



# International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: [www.ijarcsse.com](http://www.ijarcsse.com)

## A Novel Efficient VLSI Architecture for IEEE 754 Floating Point Multiplier using CSLA with BEC

Iteesha Pandagre, Prof. Manish Gurjar

Dept. of Electronics and Communication, TIT Group of Institutions,  
Bhopal, M.P., India

**Abstract**— Due to advancement of new technology in the field of VLSI and Embedded system, there is an increasing demand of high speed and low power consumption processor. Speed of processor greatly depends on its multiplier as well as adder performance. In spite of complexity involved in floating point arithmetic, its implementation is increasing day by day. Due to which high speed adder architecture become important. Several adder architecture designs have been developed to increase the efficiency of the adder. In this paper, we introduce an architecture that performs high speed IEEE 754 floating point multiplier using modified carry select adder (CSA). Modified CSA depend on booth encoder (BEC) Technique. Booth encoder, Mathematics is an ancient Indian system of Mathematics. Here we are introduced two carry select based design. These designs are implementation Xilinx Vertex device family.

**Keywords**— IEEE754, Single Precision Floating Point (SP FP), Double Precision Floating Point (DP FP), Binary to Excess-1 Converter.

### I. INTRODUCTION

The real numbers represented in binary format are known as floating point numbers. Based on IEEE-754 standard, floating point formats are classified into binary and decimal interchange formats. Floating point multipliers are very important in dsp applications. This paper focuses on double precision normalized binary interchange format. Figure 1 shows the IEEE-754 double precision binary format representation. Sign (s) is represented with one bit, exponent (e) and fraction (m or mantissa) are represented with eleven and fifty two bits respectively. For a number is said to be a normalized number, it must consist of 'one' in the MSB of the significand and exponent is greater than zero and smaller than 1023. The real number is represented by equations (i) & (2).

$$Z = (-1^s) \times 2^{(E-Bias)} \times (1.M) \tag{1}$$

$$Value = (-1^{signbit}) \times 2^{(Exponent-1023)} \times (1.Mantissa) \tag{2}$$

Biasing makes the values of exponents within an unsigned range suitable for high speed comparison.

Sign Bit	Biased Exponent	Significand
1-bit	8/11-bit	23/52-bit

Figure 1: IEEE 754 Single Precision and Double Precision Floating Point Format

#### o IEEE 754 STANDARD FLOATING POINT MULTIPLICATION ALGORITHM

A brief overview of floating point multiplication has been explained below [5-6].

- Both sign bits  $S_1, S_2$  are need to be Xoring together, then the result will be sign bit of the final product.
- Both the exponent bits  $E_1, E_2$  are added together, then subtract bias value from it. So, we get exponent field of the final product.
- Significand bits  $Sig_1$  and  $Sig_2$  of both the operands are multiply including their hidden bits.
- Normalize the product found in step 3 and change the exponent accordingly. After normalization, the leading “1” will become the hidden bit.

Above algorithm of multiplication algorithm is shown in Figure 2.

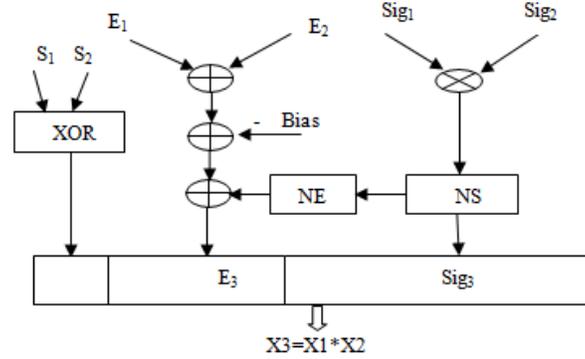


Figure 2: IEEE754 SP FP and DP FP Multiplier Structure, NE: Normalized exponent, NS: Normalized Significant NS: Normalized Significant

## II. MODIFIED CSA

Figure 3 shows schematic of the 8-bit modified CSA using dual RCA. The structure consists of three groups with different bit size dual RCA. Figure 4 shows the group 2 of the CSA in details and calculation of area (cost) and delay. The delay and area evaluation methodology considers all gates to be made up of AND, OR, and Inverter (AOI), each having delay equal to 1 unit and area equal to 1 unit. We then add up the number of gates in the longest path of a logic block that contributes to the maximum delay. The area evaluation is done by counting the total number of AOI gates required for each logic block. Based on this approach, the CSLA adder blocks of 2:1 mux, Half Adder (HA), and FA are evaluated and listed in Table I.

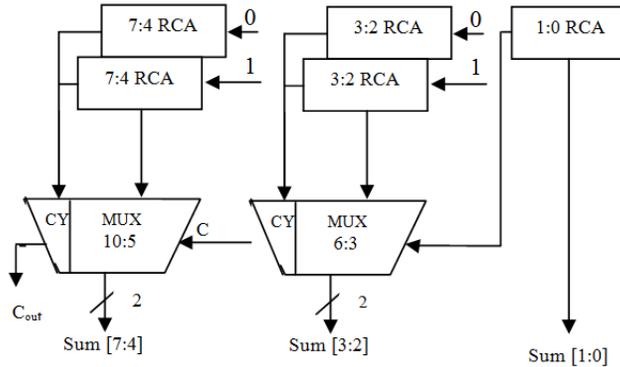


Figure 3: 8-bit Modified CSA with dual RCA

Table 1: Delay and Area Count of the Basic Blocks of CSLA

Adder Blocks	Delay	Area
XOR	3	5
2:1 MUX	3	4
Half adder	3	6
Full adder	6	13

Figure 5 shows schematic of the 16-B SQRT CSA using BEC (Binary to excess-1 converter) instead of the RCA with cin = 1.

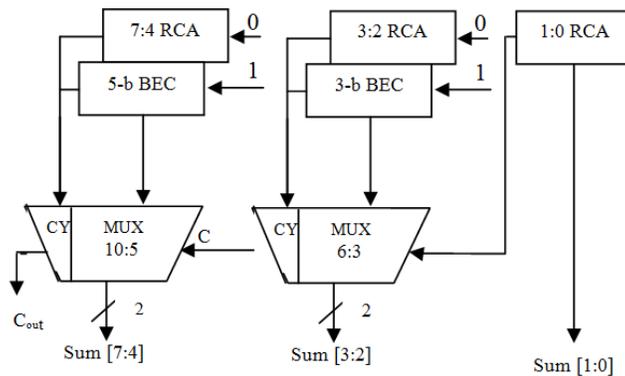


Figure 5: 8-bit Modified CSA with RCA and BEC

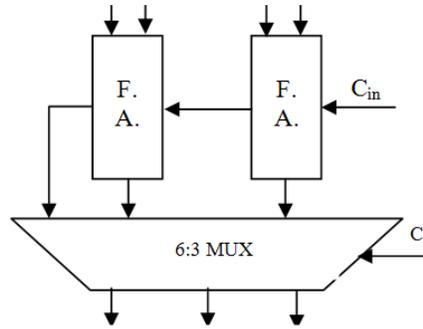


Figure 4: Basic structure of 6:3 MUX

The structure consists of three groups with different bit size RCA and BEC. The group 2 has one 2-B RCA which has 1 full adder (FA) and 1 half adder (HA) for  $C_{in} = 0$  and a 3-B BEC is used instead of another 2-b RCA with  $C_{in} = 1$ . In similar manner the area and power delay can be calculated for each group. Figure 6 shows the group 2 of the BEC in detail and calculation of area and delay.

Booth encoding is a techniques to reduce the number of partial products in n-bit encoder. Booth encoder change the binary to excess-1 converter is used to reduce the area and power consumption in CSA. Figure 6 shows the basic structure of 3-b BEC.

The Boolean expressions of the 3-b BEC is as

$$X0 = \sim B0 \tag{3}$$

$$X1 = B0 \wedge B1 \tag{4}$$

$$X2 = B2 \wedge (B0 \& B1 \& B2) \tag{5}$$

Table II: Function Table of 3-bit Booth Encoder

Binary[2:0]	Excess- 1[2:0]
000	001
001	010
010	011
...	...
111	000

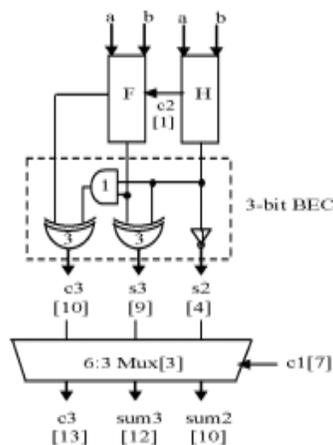


Figure 6:3-bit Booth Encoder

Table III: Comparison of the delay and cost of the 8-bit modified CSA with dual RCA and 8-bit modified CSA with RCA and BEC

Group	Previous Work		Proposed Work	
	Delay	Area	Delay	Area
Group 1	12	19	12	19
Group 2	21	57	23	43
Group 3	39	117	42	89

### III. PROPOSED DESIGN

In IEEE754 standard floating point representation, 8 bit Exponent field in single precision floating point (SP FP) representation and 11 bit in double precision floating point (DP FP) representation are need to add with another 8 bit exponent and 11 bit exponent respectively, in order to multiply floating point numbers represented in IEEE 754 standard as explained earlier. Ramesh et al [1] has used parallel adder for adding exponent bits in floating point multiplication algorithm. We proposed the use of 8-bit modified CSA with dual RCA and 8-bit modified CSA with RCA and BEC for adding the exponent bits. We have found the improved area of 8-bit modified Carry select adder with RCA and BEC over the 8-bit modified CSA with dual RCA.

o Sign bit calculation

To calculate the sign bit of the resultant product for SP FP and DP FP multiplier, the same strategy will work. We just need to xoring together the sign bits of both the operands. If the resultant bit is '1', then the final product will be a negative number. If the resultant bit is '0', then the final product will be a positive number.

o Exponent bit calculation

Add the exponent bits of both the operands together, and then the bias value (127 for SPFP and 1023 for DPFP) is subtracted from the result of addition. This result may not be the exponent bits of the final product. After the significand multiplication, normalization has to be done for it. According to the normalized value, exponents need to be adjusted. The adjusted exponent will be the exponent bits of the final product.

o Significand bit calculation

Significand bits including the one hidden bit are need to be multiply, but the problem is the length of the operands. Number of bits of the operand will become 24 bits in case of SP FP representation and it will be 53 bits in case of DP FP representation, which will result the 48 bits and 106 bits product value respectively. In this paper we use the technique of break up the operands into different groups then multiply them. We get many product terms, add them together carefully by shifting them according to which part of one operand is multiplied by which part of the other operand. We have decomposed the significand bits of both the operands in four groups. Multiply each group of one operand by each group of second operand. We get 16 product terms. Then we add all of them together very carefully by shifting the term to the left according to which groups of the operands are involved in the product term.

### IV. SIMULATION RESULT

All the designing and experiment regarding algorithm that we have mentioned in this paper is being developed on Xilinx 6.2i updated version. Xilinx 6.2i has couple of the striking features such as low memory requirement, fast debugging, and low cost. The latest release of ISE™ (Integrated Software Environment) design tool provides the low memory requirement approximate 27 percentage low. ISE 6.2i that provides advanced tools like smart compile technology with better usage of their computing hardware provides faster timing closure and higher quality of results for a better time to designing solution. These designs were compared with IEEE-754 floating point multiplier architecture proposed by Ragini *et al.* [10] to show for the improvements obtained.

So Ragini et al. [10] architecture is best in all these architectures. Implementing the Ragini et al. [10], proposed architecture IEEE-754 floating point design has been captured by VHDL and the functionality is verified by RTL and gate level simulation. To estimate the number of slice, number of 4-i/p LUTs and maximum combinational path delay (MCPD). Comparison of Xilinx result is given in Table IV-V respectively.

Table IV: Comparisons result for Exponent Adder used in SPFP and DPFP Multiplier

Design	Exponent Adder	No. of slices	No. of 4 input LUTs	MCPD (ns)
DP FP (Ragini et al.)	Carry Skip adder	16	27	17.977
	Parallel adder	13	22	20.253
	Carry select adder	13	23	15.573
Proposed Work	Modified CSA with dual RCA	17	29	15.341
	Modified CSA with RCA and BEC	11	21	13.547
SP FP (Ragini et al.)	Carry Skip adder	11	20	12.865
	Parallel adder	9	16	16.299
	Carry select adder	9	16	12.801
Proposed Work	Modified CSA with dual RCA	11	19	12.593
	Modified CSA with RCA and BEC	9	15	10.117

We have implemented the modified CSA with RCA and BEC for SP FP on vertexE whose RTL (Resistor Transistor Logic) view is shown as figure 7.

Table V: Device utilization summary (VertexE-XCV50e-8cs144) of SP FP multiplier using modified CSA with RCA and BEC

	SP FP Multiplier	
	Ragini et al.	Proposed Work
No. of Slices	353	325
No. of 4 input LUTs	610	615
No. of bounded IOBs	121	98
Maximum combinational path delay(in ns)	22.021	19.255

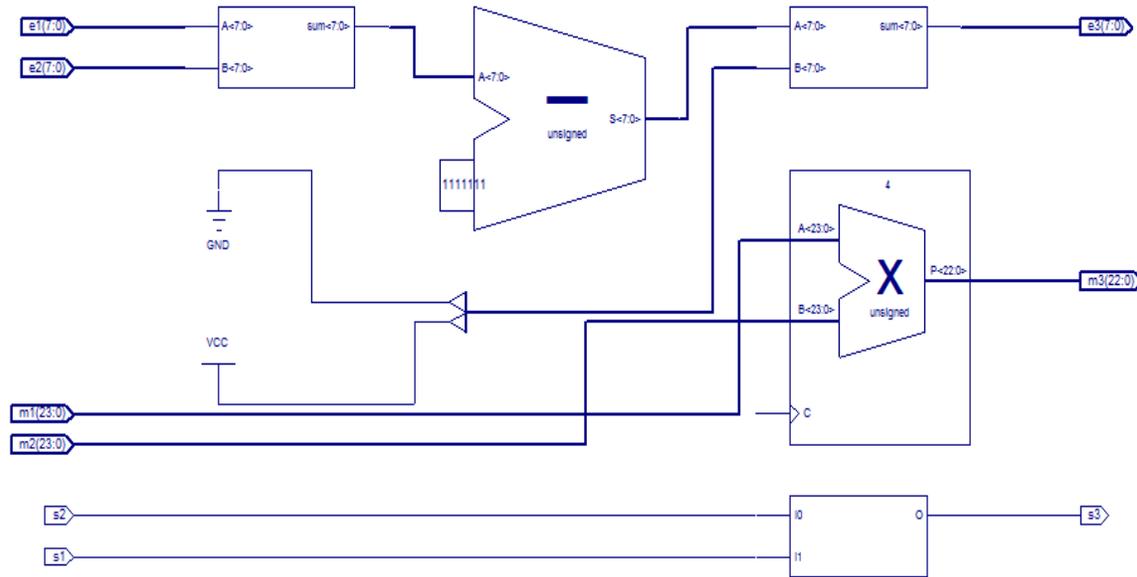


Figure 7: RTL view of SP FP Multiplier using modified CSA with RCA and BEC

## V. CONCLUSION

IEEE754 standardize two basic formats for representing floating point numbers namely, single precision floating point and double precision floating point. Floating point arithmetics has vast applications in many areas like robotics and DSP. Delay provided and area required by hardware are the two key factors which are need to be consider Here we present single precision floating point multiplier by using two different adders namely modified CSA with dual RCA and modified CSA with RCA and BEC.

Among all two adders, modified CSA with RCA and BEC is the least amount of Maximum combinational path delay (MCDP). Also, it takes least number of slices i.e. occupy least area among all two adders.

## REFERENCES

- [1] B. Fagin and C. Renard, "Field Programmable Gate Arrays and Floating Point Arithmetic," IEEE Transactions on VLSI, vol. 2, no. 3, pp. 365-367, 1994.
- [2] N. Shirazi, A. Walters, and P. Athanas, "Quantitative Analysis of Floating Point Arithmetic on FPGA Based Custom Computing Machines," Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines (FCCM'95), pp.155-162, 1995.
- [3] Malik and S. -B. Ko, "A Study on the Floating-Point Adder in FPGAs", in Canadian Conference on Electrical and Computer Engineering (CCECE-06), (2006) May, pp. 86–89.
- [4] D. Sangwan and M. K. Yadav, "Design and Implementation of Adder/Subtractor and Multiplication Units for Floating-Point Arithmetic", in International Journal of Electronics Engineering, (2010), pp. 197-203.
- [5] M. K. Jaiswal and R. C. C. Cheung, "High Performance FPGA Implementation of Double Precision Floating Point Adder/Subtractor", in International Journal of Hybrid Information Technology, vol. 4, no. 4, (2011) October.

- [6] L. Louca, T. A. Cook and W. H. Johnson, "Implementation of IEEE Single Precision Floating Point Addition and Multiplication on FPGAs", Proceedings of 83rd IEEE Symposium on FPGAs for Custom Computing Machines (FCCM'96), (1996), pp. 107–116.
- [7] Jaenicke and W. Luk, "Parameterized Floating-Point Arithmetic on FPGAs", Proc. of IEEE ICASSP, vol. 2, (2001), pp. 897-900.
- [8] Lee and N. Burgess, "Parameterisable Floating-point Operations on FPGA", Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems, and Computers, (2002).
- [9] M. Al-Ashrafy, A. Salem, W. Anis, "An Efficient Implementation of Floating Point Multiplier", Saudi International Electronics, Communications and Photonics Conference (SIEPC), (2011) April 24-26, pp. 1-5.
- [10] *Ragini Parte and Jitendra Jain, "Analysis of Effects of using Exponent Adders in IEEE- 754 Multiplier by VHDL", 2015 International Conference on Circuit, Power and Computing Technologies [ICCPCT] 978-1-4799-7074-2/15/\$31.00 ©2015 IEEE.*