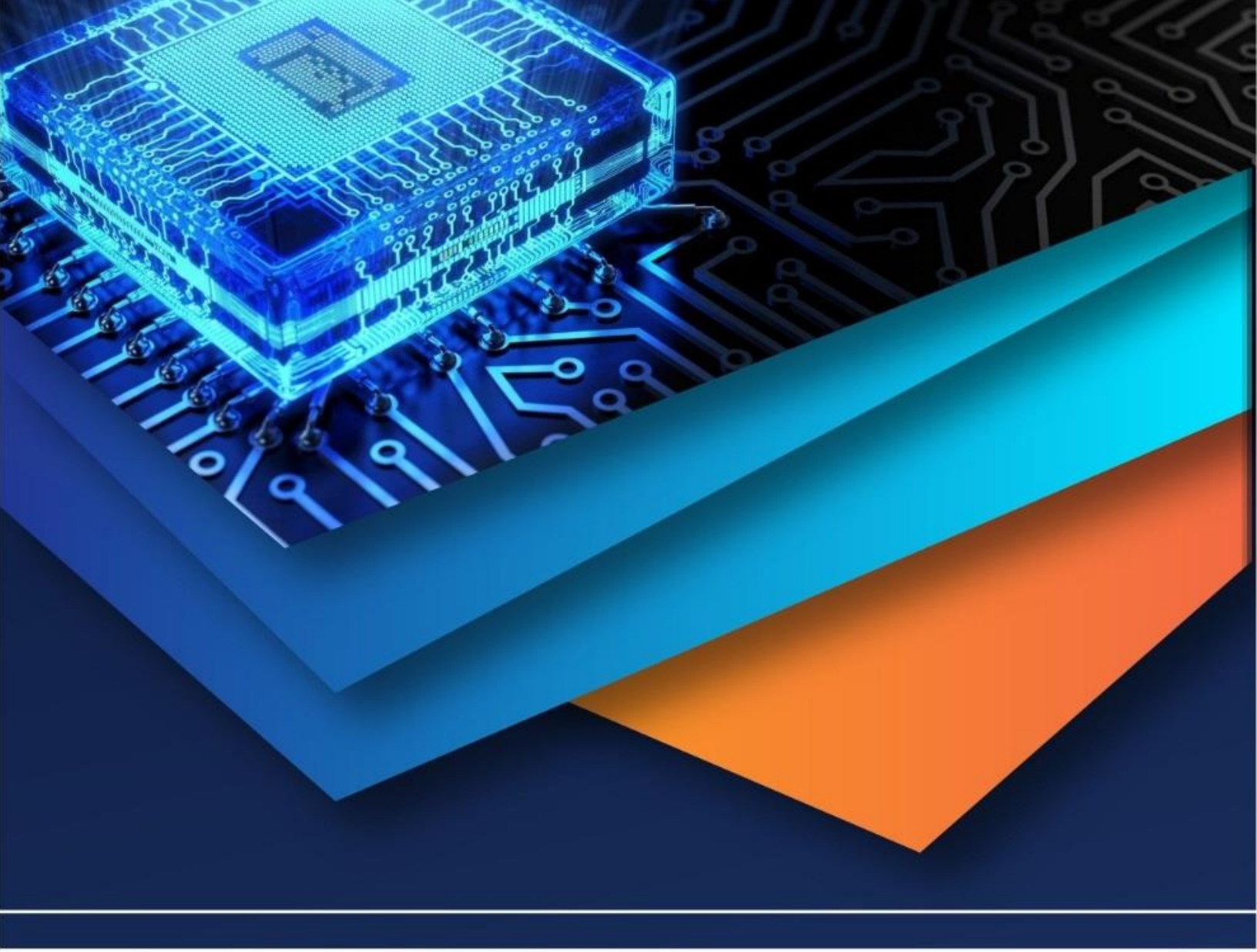


VII. CONCLUSION

This paper presents an implementation of IEEE 754 single precision floating point multiplier with pipelined architecture. The multiplier just presents the significant multiplication result as is 2 bits; this gives better precision if the whole 32 bits are utilized in another unit; the design has three pipelining stages. Timing performance is discovered with the tool Xilinx timing analyzer. Comparative results states that the planned pipelined design is of high speed style. Proposed pipelined approach is faster compared with conventional approach. This advantage is utilized in high speed DSP applications.

REFERENCE

[1] Geeta rani, “delay analysis of parallel- prefix adders”, international journal of science, engineering and technology research, July 14.
[2] IEEE 754-2008, IEEE Standard for floating- point arithmetic, 2008.
[3] Mohamed Al-Ashrfy, ashrafsalem and wagdyanis “An efficient implementation of floating point multiplier” IEEE Transaction on VLSI.
[4] Patil. S., Manjunatha , D. V., and Kiran, D. (2014, October). Design of speed and power efficient multiplier using vedic mathematics with VLSI implementation.
[5] D. Sangwan and M. K. Yadav, “Design and implementation of adder/subtracted and multiplication units for floating point arithmetic”, in international journal of electronics engineering,(2010),pp.197-203.
[6] BehnamAmelifard, FarzanFallah and MassoudPedram, “closing the gap between carry select adder and ripple carry adder: a new class of low power high-performance adders”. Sixth international symposium on quality of electronics Design, pp.148-152. April 2005.
[7] Pankaj Singh, Bhavinkakani “performance comparison of floating point multipliers by using different multiplication algorithm” international journal of electronics and communication, vol.3 issue, January 2015.
[8] Fu-Chiung Cheng Stephen H. Unger Michael Theobald Wen-Ching Cho “Delay-incentive Carry- Look ahead Adders”, VLSI Design, 1997. Proceedings. Tenth International Conference on4-7 Jan 1997.
[9] International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 1, Issue 7, September 2012.
[10] International Journal of Advanced Research in Computer Engineering (IJARCET) Volume 9,Issue 2,Mar-Apr 2014.
[11] International Journal of Advance Research in Computer Enginerring (IJARCET) Volume 3 Issue 9,September 2014.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY