ISSN 2278-3091
Volume 7, No.6, November - December 2018
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse06762018.pdf
https://doi.org/10.30534/ijatcse/2018/06762018

Square root operation of 64 bit floating point numerical data using Verilog coding

[1]S Rama Krishna, [2]R.Jayaram Kumar,[3] Lavanya.V, [4]M.Sai
[1 ,2 3, 4] Department of ECE, Godavari Institute of Engineering & Technology, Rajahmundry, A.P, India

ABSTRACT
Floating side of the point numerical operations constantly broadly utilized within expansive situated from claiming indicator transforming computation, scientific, business Also fund figuring. This usage includes methodology to registering four drifting point numerical operations with square root operation also. Done top-flight plan approach, four math modules: addition/subtraction, multiplication, division and square root would consolidate to structure a twofold precision skimming purpose unit. In this paper the suggested configuration consistent with IEEE-754 arrangement An 64 bit skimming purpose unit will be planned which will be also taking care of rounding, overflow, underflow & various exceptions to every operation. Verilog Also Questa-Sim configuration device around utilized for union and Recreation ceaselessly for exact comes about.

Keywords: twofold precision, coasting perspective unit, square root, equipment depiction dialect (HDL), IEEE-754.

1. INTRODUCTION
It is most proficient method for speaking to genuine no. In PCs. Gliding point expansion is most broadly utilized operation as a part of DSP/Math processors, robots, and air activity controller, advanced computers [1]. Digital number-crunching operations are vital in the outline of computerized processors and application in specific region. Math circuits frame an essential class of circuits in advanced frameworks. Progresses in the expansive scale incorporation (VLSI) circuit innovation, numerous intricate circuits have turned out to be effortlessly feasible today. The customary PC math systems, as well as the whimsical ones are worth examination in new outlines. In advanced sign preparing, applications including more exactness and range coasting point representation will be utilized. One of Applications will be sound preparing. A skimming - point math and rationale unit, for the most part term as Floating Point [2] Unit will be a part of a PC framework extraordinarily intended to complete operations on coasting point numbers. Skimming point portray a framework for speaking to numbers that would be too long or too short to be re-exhibited as whole number. Gliding point re-presentation held its determination and precision contrasted with the altered point number framework re-presentation.

The standard accommodates numerous firmly related configurations, contrasting in just a couple points of interest, similar to single exactness, double accuracy; two fold developed [3]. The square root is being examined here usage of effective square root administrator for figuring square foundation of twofold drifting point information, it bolster continuous sub-current and four adjusting modes. Status banner utilized for taking care of special cases are not set.

The IEEE 754 arrangements standard determines:
Basic and developed gliding point number configurations Add, subtract, duplicate, partition, square root, leftover portion, and think about operations Conversions in the middle of whole number and skimming point designs Conversions [4] between diverse gliding point groups Conversions between fundamental organizations coasting point numbers and decimal strings Floating-point special cases and their taking care of, including IEEE 754 standard presents of two distinctive drifting point groups as given by binary trade design and Decimal between change positions. The IEEE754 single exactness parallel organization representation; it comprises of an one piece sign (S), an eight piece type (E), and a twenty three piece part (M or Mantissa).If the example is more noteworthy than 0 and littler than 255, and there is 1 in the MSB of the huge then the number is said to be a standardized number [].

2. BACKGROUND
In the IEEE Standard 754 for Binary Floating-Point Arithmetic different work has done in this.
The IEEE has created a Standard to characterize Floating point representation and math. In spite of the fact that there are different representations, it is the most widely recognized representation utilized for drifting point numbers.
The standard brought out by IEEE come to be recognized as IEEE 754.
With regards to their exactness and width in bits, the standard characterizes 2 gatherings: fundamental and broadened group. The augmented arrangement is execution subordinate and doesn't concern this proposed work.
The essential organization is further isolated into single-accuracy configuration with 32-bits wide, and twofold exactness position with 64-bits wide. The 3 fundamental segments are the sign, type, and mantissa.
[6] [7] Use of Look-up Table, Vedic Approach: Dvan and Yoga Sutra, Paper and Pencil Method are more various square root algorithms have studied and implemented. By utilizing single accuracy and twofold exactness Floating

point Adder/subtraction, multiplier, divider and a great deal more modules had outline and execute. The square root calculations and usage have been tended to principally in three headings: Newton-Raphson technique SRT-Redundant strategy and Non-Redundant system.

3. IMPLEMENTATION

Those double precision coasting side of the point unit performs addition, subtraction, duplication What's more separate square root operations. Those square outline of twofold precision drifting purpose. It comprises of 5 sub squares. The obstructs need aid.

3.1 Addition and subtraction

In this part we are representing how to add and subtract two 32-bits floating point numbers. For example, there are two floating point numbers as follows such that both these floating point numbers can represent in the following form.[8] And there arithmetic operation (i.e. add/sub) can be easily understood by seeing the following process of addition and subtraction. [(-1) ^s1 * f1 * 2^e1] +/- [(-1) ^s2 * f2 * 2^e2]

Suppose e1>e2, then we can write it as, [(-1) ^s1 * f1 * 2^e1] +/- [(-1) ^s2 * f2‟‟ * 2^e2] (where, f2‟‟ = f2/ 2^ (e1-e2))

The result is-[(-1) ^s1 * (f1 +/- f2‟‟) * 2^e1] [6][4]

3.2 Multiplication

Suppose there are two floating point numbers as follows such that both these floating point numbers can be representing the following form. And there multiplication can be easily understood by seeing the following process of multiplication.

[(-1) ^s1 * f1 * 2^e1] * [-1) ^s2 * f2 * 2^e2] = (-1) ^ (s1 xor s2) * (f1 * f2) * 2^ (e1+e2)

Since 1 < (f1 * f2) < 4, result may need to be normalized. Conceptually, multiply operation is somewhat like simpler. If we look at two numbers-

(-1)^s1 * f1 * 2^e1 – 1st floating point number.

(-1) ^s2 * f2 * 2^e2 – 2nd floating point number.

The sign of the result is exclusive or of signs - if both are 1‟‟s or both are 0‟‟s – result will be zero. Fraction part is the product of two fractions and the exponent gets summed. There is no initial alignment which is required. But off course, we need to do normalization and rounding, because the product (f1 * f2), which are individually in the range 1 to 2. The resulted product would in range 1 to 4. That means the larger side, it may exceed two and it may require one small adjustment. If it exceed by 2, we may divide it by 2 and increment the exponent part [9].

3.3 Division

Suppose there are two floating point numbers as follows such that both these floating point numbers can represent in the following form. And there division can be easily understood by seeing the following process of addition and subtraction.

[(-1) ^s1 * f1 * 2^e1] / [(-1)^s2 * f2 * 2^e2] = (-1) ^ (s1 xor s2) * (f1 / f2) * 2^ (e1 – e2).

Since, 0.5 < (f1 / f2) < 2, result might essential to be normalized. (Assume f2 not equal to 0). Divide is similar. Two numbers are taken in same way.

If we look at same numbers-

As in figure 1 & 2 There division can be brought straight away. SIGN is written exactly in same way. Sign of result is exclusive or of signs. If both are 1's or 0's result will be zero. The fraction gets divided and exponent gets subtracted. Now, ratio of fractions would be in the range [10]

0.5 < (f1 / f2) < 2.

So it will not exceed 2, because each one is in the range 1 and 2. But it can become small; hence we may need to normalize.

4. RESULTS

4.1 Addition and subtraction module

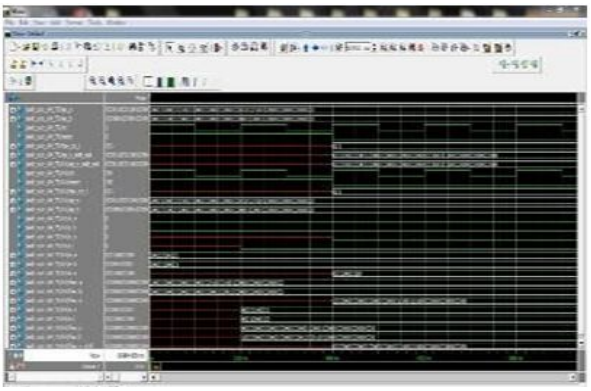

Figure 1: Simulation window- addition and subtraction module

4.2 Multiplication module

Inputs op_a and op_b are of 32-bit binary floating point.

op_a = 10110010010010000000000000000000

op_b = 00110100010000000000000000000000

After giving the select line value, fpu_op_i = 010. We are getting output, and we will the output of size 64-bits floating point number.

Op_o_mul=1010100110001001000000000000000000000000000000000000000000000000

Figure 2: Simulation window- multiplication module

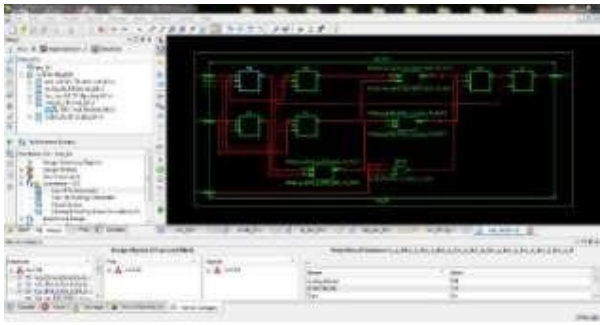

Figure 3: RTL Window- multiplication module

4.3 Division module

Inputs op_a and op_b are of 32-bit binary floating point. op_a = 00110111000011000000000000000000
op_b = 10110010000110000000000000000000
After giving the select line value, fpu_op_i = 011. We are getting output, and output will be of 32-bits floating point number. As shown in figure 3,4 & 5.
Op_o_div=11000100000000000000000000000001

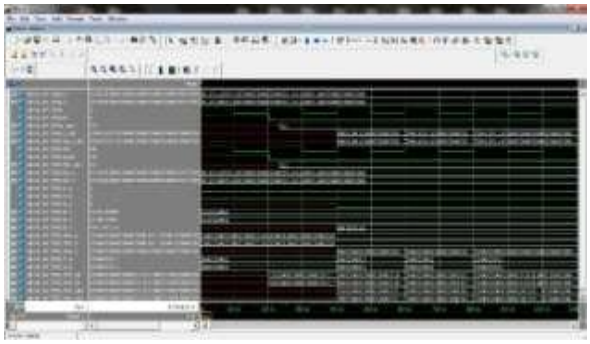

Figure.4. Simulation window-division window

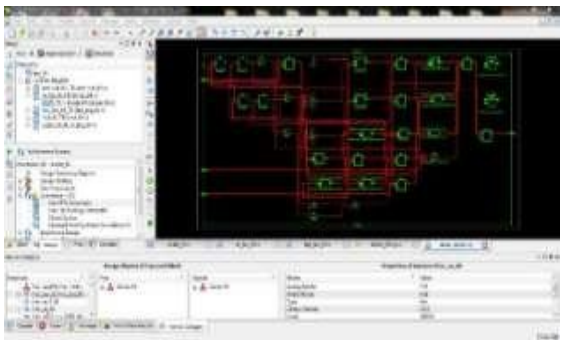

Figure.5. RTL Window-division module

4.4 Square root module

Input op_a are of 32-bit binary floating point. op_a = 00001000010111001010100000000
After giving the select line value, fpu_op_i = 100. We are getting output, and output will be of 32-bits floating point number as in figure 6. Op_o_sr = 001000111101101111011

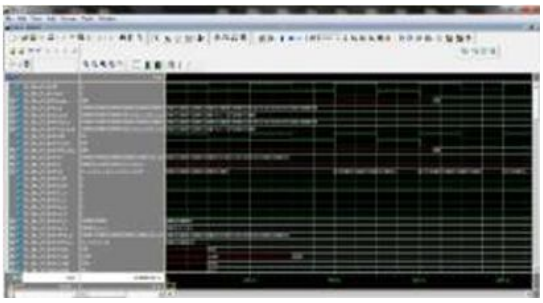

Figure 6: Simulation window- square root module

4.5 Top level module

Subsequent to giving inputs op_an and op_b, to every one of the four sub modules, i.e. Expansion/Subtraction module, Multiplication module, Division module and Square root module as in figure 7, we will therequired result by applying qualities to a specific selecting pin.

Figure 7: Simulation window- top level module

Figure 8: RTL window- top level module

Figure 8 representing the register transfer level window of a Top level module. In this particular register transfer window we are now able to see the detailed interconnections between the several transistor blocks of addition/subtraction module

5. CONCLUSION

Till now, we are getting just four functionalities: expansion, subtraction, duplication, and division, from already proposed outlines.In this particular design we had further introduced a new functionality of square root such that this design will be now capable of doing five operations: addition, subtraction, multiplication, division, and square root of double precision numbers. Each module is independent from each other. The modules are realized and validated using Verilog simulation in the Questasim and synthesis using Xilinx ISE Design Suite13.3.

More work can be done in this particular project regarding power consumption. We can simply use the concept of clock gating, to resolve the problem. Such that only the required sub module will perform its operation, and all other sub module will remainoff, on applying the values select pins.

REFERENCES

[1] Preethi Sudha Gollamudi, M. Kamaraju„"Design of High Performance IEEE- 754 Single Precision (32 bit) Floating Point Adder UsingVHDL""inInternationalJournalofEngineeringResearch&Technology(IJERT)Vol.2Issue7,July–2013.

[2] Addanki purna Ramesh, Ch.Pradeep "Fpga Based Implementation of Double Precision Floating Point Adder/Subtractor Using Verilog" International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 7, July2012).

[3] SARC International Conference, 30th CHAITANYA A. KSHIRSAGAR, P.M. PALSODKAR "An FPGA Implementation ofIEEE - 754 Double Precision Floating Point Unit Using Verilog" Proceedings of 4thSARC International Conference, 30th March-2014, Nagpur, India.

[4] Swathi. G. R, Veena. R"Design of IEEE-754 Double Precision Floating Point Unit Using Verilog"in International Journal of Engineering Research & Technology (IJERT)International Journal of Engineering Research & Technology (IJERT)Vol. 3 Issue 4, April – 2014.

[5] Kusuma Keerthi "Design of High Performance Single Precision Floating Point Multiplier in (ITSI-TEEE)ISSN (PRINT) 2320 – 8945, Volume -2, Issue-1,2014.

[6] Ravi Payal "Simulation and Synthesis Model for the Addition of Single Precision Floating Point Numbers Using VERILOG" InternationalJournalofEngineeringResearch&Technology(IJERT)ISSN:2278-0181Vol.2Issue9,September–2013.

[7] UshasreeG,RDhanabalSaratKumarSahoo"ImplementationofaHighSpeedSinglePrecisionFloatingPointUnitusingVerilog"

[8] NajibGhatte, ShilpaPatil, Deepak Bhoir "Double Precision Floating Point Square Root Computation" International Journal of Engineering Trends and Technology (IJETT) –Volume13Number6–Jul2014.

[9] Sravana J , M Lakshmi Raviteja, G Anil Reddy," IC Layout Design Of Decoder Using LVS", International Journal Of Advanced Trends In Computer Science And Engineering, Vol.5 , No.1, Pages : 60-64, (2016)

[10] Srihari C1,Anil Reddy G, Sruthi L,"An High Speed Two 'S Complement Multiplier Realization In Fpga", International Journal Of Advanced Trends In Computer Science And Engineering, Vol.5 , No.1, Pages : 65-69, (2016)