ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Implementation of Single Precision Floating Point Multiplier

Jannu Chaitanya
PG Scholar In VLSI Design,
Department of ECE,
Dhanekula Institute of Engineering & Technology,
Ganguru, Krishna Dist., Andhra Pradesh, India.

K. Rama Koteswara Rao
Associate Professor,
Department of ECE,
Dhanekula Institute of Engineering & Technology,
Ganguru, Krishna Dist., Andhra Pradesh, India.

ABSTRACT

To represent very large or small values, large range is required, as the integer representation is no longer appropriate. These large range values can be represented using the IEEE-754 standard based floating point representation. An arithmetic circuit which performs digital arithmetic operations has many applications in digital coprocessors, application specific circuits, etc. Because of the advancements in the VLSI technology, many complex algorithms that appeared impractical to put into practice, have become easily realizable today with desired performance parameters so that new designs can be incorporated. The standardized methods to represent floating point numbers have been instituted by the IEEE 754 standard through which the floating point operations can be carried out efficiently with modest storage requirements. This project work deals with the design of high speed floating point multiplier which performs multiplication on 32-bit operands that use the IEEE 754-2008 standard. The algorithm is modeled in Verilog HDL and the RTL code for the floating point multiplier is synthesized using cadence RTL compiler where the design is targeted for 180nm TSMC technology with proper constraints in terms of area and power. The Layout is generated by cadence SOC Encounter. And the same is implemented on Hardware. Here we use Xilinx FPGA virtex5 family. The chip-scope pro is used to observe the simulation results.

INTRODUCTION

Very large scale integration (VLSI) is the process of creating an integrated circuit (IC) by combining thousands of transistors into a single chip. VLSI began in the 1970s when complex semiconductor and communication technologies were being developed. The microprocessor is a VLSI device. Before the introduction of VLSI technology most ICs had a limited set of functions they could perform. An electronic circuit might consist of a CPU, ROM, RAM and other glue logic. VLSI lets IC designers add all of these into one chip. Power dissipation is recognized as a critical parameter in modern VLSI design field. To satisfy Moore's law and to produce consumer electronics goods with more backup and less weight, low power VLSI design is necessary. Fast multipliers are essential parts of digital signal processing systems. The speed of multiply operation is of great importance in digital signal processing as well as in the general purpose processors today, especially since the media processing took off. Multipliers are key components of many high performance systems such as FIR filters, microprocessors, digital signal processors, etc. A system's performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the system. Furthermore, it is generally the most area consuming. Hence, optimizing the speed and area of the multiplier is a major design issue. The IEEE 754 standard have instituted some standardized methods to represent floating point numbers. IEEE 754 standard supports both single and double precision ranges. The IEEE 754 standard presents two floating point formats, Binary interchange and decimal inter change format. Multiplying floating point numbers is a critical requirement for DSP applications involving large dynamic range. This paper focuses only on single precision normalized binary interchange format. Fig. 1 shows the IEEE 754 single precision binary format representation; There are three basic components in IEEE 754 standard floating point representation, those are one bit sign (S), an eight bit exponent (E), and a twenty three bit fraction(M or Mantissa). An extra bit is added to the fraction to form what is called the significand. The exponent with 8 bits represents both positive and negative exponents. A bias of 127 is added to the exponent to get the stored exponent [2]. Table 1 show the bit ranges for single (32-bit) and double (64-bit) precision floating-point values [2].

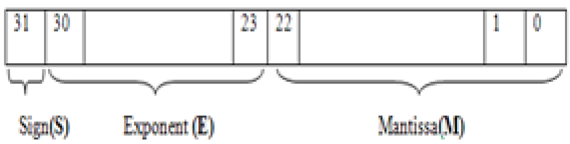


Fig.1 IEEE 754 single precision binary format representation

The value of floating point number is as follows

$$Z=(-1) * 2^{(E-Bias)} *(1.M)$$

$$\text{Where } M = m_{22}2^{-1} + m_{21}2^{-2} + m_{20}2^{-3} + \ldots\ldots + m_{1}2^{-22} + m_{0}2^{-23}$$

$$Bias = 127.$$

Table 1: Bit ranges for single (32-bit) and double (64-bit) precision floating-point values

Precision	Sign	Exponent	Mantissa	Bias
Single	1[31]	8[30-23]	23[22-0]	127
double	1[63]	11[62-52]	52[51-00]	1023

There are some exceptions that arise during floating point multiplication. The Overflow exception is raised whenever the result cannot be represented as a finite value in the precision format of the destination [13]. The Underflow exception occurs when an intermediate result is too small to be calculated accurately, or if the operation's result rounded to the destination precision is too small to be normalized.

ALGORITHM AND ARCHITECTURE

The following shows the algorithm steps for single precision floating point multiplication.

1. Multiplying the significand: Non-signed multiplication of mantissas, it must take account of the integer part, implicit in normalization. The number of bits of the result is twice the size of the operands (48 bits).
2. Adding the exponents: Addition of the exponents, taking into account the bias.
3. Obtaining the sign: calculation of the sign.
4. Normalizing the result: The exponent can be modified accordingly.
5. Checking for Overflow/Underflow occurrence: The underflow/overflow may when the resultant exponent is too small/too large to represent in the exponent field.

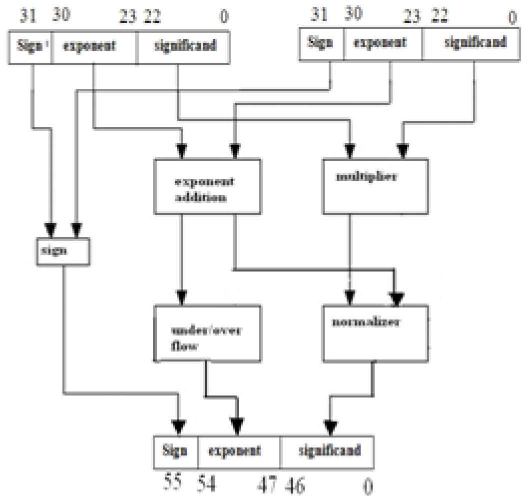


Fig 2 Block Diagram of floating point multiplier

Here we use the ripple carry adder for addition of exponents and the ripple subtractor is used to subtractor the bias from the result of ripple adder which is our exponent output of floating point multiplier. The significant multiplication is done on 24 bit. Multiplication consists partial product generation and partial product addition. Different Partial product reduction mechanisms are used in order to reduce the overall area and power. Modified booth encoding (Bit pair recoding) technique is the advanced partial

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

product reduction. Bit pair recording is shown in below table2. In high speed designs the Wallace tree construction is usually used to add the partial products in a tree-like fashion in order to produce two rows of partial products that can be summed up in the last stage. By using this number of addition stages are increased, and there by area, power are increased proportionally. Hence to decrease the area and to achieve low power and to enhance the speed we use compressors to add the partial products. The proposed method considers all the bits in each column at a time and compresses them into two bits. Here we use 4:2 compressors

Table2: Bit pair recording

Block	Re-coded digit	operation
000	0	0X
001	+1	+1X
010	+1	+1X
011	+2	+2X
100	-2	-2X
101	-1	-1X
110	-1	-1X
111	0	0X

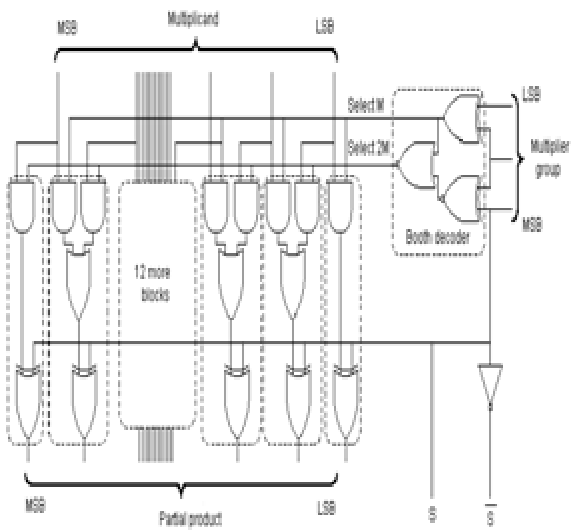


Fig.3 Partial product generator

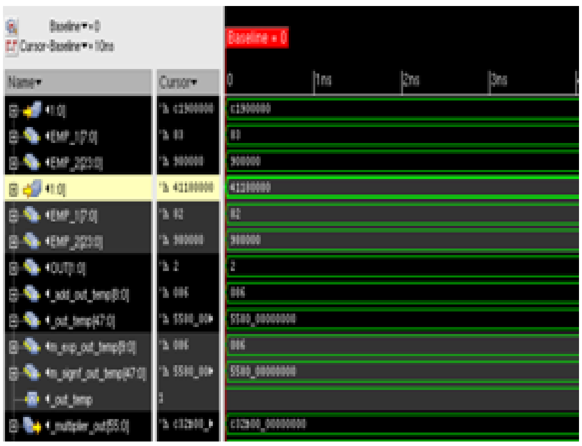
HARDWARE IMPLEMENTATION

The single precision floating point multiplier is implemented on Vertix5 FPGA. The floating point multiplier require two inputs each of length 32bit and output is of length 64bit.Since the size of I/O controllers on the FPGA kit are not sufficient. So we use chip-scope pro analyzer to observe the simulation results.

SIMULATION RESULTS

By using cadence RTL compiler:

A=-18.0= 1 10000011 00100000000000000000000
= C1900000
B=9.5=010000010 00110000000000000000000
= 41180000
AXB = -171.0 = 110000110 0101011000000000000000000000
00 00000000 000000 = C32B 0000 0000 00

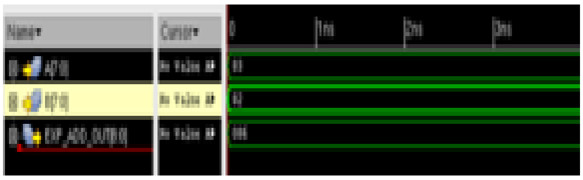


Output of sub modules:

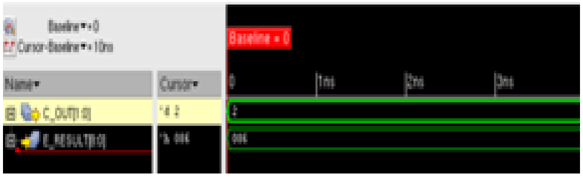
1. Sign:

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

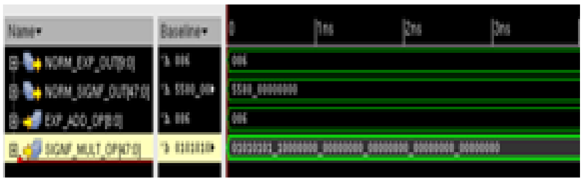
2. Exponent adds:



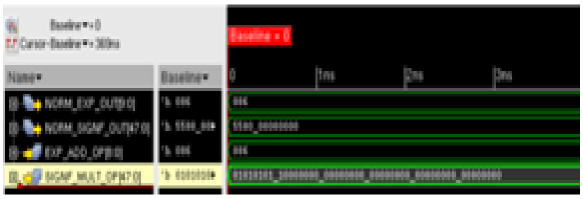
3. under/over flow:



4. Multiplier:



5. Normalizer:



Simulation by using virtex5 FPGA:

A = -18.0 = 1 10000011

00100000000000000000000 = C1900000

B = 9.5 = 0 10000010

00110000000000000000000 = 41180000

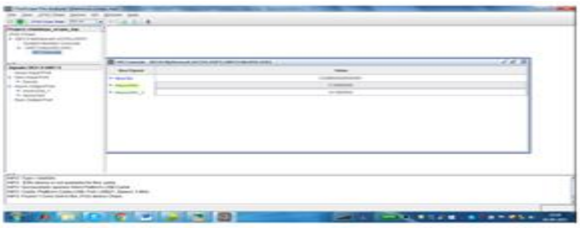
AXB = -171.0

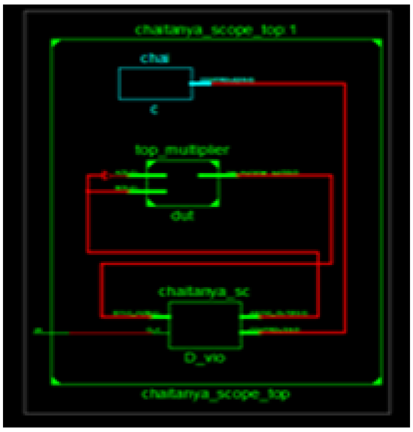
1 10000110

0101011000000000000000000000000

00000000 000000

= C32B 0000 0000 00





RTL diagram of floating point multiplier

Parameter	Existing Design	proposed Design
1.Area	46656(um²)	45548(um²)
2.power	7.5(mw)	6.0(mw)
3.Timing	28982(ps)	20702(ps)
4.Gates	46656.086	45548
5. Number of cells	1979	1941

CONCLUSION & FUTURE SCOPE

In image and signal processing applications floating-point multiplication is major concern in calculations. Performing multiplication on floating point data is a long course of action and requires huge quantity of processing time. By improving the speed of multiplication task overall speed of the system can be enhanced. The Bottleneck in the floating point multiplication process is the multiplication of mantissas which needs 24*24 bit multiplier for single precision floating point numbers. By improving the speed of multiplication task the overall speed of the system is be improved. Both the designs are compared in terms of area and power and the physical design is

implemented. The proposed design is highly suitable for performing digital signal processing applications.
The present work on the multiplier architecture can be extended in various directions as to enhance the performance higher order compressors 7:2, 9:2, can be used to accumulate partial products. A double precision IEEE Floating point Multiplier can be designed for multiplication intensive applications, such as DSP or graphics, could benefit several high performance multipliers on same chip. A high throughput multiplier or several multipliers working on same chip can be used for single chip video signal processing.

REFERENCES

1. Ushered G ECE-VLSI, R Hannibal ECE-VLSI and Dr Sara kumar sahoo ECE-VLSI, VIT University, Vellore, Tamil Nadu, India, “VLSI Implementation of a High Speed Single Precision Floating Point Unit Using Verilog”

2. Rudolf Usselmann, “Open Floating Point Unit, The Free IP Cores Projects”.

3. Edvin Catovic, Revised by: Jan Andersson, “GRFPU – High Performance IEEE754 Floating Point Unit”, Gaisler Research, Första Långatan 19, SE413 27 Göteborg, and Sweden.

4. David Goldberg, “What Every Computer Scientist Should Know About Floating-Point Arithmetic”, ACM Computing Surveys, Vol 23, No 1, March 1991, Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, California 94304.

5. Ronak Bajaj, Saransh Chabra, Sreehari Veeramachaneni, M B Srinivas, “A Novel, Low-Power Array Multiplier Architecture”, 2002

6. W.C. Yeh and C.W.jen , “High Speed Booth Encoded parallel Multiplier Design “IEEE Trans,.

7. Anna Jain, Baisakhy Dash, Ajit Kumar Panda, Member, IEEE, Muchharla Suresh, Member, IEEE, “FPGA design of a fast 32-bit floating point multiplier unit,” IEEE 15-16 March 2012.

8. IEEE 754-2008, IEEE Standard for Floating-Point Arithmetic, 2008

9. Mohamed Al-Astray Mentor Graphics, Ashtray Salem Cairo, Egypt and Wagdy Anis Communications and Electronics Engineering Aim Shams University Cairo, Egypt “An Efficient Implementation of Floating Point Multiplier”

10. Al-Ashrafy, M. Salem, A. Anis, W., Mentor Graphics, Cairo, Egypt, “An efficient implementation of floating point multiplier,” IEEE 24-26 April 2011

11. D. Radhakrishnan, A.P. Preethy, ―Low Power CMOS Pass Logic 4-2 Compressor for High-Speed Multiplication,‖ Proc. IEEE Midwest Symp. on Circuits and Systems, 2000, pp. 1-3.

12. Chip-Hong Chang, Jiangmin Gu and Mingyan Zhang, ―Ultra Low-Voltage Low-Power CMOS 4-2 and 5-2 Compressors for Fast Arithmetic Circuits,‖ IEEE Transactions on circuits and systems, Vol. 51, No. 10, 2004, pp. 1985-1997.

13. N. Shirazi, A. Walters, and P. Athanas, “Quantitative Analysisof Floating Point Arithmetic on FPGA Based CustomComputing Machines”, Proceedings of the IEEE Symposiumon FPGAs for Custom Computing Machines (FCCM’95), Pp.155–162, 1995.