IOSR Journal of Engineering (IOSR JEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
PP 63-68

Single Precision FPU Using Verilog: Review of Delay and Speed Parameters

Shraddha Hegde[1], Charmi Gandhi[2],Khushbu Patel[3], Prof. Kishor Bhosale[4]

[1](Department of Electronics, Atharva College of Engineering,Malad-West, Mumbai-400095,
[2](Department of Electronics, Atharva College of Engineering, Malad-West, Mumbai-400095,
[3](Department of Electronics, Atharva College of Engineering,Malad-West,Mumbai-400095,
[4](Department of Electronics, Atharva College of Engineering, Malad-West, Mumbai-400095,

Abstract: *A Floating point unit colloquially a math processor, is specifically designed to carry out operations on floating point numbers .For single precision itself more than 32 operations are possible. This paper deals with high speed ASIC implementation of single precision(32-bit) FP-ALU which performs add, subtract AND and OR operations using Verilog HDL language according to IEEE -754 standard . The delay parameter of simple Add operation is checked with RCA and CLA using Xilinx 14.7 ISE. The block diagram schematics are also viewed in Ni Multisim 14.0 software.*

Keywords: *CLA, RCA, FPU, Xilinx, Verilog, IEEE-754,Ni Multisim.*

I. Introduction

Due to its dynamic representation capability, Floating Point numbers are widely adopted in many applications.in comparison to integers and fixed point numbers, Floating point numbers provides a large range of values for computation but require either costly hardware or lengthy software implementations. So powerful techniques are required which reduces the hardware and improves performance like power, area ,delay timings etc.

In this paper, ASIC implementation of high speed single precision FPU has been carried out using efficient arithmetic operations like addition and subtraction along with logical operations like AND and OR using Verilog. Again the same FPU is designed using Ripple Carry Adder(RCA) as well as Carry Look Ahead Adder (CLA) to review their delays in Xilinx respectively.

I. IEEE-754 STANDARD FORMAT

IEEE-754 standard is used for the standardization of the floating point units. Here, the floating point operations are carried out efficiently with modast storage requirements. The standard addressed many problems found in the diverse floating point implementations that made them difficult to use reliably and reduced their portability. IEEE Standard 754 floating point is the most common representation today for real numbers on computers, including Intel-based PC's, Macs, and most Unix platforms.

IEEE 754 is the most efficient in most of the cases out of the several ways that is introduced. IEEE 754 has mainly 3 basic components: Sign, Exponent and Mantissa.

Single Precision
IEEE 754 Floating-Point Standard

IEEE 754 numbers are divided into two based on the above three components: single precision (32-bit)and double precision(64-bit). .So in our Single precision 32 bit, the last 23 least significant bits are the mantissa. The most significant bit is sign bit and the remaining 8 bits are the exponential part. Let's take an example to analyze it more clearly: 85.125
Each number is converted to binary form.
Then divided into it's 3 components.
85 = 1010101
0.125 = 001
85.125 = 1010101.001
=1.010101001 x 2^6

sign = 0
Single precision:
biased exponent 127+6=133
133 = 10000101
Normalised mantissa = 010101001
we will add 0's to complete the 23 bits.
The IEEE 754 Single precision is:
= 0 10000101 01010100100000000000000
This can be written in hexadecimal form as 42AA4000.
IEEE has also reserved some values that can ambiguity like denormalised , zero, infinity and not a number(NAN). The IEEE standard has been revised in 2008 and 2015 with some changes added to original one.

II. Literature Review

Memorydevices are getting economical and compact day by day due to it's increasing advancement in science and technology. Many applications require the representation and manipulation of large range of numbers which is why FPUs are used. There are many parameters that comes into picture when the design of this unit takes place. Different algorithims are also added to it.

- **Reduction in number of gates:**

Recently some authors have used the TSMC technology with proper constraints for add, substract, multiply and divide operations using multisim platform for simulation. Here, number of gates has been reduced from 2539 to 581. (Nisha Singh,R Dhanabal,2018).

- **Reduction in cell area:**

In the application of DSP processors mainly the floating point units are used. Here, to provide better latency results the counterpart of add-sub unit are fused together butterfly unit computation in fast Fourier transform algorithims(FFT). This fused unit helps in the reduction of cell area thereby increasing the speed and power consumption. (Aditi Sharma, Sukanya Singh, Abhay Sharma, 2017)

- **Supporting the denormal inputs:**

We already know that the FP arithmetic cores that are available in FPGA does not support denormal inputs. The xilinx implementation also treats denormal inputs as zero. The authors have tried to operate the frequencies greater than 300MHz for the adder unit implementation of IEEE754 that supports the denormal inputs along with the improvement in the performance and resource utilization. (Milind Shirke, Sajish Chandrababu, Yogindra Abhyankar,2017).

- **Reduction in power dissipation:**

The multi-threshold voltage technique is used in the reduction of power dissipation. The design of carry free adders along with clock gating techniques are used here. This method includes low V_t cells for low threshold gates while high V_t for higher ones. It also handles the exception and rounding-off errors like overflow, underflow and notes it down. (Shilpa Kukati Sujana D.V, 2013). For the efficient multiplier system, some authors used the parallelsim feature in multiplier present in booth recoded multiplier of mantissa. It has been designed for Spartan 6 of FGPA. This method helps in the improvement of certain parameters.(Sangeeta Palekar , Nitin Narkhede , 2016).

III. Block Diagram

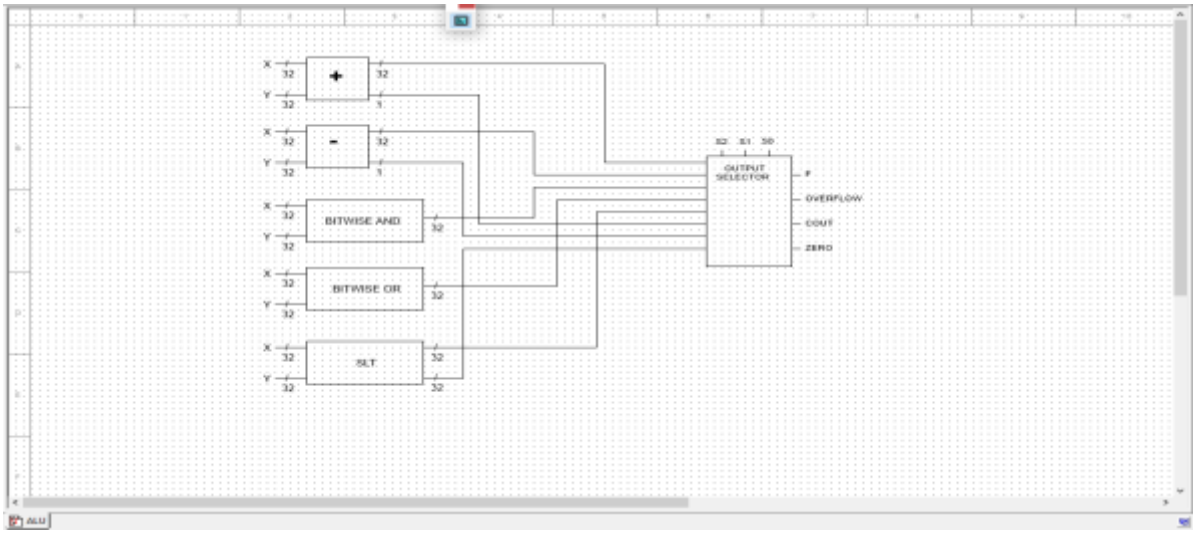

Fig.1. *Block Diagram of FPU unit.*

III. Proposed Method

8:1 mux has three select lines (S2,S1,S0) range is 0-7 (000-111) so each operation will select eg: 000-addition,001 subtraction.so according to opcode operation is select, opcode is based on select lines.

Table1:*output selector module*

Select line 1	Select line 2	Select line 3	operation
0	0	0	Addition
0	0	1	Subtraction
0	1	0	Multiplication
0	1	1	AND
1	0	0	OR

Floating point ALU (Addition, Subtraction , Multiplication):-
The implementation of a 32-bit Floating Point Adder with Verilog code is done by following steps

- Extracting signs, exponents and mantissas of both A and B numbers. As it has been said, the numbers format is as follows:
- Treating the special cases: 1)Operations with A or B equal to zero 2) Operations with ±∞.
- Shifting the lower exponent number mantissa to the right [Exp1− Exp2] bits. Setting the output exponent as the highest exponent. A's Exponent → 3, B's Exponent → -1, Difference (A-B) → 4 Number B: 1 1 0 1 0 0 1 → 0 0 0 0 1 1 0 1 0 0 1
- Working with the operation symbol and both signs to calculate the output sign and determine the operation to do. Table 1. Sign Operation A's Sign Symbol B's ,

Sign operation

- **Addition/Subtraction** of the numbers and detection of mantissa overflow (carry bit)
- Standardizing mantissa shifting it to the left up the first one will be at the first position and updating the value of the exponent according with the carry bit and the shifting over the mantissa. 0.101010123 → 1.01010122
- its (1 implicit bit + 23 mantissa's bits + 4 guard bits) is had in the Floating point Adder design then up to 28 positions must be able to shift. Because of the fact that this shifter consists of 5 stages: the first stage shift one position, the second stage 2, the third one 4, the fourth one 8 and the last one 16. Using any combination 32 positions are able to shift which is big enough to the design purpose.
- Detecting exponent overflow or underflow.

Eg:

1239.456+19.36997= (1.23456×10^3)+(1.936997×10^1)
= (1.23456×10^3)+(0.01936997×10^3)
=(1.25882997) ×10^3

=1258.82997

Multiplication

In this designed the single-precision multiplier for floating-point numbers. Use 23 bit fractions and 9 bit exponents, with negative numbers represented in 2's complement. Given two floating-point numbers, the product is

$(F1\times2E1) \times (F2 \times 2E2) = (F1 \times F2) \times 2(E1+E2) = F \times 2E$

The fraction part of the product is the product of fractions, and the exponent part of the product is the sum of exponents. Assumed that F1 and F2 are properly normalized; if they are not normalized than first normalize the fraction part of product.

Floating point ALU (RCA,CLA)

1) Ripple carry adder

If we consider the delay in terms of X units of time. For an n-bit adder, there will hence be a delay of, "n.X" While the adders are working in parallel, the Carry must "ripple" their way from the LSB and work their way to the MSB. It takes X units for the carry out of the rightmost column to make it as input to the adder in the column to its immediate left.

Ripple carry adder is an adder in which the carry bit ripple through all the stages of the adder. The ripple carry adder contain individual single bit full adders which consist of 3 inputs (*Augend, Addend and carry in*) and 2 outputs (*Sum, carry out*). These full adders are connected together in 'n' full adders combine together to form 'n' bit adder.

'n' bit adder will have 'n' stages and each stage's output will depend upon previous stage's carry out.

2) CLA (carry look ahead adder):

Ripple carry adder has certain limitation such as for large digit carry propagation at each stage makes the system slow to overcome this CLA is uses.

Considering two 32 bit no A and B are adding, C0 carry in and C32 are carry out,Carry look ahead adder's (CLA) logic diagram is given below. It contains 3 blocks; "P and G generator", "Carrylook ahead" block and "adder block". Input "Augend", "Addend" is provided to the "P and G generator" block whose output is connected with CLA and the adder block.

CLA block produces carry bits C_0 *to* C_{31}, and provide it to the adder block and evaluate sum S_0 *to* S_{31}, based on these inputs. $\mathbf{C_{31}}$ is taken as $\mathbf{C_{out}}$.

Carry = AB + C_{in} (A XOR B)

P = (A XOR B)

G = AB

C_n= G + C_{in} P

S_n = C_n XOR(A XOR B) for (n=0 to 31)

A CLA adder is faster and unique since it calculates multiple carries in parallel. Even for large numbers, the complexity of this form of addition stays simple. Stages of super-groups are added when needed. Considering the increase in digits, the corresponding increase in the number of gates is quite feasible.

IV. Synthesis Report

The synthesis of our FPU was done using Xilinx 14.7 ISE. All the timing diagrams generated from Isim simulator. Also the family of FPGA chosen for this program was Spartan 3 and the device name is XC3S50. The table below analyzes the delay of RCA and CLA based units. We can see that CLA has less delay compared to all three of them.

Table1:*review of delay parameter*

Design	**Delay** (in ns)
32-bit RCA	51.052
32-bit CLA	46.839

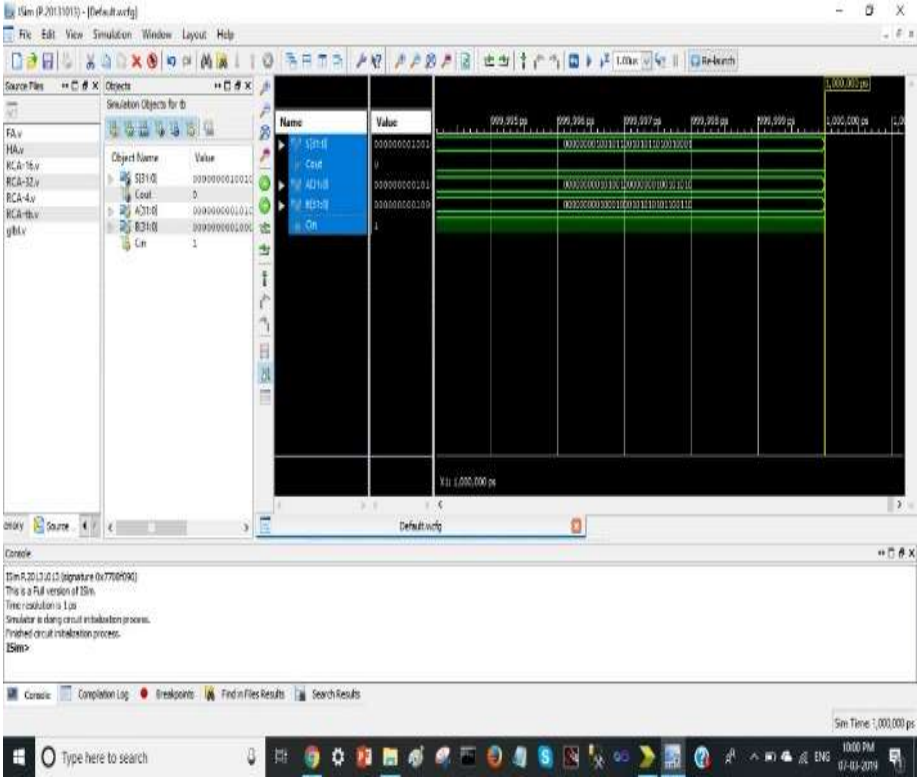

Fig 2:*RCA timing diagram from Isim*

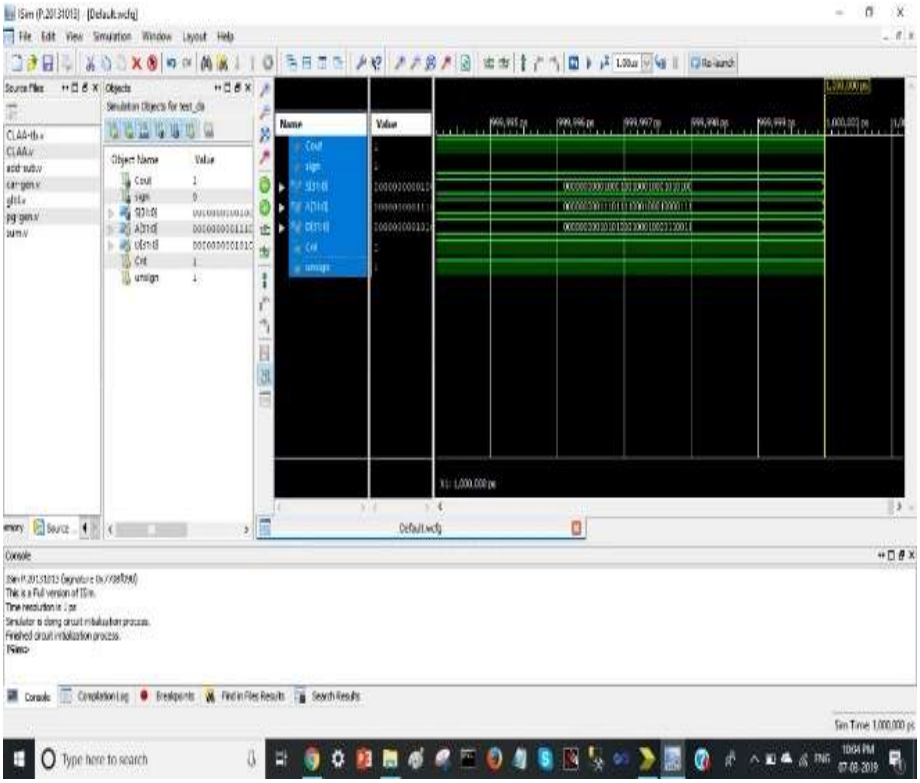

Fig 3:*CLA timing diagram from Isim*

V. Conclusion

The proposed design with proposed algorithims for arithmetic and logical operations accommodate an imperforate IEEE compatible system which is able to accomplish these operations at high speed. Their performances are compared on the basis of delay parameters.

References

[1]. Nisha Singh, R Dhanabal et al "Design Of Single Precision Floating Point Arithmetic Logic Unit"2018,IEEE.

[2]. Milind Shirke, Sajish Chandrababu, Yogindra Abhyanka et al "Implementation Of IEEE754 Compliant Single Precision Floating Point-Adder Unit Supporting Denormal Inputs On Xilinx FPGA."IEEE International Conference on Power, Control, Signals and Instrumentation Engineering(ICPCSI-2017).

[3]. Aditi Sharma, Sukanya Singh, Abhay Sharma, et al "Implementation Of Single Precision Conventional And Fused Floating Point Fused Add-Sub Unit Using Verilog"IEEE WiSPNET 2017 Conference.

[4]. Sangeeta Palekar, Nitin Narkhede et al "High Speed And Area Efficient Single Precision Floating Point Arithmetic Unit" ,IEEE International Conference on recent trends in Electronics Information Communication Technology ,May 20-21,2016.

[5]. Jenil Jain, Rahul Agrawal et al "Design And Development Of Efficient Reversible Floating Point Arithmetic Unit",IEEE 2015 Fifth International Conference On Communication Systems And Network Technologies.

[6]. Shilpa Kukati, Sujana D V, Shruti Udaykumar, Jayakrishnan P,Dhanabal R et al "Design And Implementation Of Low Power Floating Point Arithmetic Unit",IEEE2013.