Floating Point ALU

Pradnya Shengale, Prof. Vidya Dahake, Prof. Mithilesh Mahendra

Electronics and communication, Abha Gaikwad Patil Collage of Engineering and Technology, Nagpur, Maharastra, India

Abstract:- The floating point unit (FPU) implemented during this project, is a 32-bit processing unit which allows arithmetic operations on floating point numbers. The FPU fully compiled using the IEEE 754 Standard [1]. The FPU supports the following arithmetic operations Add, Subtract, Multiply, Divide. For each operation the rounding modes are supported such as Rounded to nearest even number, Round to zero, Round up, Round down. The FPU is generally written in VHDL with top priority to be able to run at approximately 100-MHz and also should be as small as possible. Meeting both results at the same time was very difficult and tradeoffs were made.

Keywords: ALU, Pipelining,IEEE standard 754, floating point , VHDL.

I. INTRODUCTION

The concept of floating-point representation over integer fixed-point numbers, which consist purely of significantis that expanding it with the exponent component achieves greater range. For example in order, to represent large values, e.g. all 39 decimal are needed to be place down to femtometre-resolution in order to determine the distance between two galaxies. But if the best resolution is assumed in light years, only the 9 most significant decimal digits, where will be of importance as the remaining 30 digits purely carry noise, and thus can be safely avoided. The term floating point actuallyrefers to the fact that a radix point of any number, (decimal point, or, more commonly in computers, binary point) can "float"; meaning that it can be laced anywhere inrelation to the significant digits of the number. From the past few years a variety of floating-point representations have been utilized in computers, The most commonly referred representation is that which is defined by the IEEE 754 Standard, since 1990's.

II. SYSTEM MODEL

a) Floating point numbers

The floating-point representation is one way to represent real numbers. A floating-point number *n* is represented with an exponent *e* and a mantissa *m*, so that:

$n = b^e \times m$, …where b is the base number (also called radix)

So for example, if we choose the number n=17 and the base b=10, the floating-point representation of 17 would be:

$$17 = 10^1 \times 1.7$$

Another way to represent real numbers is to use fixed-point number representation. A fixed-point number with 4 digits after the decimal point could be used to represent numbers such as: 1.0002, 12.1029, 34.0001, etc. Both representations are used depending on the condition. For the implementation on hardware, the base-2 exponents are used, since digital systems work with binary numbers.

b) FPU ARCHITECTURE

Figure 2 gives the architecture of floating point unit. This is a simple single precision floating point unit.

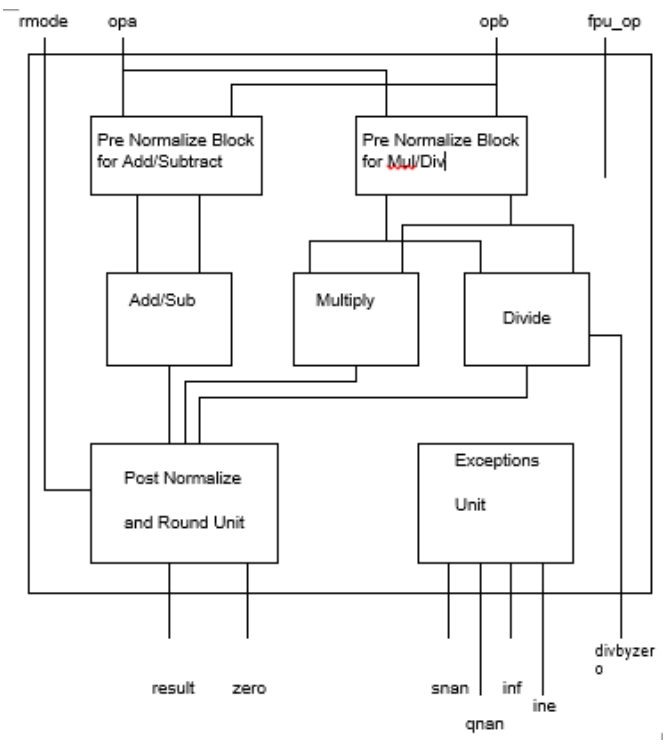

Fig 2 :FPU Architecture

Two pre normalization units adjust the fractions, first does it for add and subtract operation, the second for multiply and divide operation. The FPU supports the following mode In this section author should explain in little bit dept about his research or model he/she is working on. Author can be use

suitable diagrams and images with the references mentioned [1] in square brackets from particular resource image or diagram author taken.

III. PREVIOUS WORK

In recent years, Floating-point numbers are widely adopted in many applications due to its high dynamic range and good robustness against quantization errors, capabilities. Floating-point representation is able to retain its resolution and accuracy. IEEE specified standard for floating-point representation is known as IEEE 754 standard. This standard specifies interchange and arithmetic formats and methods for binary and decimal floating-point arithmetic in computer programming environments.[IEEE 754-2008]

The main objective of implementation of floating point operation on reconfigurable hardware is to utilize less chip area with less combinational delay [Karan Gumber et.al, May 2012] which means less latency means faster speed. A parameterizable floating point adder and multiplier implemented using the software language such as Handel-C. Using the Xilinx XCV1000 FPGA, a five stages pipelined multiplier achieved for 28MegaFlops [A. Jaenicke et. Al, 2001]. The hardware used for the parallel 32-bit multiplier is approximately 3 times that of serial.

A single precision floating point multiplier that doesn't support rounding modes can be implemented using a digit-serial multiplier [L. Louca et. al, 1996]. The ALU is a basic building block of the central processing unit of a computer or microprocessors, and even the simplest microprocessors contain one for purposes such as maintaining the timers. By using pipelined ALU design, ALU provides a good performance with pipelining concept. ALU execute many instructions simultaneously [Suchita Pare et. al, 20]

rmode	Rounding Mode
0	Round to nearest even
1	Round to Zero
2	Round to +INF (UP)
3	Round to -INF (DOWN)

In IEEE 754, the IEEE has standardized the computer representation for binary floating-point numbers. Almost all modern machines follows this standard. It is an effort for the computer manufacturers to conform to a common representation and arithmetic convention for floating point data. The standard defines:

Arithmetic formats: binary and decimal floating-point data sets, which contains finite numbers (including signed zeros and subnormal numbers), infinities, and special "not a number" values (NANs).

Interchange formats: encodings (bit strings) that may be utilized to exchange floating-point data in a better and compact form.

Rounding rules: Satisfaction of certain properties requires during arithmetic and conversions to perform rounding of numbers.

Operations: arithmetic and other operations on arithmetic formats Exception handling: indications of conditions like division by zero, overflow, etc.

A 32 bit word is required for the IEEE single precision floating point standard representation requires whose bits may be represented as numbered from 0 to 31, left to right. Figure 1 shows the format of single precision floating point.

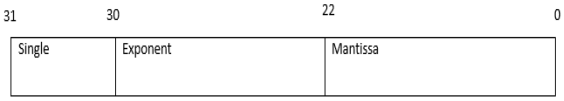

Fig 1. Single precision floating point

Sign bit indicate whether the number is positive or negative. If it is '1' then the number is negative and if it is '0' then the number is positive. "Exponent" is of 8 bit which provides the exponent range from E (min) =-126 to E (max) =127. The fractional part of a number is given by the Mantissa which is of 23 bit. The mantissa must not be confused with the significand. The leading "1" in the significand is made implicit.

IV. PROPOSED METHODOLOGY

The FPU can perform a floating point operation every cycle. It will latch the operation type, rounding mode and operands and deliver a result four cycles later.

The FPU will never generate a SNAN output. The SNAN output is asserted when one of the operands was a signaling NAN (output will be a quiet NAN). When performing a floating point to integer conversion, the output (representing an integer) can take on forms of a NAN or INF, which are purely legal integers.

1 ADDITION/ SUBTRACTION

While adding the two floating point numbers, two cases may rise.

Case 1: when both the numbers are of same sign i.e. if both the numbers are either +ve or -ve.

In this case MSB of both the numbers are either 1 or 0.

Case II: when both the numbers are of different sign i.e. if one number is +ve and other number is –ve. In this case the MSB of one number is 1 and other is 0.

Case 2:When two numbers have different signs

Take two numbers check the sign of the two numbers. If the Sign of any of the two numbers is different then take the 2's complement of the respective number and then add the two numbers. Following are the algorithm to make addition and subtraction.

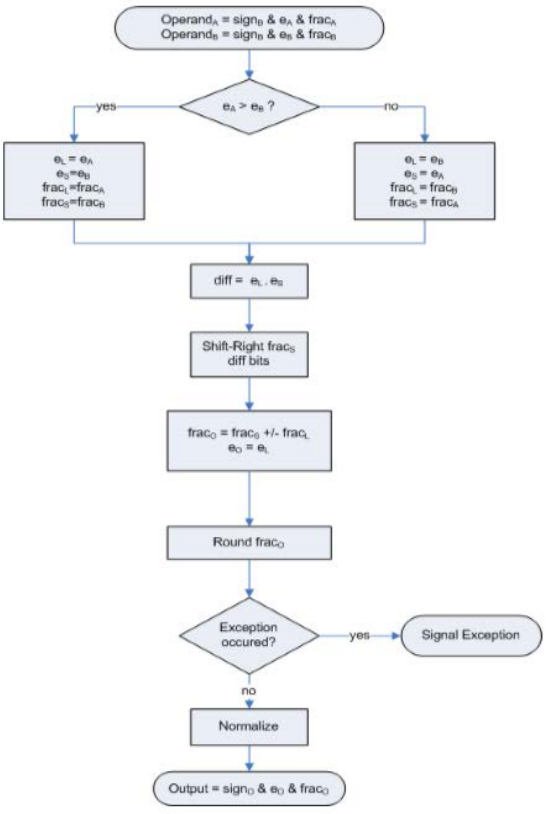

Fig.3: Flow chart for Addition /Division

2. MULTIPLICATION

Take the two normalized operands. Multiply thesignifcands. Then add the exponents and determine the sign .Normalize mantissa and update exponent. Find exception flags and determine also special values for over flow and underflow. The algorithm used for the multiplication is shown below.

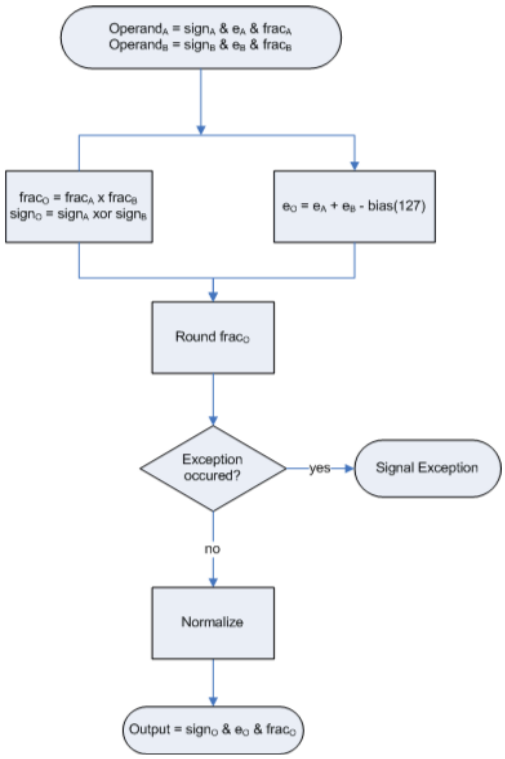

Fig. 4:- Flow chart of single precision multiplication

3. .DIVISION

Divide signifcands, subtract exponents, and determine sign.Normalise mantissa and update exponent. Find exception flags and determine also special values for over flow and underflow.

V. SIMULATION/EXPERIMENTAL RESULTS

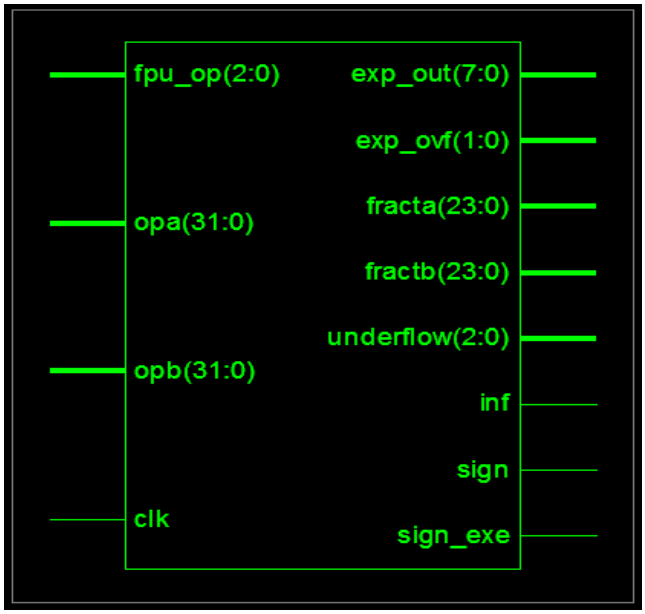

Fig a :-Top level entity of proposed single precision FPU

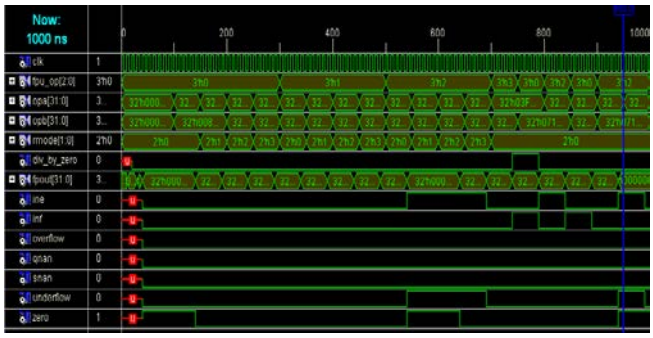

Fig b :-Waveform of FPU 32 Bit.

VI. CONCLUSION

The Floating point Arithmetic and unit has been discussed and suitable algorithm has been developed to perform operation such as addition, subtraction, multiplication and division.

VII. FUTURE SCOPES

The algorithm can be implemented in pipelined way to reduce the delay and increase the computation time for operation. The floating point numbers. IEEE 754 standard based floating point representation also can be used to operation like square root.

REFERENCES

[1] Simulink HDL Coder 1; User's Guide; 2006-2010 by the MathWorks, Inc.

[2] Hikmat N. Abdullah and Hussein A. Hadi "Designand Implementation of FPGA Based Software Defined Radio Using Simulink HDL Coder". Engineering and Technology Journal, Iraq, ISSN 1681-6900 01/2010; Vol.28 (No.23):pp.6750-6767.

[3] B. K. Mishra, S. Save, R. Mane. A frame work for model based designing of analog circuits using Simulink. ICWET '11 Proceedings of the International Conference & Workshop on Emerging Trends in Technology. Pages 1225-1228;

[4] Sign of FPGA based 32-bit Floating Point Arithmetic Unit and verification of its VHDL code using MATLAB ACM New York, NY, USA ©2011 ISBN: 978-1-4503-0449-8.

[5] Alejandro A. Valenzuela, Hikmat N. Abdullah. A Joint Matlab/FPGA Design of AM Receiver for Teaching Purposes. Electromagnetics and Network Theory and their Microwave Technology Applications 2011, pp189-199.