International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-1S5, December 2019

Merged Floating Point Multipliers

Mrudula Singamsetti, Sadulla Shaik, T. Pitchaiah

Abstract: ***Floating point multipliers are extensively used in many scientific and signal processing computations, due to high speed and memory requirements of IEEE-754 floating point multipliers which prevents its implementation in many systems because of fast computations. Hence floating point multipliers became one of the research criteria. This research aims to design a new floating point multiplier that occupies less area, low power dissipation and reduces computational time (more speed) when compared to the conventional architectures. After an extensive literature survey, new architecture was recognized i.e, resource sharing Karatsuba –Ofman algorithm which occupies less area, power and increasing speed. The design was implemented in mat lab using DSP block sets, simulator tool is Xilinx Vivado.***

Keywords: Floating point multiplier, Booth Multipliers, Dadda Multiplier, Resource sharing Karatsuba algorithm ,Wallace Multiplier

I. INTRODUCTION

IEEE-754 has standard format for representing the floating point numbers i.e. basic formats and extended format. Three of them are extensively used in computer hardware. These formats are classified as Half Precision(HP), Single Precision(SP), Double Precision (DP), Double Extended Precision (DEP), Quadruple precision (QP) shown in fig 1,fig2,fig3,fig4.

Normalised Floating point number representation = $-1^S \times M \times 2^E$ ….. (1)

The formulae for representing Exponent Bias $=2^{p-1}-1$ ….. (2)

p is the no.of bits in the exponent

For example

The number of exponent bits p=5 for half precision. Hence Exponent Bias =16.

Sign	Exponent	Mantissa	Total
1	5	10	16

Fig1: Half Precision

Sign	Exponent	Mantissa	Total
1	8	23	32

Fig2: Single Precision

Manuscript published on 30 December 2019.
***** Correspondence Author (s)
Mrudula Singamsetti, Department of Electronics & Communication Engineering
Vignan's Foundation for Science, Technology and Research, Guntur-522213, A.P, India. mrudulasingamsetti@gmail.com
Sadulla Shaik,T.Pitchaiah Department of Electronics & Communication Engineering
Vignan's Foundation for Science, Technology and Research, Guntur-522213, A.P, India. sadulla09@gmail.com
T.Pitchaiah, Department of Electronics & Communication Engineering
Vignan's Foundation for Science, Technology and Research, Guntur-522213, A.P, India. tpvignan@gmail.com

Sign	Exponent	Mantissa	Total
1	11	52	64

Fig3: Double Precision

Sign	Exponent	Mantissa	Total
1	15	112	128

Fig4: Quadruple Precision

Procedure for Floating Point Multiplication with the help of one example:

1.Sign of the multiplication operation is done by exoring the sign bit of input operands

Sign= $S_1 \oplus S_2$ ….. (3)

2.Adding the Exponents

Exponent= $E_1 + E_2$ -Bias ….. (4)

3.Mantissa multiplication

In floating point multiplication, the main concentration is on mantissa multiplication because for larger widths, which requires more partial products, larger widths additions. Due, to which it occupies larger area, more power dissipation, larger delay. By using Karatsuba algorithm, the above mentioned problems can overcome.

II. LITERATURE SURVEY

In digital design, there are several multipliers like series multiplier, parallel multiplier, Array multiplier, Booth multiplier, Dadda multiplier, Wallace Tree multipliers.

A. Booth Multipliers

The booth multipliers generate the product for two negative numbers. By using booth encoding technique there is a reduction in the partial products to generate the multiplication. The total delay of the multiplier depends on log (word length) [10].

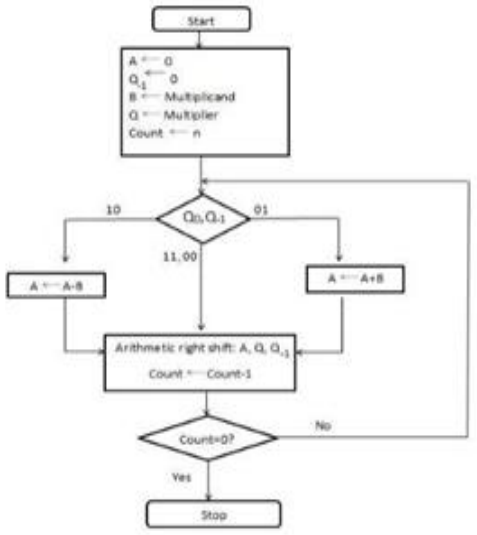

Fig 2: Booth Encoding Algorithm

B. Dadda Multiplier

The dadda 8*8 multiplier [11] reduces the number of adder blocks of partial products which is shown in Fig 3. This can be done by including either half adder or full adder at summing stages. This is slighter faster with few additional hardware required

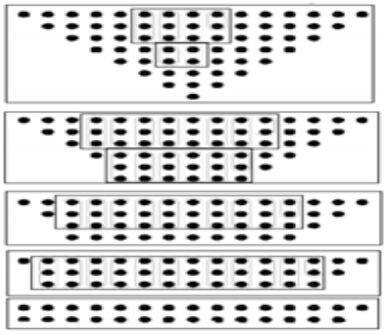

Fig 3:8*8 Dadda multiplier

C. Wallace Multiplier

Wallace multiplier is a necessary multiplier in low power VLSI circuits. The Wallace tree [12] consists of an AND array, a carry save adder for adding partial products and a final adder for at the final stage of addition. For adding partial products, we use Ripple Carry Adder(RCA) but the delay is large at the last stage of the adder. Hence it is replaced by carry save adder. After comparing [13] Dadda multiplier, Booth Encoding Algorithm, Wallace tree multiplier, it was concluded that Wallace tree multiplier has the lowest power dissipation and fastest among all multipliers.

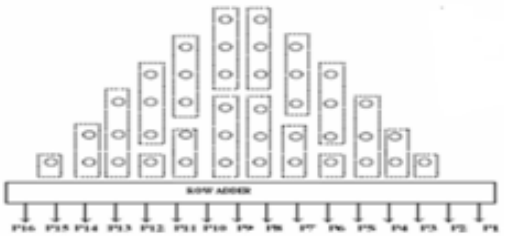

Fig 4: 8-bit High speed Wallace tree multiplier

There are so many fast multiplication algorithms for immense multipliers [5] are

1. Complex multiplication algorithm
2. Karatsuba Multiplication

D. Complex multiplication algorithm

There are 4 multiplications and two additions are required for the below mentioned example

(a+bi) (c+di) = (ac-bd) + (bc+ad)i …... (5)

Yet, there is an alternative way for decreasing [14] the number of multiplications to three

$k_1 = c \cdot (a + b)$ …... (6)

$k_2 = a \cdot (d - c)$ …... (7)

$k_3 = b \cdot (c + d)$ …... (8)

Real part = $k_1 - k_3$ …... (9)

Imaginary part = $k_1 + k_2$ .... (10)

The equation 5 is replaced with sum of equation 9 and equation 10. The algorithm utilizes 3 multiplications, rather than 4, and 4 additions or subtractions desirably 2.Suppose a multiplier is further costlier than 3 additions or subtractions, by school book method then there is increase in speed. On present day computers a multiply and an add and will calculate in the same time. There is reciprocation that there is loss of precision when utilizing floating point.

E. Karatsuba multiplication

This algorithm is used for large input widths, multiplication of immense multipliers are very slow. In1962 a new algorithm called Karatsuba multiplication was discovered. The advantage of Karatsuba multiplier is 2-digit multiplication requires 3 multiplications instead of 4 multiplications by schoolbook method.This can be done by utilizing a divide and ruling technique.Assume we want to multiply two 4 digit numbers:

X_3 X2|X1X0

$Y_3Y_2|Y_1Y_0$

A=X_3 X_2, B= X1X0

C=Y_3Y_2, D=Y_3Y_2

1. calculate X=A.C
2. calculate Y=B.D
3. calculate Z=(A+B). (C+D)
4. calculate *Z-Y-X*, call the result as *K*; this number is equal to $x_1 \cdot y_2 + x_2 \cdot y_1$
5. Hence we are performing 4-bit multiplication we have to padd 4 zeros at the trailing edge to the values of AC, that is added with the BD, call the result as L.
6. Half of the input bit width number of zeros have to be added to the trailing edge of K

The above mentioned algorithms are simulated and synthesized using Xilinx ISE Design suite. Which will occupy more slices and combinational path delay is more. Hence to achieve better performance in terms of speed, power and delay optimization, we are utilizing DSP blocksets in matlab Simulink to achieve speed and less power dissipation as well for area and logic utilization we taking LUTs in to consideration. [1]

Most of the processors supports QP or DP or SP which performs scientific computations This paper supports merged FP multiplier which also supports quadruple precision or two parallel double precision or twin SP. since the area required for quadruple precision is too costly. Hence we represented the study on merged floating point multiplier which requires less hardware and less delay when compared with individual precision multiplier i.e., quadruple precision requires 3 clock cycles where as two parallel double precision requires 2 clock cycles for implementing quadruple precision. After estimating twin DP multipliers operating concurrently, the author presented a design that has 5% increase in area and 30% increase in delay, but the design is more capable to perform QP multiplications much faster.

QP floating point multiplier [2], supports twin parallel DP multiplications because single Double precision multiplier is performing poor in terms of accuracy for many scientific computations. This technique is used for single quadruple precision or dual mode DP floating point multiplier that carries out a DP multiplication or twin SP multiplications in collateral. Synthesis outputs displays that twin mode DP requires 43% less area compared to the traditional floating point multiplier.

IEEE 754 standard floating point multiplier furnishes most precise computations to attain less area and throughput on the IC have been become better by introducing the pipelining technique[3]. To implement pipelining, subdivide the input process each of which is executed concurrently without additional computing units. In this paper author implemented both un pipelined and pipelined floating point multiplier, simulated and synthesized using VHDL code, and the target device is Xilinx Virtex-II FPGA. By increasing the pipeline stages successfully increasing the operating frequency device utilization and power dissipation is reduced, farther the speed of the output increase from 15.671 MHz to 325.733 MHz Hence throughput increases.

FPGAs trademarks Xilinx DSP block set in small-scale embedded multiplier for example SP (24). An immense multiplier for specimen DP (53) multiplier requires many DSP blocks. In this paper the author discusses 3 nonstandard algorithms for immense multiplier: The Karatsuba-Ofman algorithm, nonstandard multiplier tiling, and specialized squares. In this paper the author discusses the technique to reduce the DSP block usage. Any multiplier occupies more than 2 DSP block usage, there are different ways for deducing DSP block usage, the simplest one is to reduce logical blocks. LUT deploying multipliers has an immense LUT cost (at least n^2 LUTs for n-bit numbers, the flip-flops for pipelined implementations), performance cost: long latency and slow clock. For Two-part splitting and three-part splitting for mantissa multiplication using Karatsuba technique mentioned below. However, some multipliers require a part of the DSP block. Hence their focus is on deduction of DSP cost and to reduce the LUTs cost in terms of latency if any, they permit immense multipliers for functioning at the maximum frequency while decreasing DSP block usage The immense multipliers has implemented by using Karatsuba ofman algorithm[4]. In this algorithm number of multipliers are reduced at the cost of additions. Hence the DSP cost for an immense multiplier are deducted from 4 to 3,9 to6,16to 10. These multipliers have implemented on DSP-enhanced FPGA from Xilinx or Altera, but this is less economically on more recent chips, which are less flexible. Hence a tiling technique was introduced that enlarges the space on vertex5 devices both for SP and DP multipliers i.e., the circuits are designed by using rectangular multipliers. Finally, they designed computations of squares. Squaring is most commonly used in fast computations, polynomial calculations, statistical computations.

A twin mode floating point multiplier architecture, supports DP or twin SP multiplication with efficient resource sharing and hence leads to minimum circuitry [6]. Compared to the single mode double precision DPDSP (Double precision with dual mode single precision) architecture needs 11.6% more area and 6.74% more than SP (Single precision).

III. RESULT DESCRIPTION

The utmost 3 familiar algorithms specifically, Booth ;Normal Karatsuba; Recursive Karatsuba multipliers have been implemented[7]. Both SP and DP mantissa multiplication has been performed on the above mentioned algorithms and obtained the computed results on the reconfigurable device. The two main criteria when evaluating the performance is FPGA resources and processing speed. The Recursive Karatsuba algorithm achieves better than normal Karatsuba and Booth multiplier.

Table1: Comparison of Mantissa Multiplier

Parameter	Array	Wallace Tree	Daddas	Bau gh Wooley	Vedic	Karatsuba
Speed	Very Low	Low	Low	Highest	High	Very High
Time Delay	Too High	High	Moderate	Very Less	Less	Less
Area	Less	Less	Less	Medium	Maximum	Maximum
Complexity	Less	Less	Moderate	More	Most	Less
Power Consumption	Most	More	Moderate	Very Less	Less	Less

An area and power efficient repetitive floating point multiplier architecture and implemented on a FPGA device with pipelined operations. This architecture reinforces both SP and DP operations[8]. The precision can be altered during runtime by selecting the precision signal. There are different steps involved in the conventional floating point multiplier architecture

1.Deciding the sign of the product by exoring the sign of the multiplicand and multiplier.
2.Adding the exponent bits of the input operands
3.Mantissa Multiplication

From the above steps it is decided to choose efficient adder and multiplier which occupies less area and power consumption. Hence Karatsuba algorithm is used for mantissa multiplication to decrease the number of DSP blocks. For performing DP functions repetitive blocks are used which occupies less hardware and decreases the power consumption than a conventional DP multiplier. In order to decrease the power consumption more, the unutilised blocks are deactivated. Compared to the conventional multiplier the proposed design has 33% decrease in the DSP units ,4.3% reduction in LUT's (Look up tables) and 31.2% decrement in the utilization of

flip-flops and achieves 4% faster clock frequency on Virtex-5 devices.

In 32-bit complex floating point (FP) multiplier utilizing 4 FP real multipliers with less path delay[9]. The whole 32-bit real FP multiplier requires a 24-bit fixed point real multiplier for mantissa multiplication. complex floating point Vedic multiplier is two times quicker in contrast to Booth and array multipliers yet there is an increase in power consumption.

Table2: Comparison of the above Floating point multiplication algorithms

Ref .No	Type	Multiplication Technique	Area	Delay(ns) /(Number of clk cycles)	Speed(MHz)
[1]	QP&DP	-	5%>QP using DP	30%>QP using DP	QP-163
					DP-230
[2]	Dual mode-DP,QP	Karatsuba Algorithm	43%<conventional DP	25%> conventional DP	169
			17%> Dual mode DP	51%> Dual mode DP	230
[3]	SP	Un Pipelined	21% of LUTs utilised	63.812ns	15.67
		Pipelined	3% of LUTs utilised	1.26ns	325.733
[4]	DP	KSA-2	3 DSP+48 LUTs utilised	3 Clock cycles	317
		KSA-3	6 DSP+159 LUTs utilised	3 Clock cycles	317
[8]	SP DP	KSA	2DSP+1% of LUT utilized	1.00ns	438.8

IV. CONCLUSION

In this paper, we have furnished an overview of the challenges faced and solutions on floating point multiplication algorithms. From the investigation it was observed that conventional QP, DP, SP multipliers is occupying more area, delay and high power consumption compared to the merged floating point multipliers. In order to attain high speed, we are combining DSP blocks with floating point multipliers. To obtain the overall performance we can use DSP blocks with merged floating point multiplier.

REFERENCES

1. Ahmet, Michael J. "A Quadruple Precision and Dual Double Precision Floating-Point Multiplier",Proceedings of the Euromicro Symposium on Digital System Design (DSD'03) 0-7695-2003-0/03 © 2003 IEEE
2. Ahmet. Akkas, Michael. J.Schulte," Dual-mode floating-point multiplier architectures with parallel operations",Journal of Systems Architecture:TheEuroMicroJournal,Volume 52 Issue 10,October 2006,pages 549-562
3. KavitaKhare, KhareNilay, RPSingh" Comparison of pipelined IEEE-754 standard floating point multiplier with un pipelined multiplier," Journal of Scientific &Industrial Research, November 2006, pages:900-904
4. DinechinFlorent de, Pasca Bogdan, "Large Multipliers with fewer DSP block", proceedings of ,31 Aug.-2 Sept. 2009,**ISBN:** 978-1-4244-3892-1,pp-250-255
5. *Karatsuba.A ,Ofman Yu. (1962). "Multiplication of Many-Digital Numbers by Automatic Computers". Proceedings of the USSR Academy of Sciences. 145: 293–294. Translation in the academic journal* Physics-Doklady, *7 (1963), pages: 595–596*
6. Manish Kumar Jaiswal and Hayden K.-H So, "Dual-Mode Double Precision / Two-Parallel Single Precision Floating Point Multiplier Architecture", Proceedings of IEEE International Conference on Very Large Scale Integration (VLSI-SoC) 2015, pages:213-218
7. Ravi Kishore Kodali, Lakshmi Boppana, Satya Kesav. Gundabathula,"FPGA Implementation of IEEE-754 Floating Point Karatsuba Multiplier",Proceedings of 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT) Pages:300-304
8. Hao Zhang1, Dongdong Chen2, Seok-Bum Ko1,"Area- and power-efficient iterative single/ double-precision merged floating-point multiplier on FPGA",IET Computers & Digital Techniques,22nd May 2017 doi: 10.1049/iet-cdt.2016.0100 www.ietdl.org, pages 149-158.
9. Deergha Rao. KK, P.V. Muralikrishna, Ch. Gangadhar , (2018)," FPGA Implementation of 32 Bit Complex Floating Point Multiplier Using Vedic Real Multipliers with Minimum Path Delay", Proceedings of 2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering,Pages:1-6

10. Marc.Hunger, Daniel..Marienfeld, "New Self-Checking Booth Multipliers", International Journal of Applied Mathematics Computer Sci.,2008, Vol. 18, No. 3, 319–328
11. L. Dadda, "Some schemes for parallel multipliers," Alta Frequenza, vol. 34, pp. 349-356, August 1965.
12. D. J. Kinniment, "An evaluation of asynchronous addition", IEEE transaction on very large scale integration (VLSI) systems, vol.4, pp.137-140, March 1996.
13. DeepikaPurohit, HimanshuJoshi,"Comparative Study and Analysis of Fast Multipliers", International Journal of Engineering and Technical Research (IJETR) ISSN: 2321-0869, Volume-2, Issue-7, July 2014
14. Knuth, E.Donald, *(1988)*, The Art of Computer Programming volume 2: Seminumerical algorithms, Addison-Wesley, *pp. 519, 706*