DSP APPLICATIONS DESIGN OF ROUNDING BASED MULTIPLIER FOR SIGNED AND UNSIGNED DATA OPERATIONS

[1]Abdul Maqseed Shaik, [2]Srinivasa Reddy Dumpa, [3]Rajashekar Kakoju, [4]Ayuluri Sireesha
[1,2,3]Assistant Professor, [4]UG Student, [1,2,3,4]Department of Electronicsand Communication Engineering, Brilliant Grammar School Educational Society Group of Institutions Integrated Campus, Hyderabad, India

ABSTRACT

The great majority of contemporary computer systems use the IEEE Standard for Floating-Point Arithmetic (IEEE 754), which has been the industry standard for floating-point arithmetic for many years. Recently, posit (Type III unum), a new number representation format, has been suggested as an alternative to the widely used IEEE 754 arithmetic. John L. Gustafson says this new format may deliver superior accuracy utilizing equal or less bits and simpler hardware than current standard. This undergraduate thesis analyses and contrasts the innovative posit number format's qualities and properties with the accepted practice for floating-point numbers (floats). We concentrate on assessing if posits would be a good "drop-in replacement" for floats based on claims made in the literature. First we propose a low-level design for posit arithmetic multiplier using the Xilinx tool to generate synthesizable HDL code which helps in the case of only unsigned numbers of multiplication. Where as in the practical, we need to focus on both signed and un-signed numbers. So here we proposed a new technique called RoBA (Rounding Based Approximate) multiplier which helps in reducing the area, delay and power by 10%, 40% and 54% respectively. To conclude this work, we propose a low-level design for posit arithmetic (signed and un-signed) RoBA multiplier using the Xilinx tool to generate synthesizable HDL code. Designed using XilinxISE14.7 software.

Keywords: Computer arithmetic, Floating point, Posit number system, Numerical error, RoBA Multiplier.

INTRODUCTION

Figure following depicts the parameterized datapath of the suggested posit multiplier. Posit component extraction, mantissa multiplier, final adder and normalisation, posit component packing, and rounding are all included in the critical route. The sign and exponent are treated independently as well. A modified Booth multiplier with bit radix-4 is what the mantissa multiplier is. Two adjustments are made in the suggested design to prevent pointless signal tripping and lower power usage. The first modification is the production of the mantissa multiplier's control signal, which only enables the relevant portions of the multiplier. The breakdown of the mantissa multiplier is the second change, and the control signal controls each of the small portion.

Advantages of Project:

The multiplier structure was designed with Less area and it consumes less power

For completing the design of our multiplier, we need a control signal. We divide the 16-bit digit input into four parts as already mentioned above. To generate Control Signal for both the inputs, one has to identify the position in where the binary digit _1‘ makes its appearance. After discovering the position of 1, comparison of the divided subgroups and our input digit isperformed.

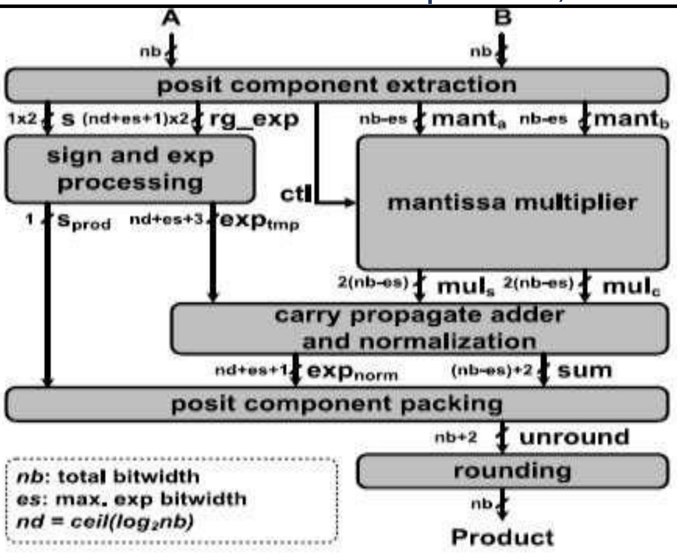


Fig.1. Flow for Proposed Posit multiplier

According to that grouping, we generate the control signal for both inputs. This in turn implies the selection of smaller multiplier which we need to use for the process. All in all we sum-up the use of control signal to select the smaller multiplier we have partitioned for the operation.

Existing System

The IEEE-754 standard describes how to store, format and compute with real-valued numbers. The current version supports five different widths, ranging from the small 16-bit half-precision format, all the way up to a 256-bit octuple precision format. The width of the format largely depends on the application at hand; deep-learning training is performed using single-precision while image manipulation often allows for a reduction in precision. Some applications, such as GROMACS, even mix different precisions for performance reasons.

Disadvantages:

It occupies more Area

It consumes more power

PROPOSED METHOD

In addition to the image and video processing applications, there are other areas where the exactness of the arithmetic operations is not critical to the functionality of the system. Being able to use the approximate computing provides the designer with the ability of making tradeoffs between the accuracy and the speed as well as power/energy consumption. Applying the approximation to the arithmetic units can be performed at different design abstraction levels including circuit, logic, and architecture levels, as well as algorithm and software layers. The approximation may be performed using different techniques such as allowing some timing violations (e.g., voltage over scaling or over clocking) and function approximation methods (e.g., modifying the Boolean function of a circuit) or a combination of them. In the category of function approximation methods, a number of approximating arithmetic building blocks, such as adders and multipliers, at different design levels have been suggested. In this paper, we focus on proposing a high-speed low power/energy yet approximate multiplier appropriate for error resilient DSP applications. The proposed approximate multiplier, which is also area efficient, is constructed by modifying the conventional multiplication approach at the algorithm level assuming rounded input values. We call this rounding-based approximate (RoBA) multiplier. The proposed multiplication approach is applicable to both signed and unsigned multiplications for which three optimized architectures are presented. The efficiencies of these structures are assessed by comparing the delays, power and energy consumptions, energy-delay products (EDPs), and areas with those of some approximate and accurate (exact) multipliers. The contributions of this paper can be summarized as follows: 1) presenting a new scheme for RoBA multiplication by modifying the conventional multiplication approach; 2) describing three hardware architectures of the proposed approximate multiplication scheme for sign and unsignedoperations.

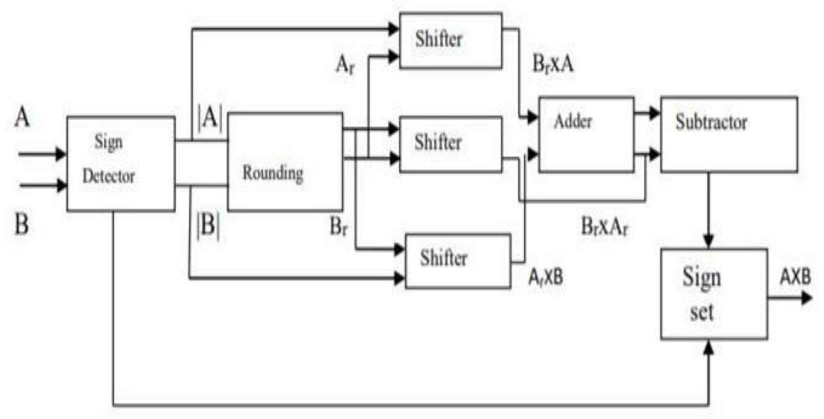
The main idea behind the proposed approximate multiplier is to make use of the ease of operation when the numbers are two to the power n (2n). To elaborate on the operation of the approximate multiplier, first, let us denote the rounded numbers of the input of A and B by Ar and Br, respectively. The multiplication of A by B may be rewritten as

$$A \times B = (A_r - A) \times (B_r - B) + A_r \times B + B_r \times A - A_r \times B_r.$$

The key observation is that the multiplications of Ar × Br, Ar ×B, and Br ×A may be implemented just by the shift operation. The hardware implementation of (Ar − A) × (Br − B), however, is rather complex. The weight of this term in the final result, which depends on differences of the exact numbers from their rounded ones, is typically small. Hence, we propose to omit this part, helping simplify the multiplication operation. Hence, to perform he multiplication process, the following expression is used:

$$A \times B \cong A_r \times B + B_r \times A - A_r \times B_r$$

In this approach, the nearest values for A and B in the form of 2n should be determined. When the value of A (or B) is equal to the 3 × 2p−2 (where p is an arbitrary positive integer larger than one), it has two nearest values in the form of 2n with equal absolute differences that are 2p and 2p−1. While both values lead to the same effect on the accuracy of the proposed multiplier, selecting the larger one (except for the case of p = 2) leads to a smaller hardware implementation for determining the nearest rounded value, and hence, it is considered in this paper.
It originates from the fact that the numbers in the form of 3 × 2p−2 are considered as do not care in both rounding up and down simplifying the process, and smaller logic expressions may be achieved if they are used in the rounding up. The only exception is for three, which in this case, two is considered as its nearest value in the proposed approximate multiplier.

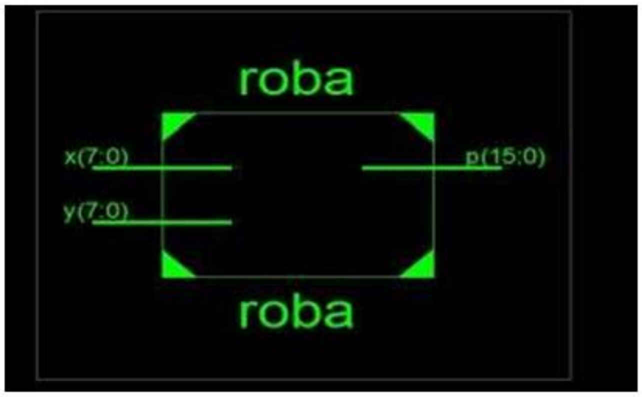


Block Diagram for the Hardware Implementation of the ROBA Multiplier

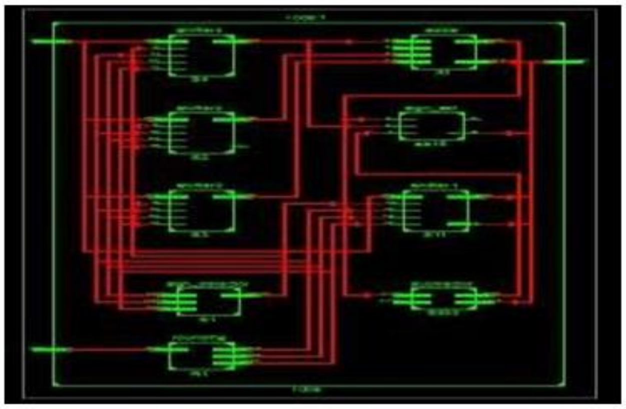
It should be noted that contrary to the previous work where the approximate result is smaller thanthe exact result, the final result calculated by the RoBA multiplier may be either larger or smallerthan the exact result depending on the magnitudes of Ar and Br compared with those of A and B, respectively. Note that if one of the operands (say A) is smaller than its corresponding rounded value while the other operand (say B) is larger than its corresponding rounded value, then the approximate result will be larger than the exact result. This is due to the fact that, in this case, the multiplication result of (Ar − A) × (Br − B) will be negative. Since the difference between them is precisely this product, the approximate result becomes larger than the exact one. Similarly, if both A and B are larger (or) both are smaller than Ar and Br, then the approximate result will be smaller than the exact result. Finally, it should be noted the advantage of the proposed RoBA multiplier exists only for positive inputs because in the two‘s complement representation, the rounded values of negative inputs are not in the formof 2n. Hence, we suggest that, before the multiplication operation starts, the absolute values of both inputs and the output sign of the multiplication result based on the inputs signs be determined and then the operation be performed for unsigned numbers and, at the last stage, the proper sign be applied to the unsigned result.

SIMULATION RESULTS

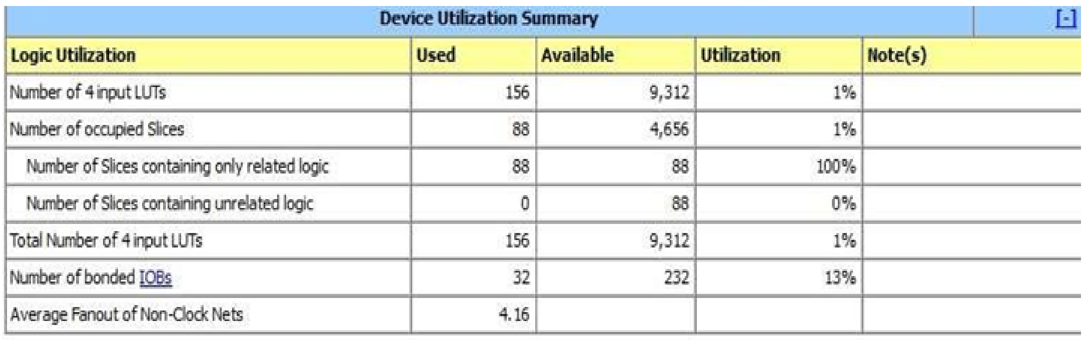
RTL SCHEMATIC



Internal block diagram

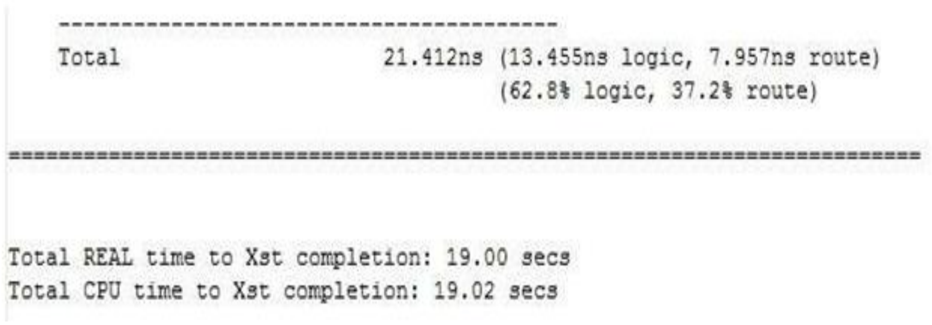


Area



Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	156	9,312	1%	
Number of occupied Slices	88	4,656	1%	
Number of Slices containing only related logic	88	88	100%	
Number of Slices containing unrelated logic	0	88	0%	
Total Number of 4 input LUTs	156	9,312	1%	
Number of bonded IOBs	32	232	13%	
Average Fanout of Non-Clock Nets	4.16			

Delay

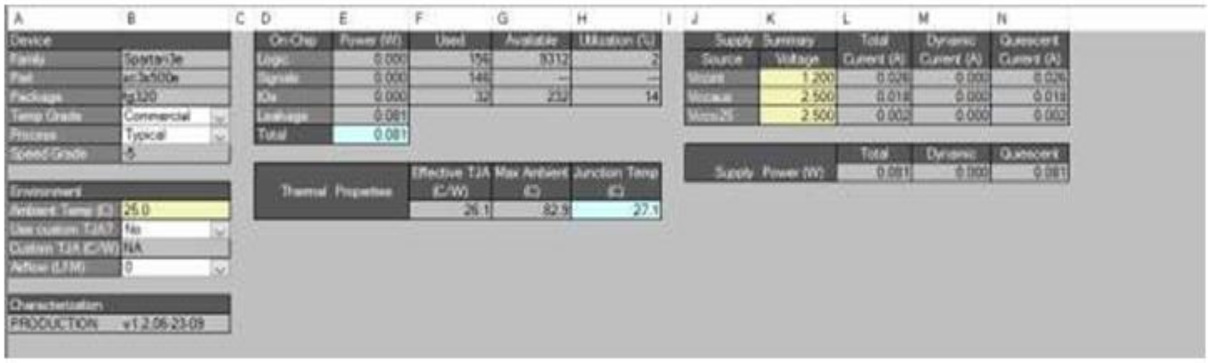


```
   ----------------------------------------
   Total                      21.412ns (13.455ns logic, 7.957ns route)
                                       (62.8% logic, 37.2% route)

=========================================================================


Total REAL time to Xst completion: 19.00 secs
Total CPU time to Xst completion: 19.02 secs
```

Power



Device	
Family	Spartan3e
Part	xc3s500e
Package	fg320
Temp Grade	Commercial
Process	Typical
Speed Grade	-5

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	0

Characterization	
PRODUCTION	v1.2,06-23-09

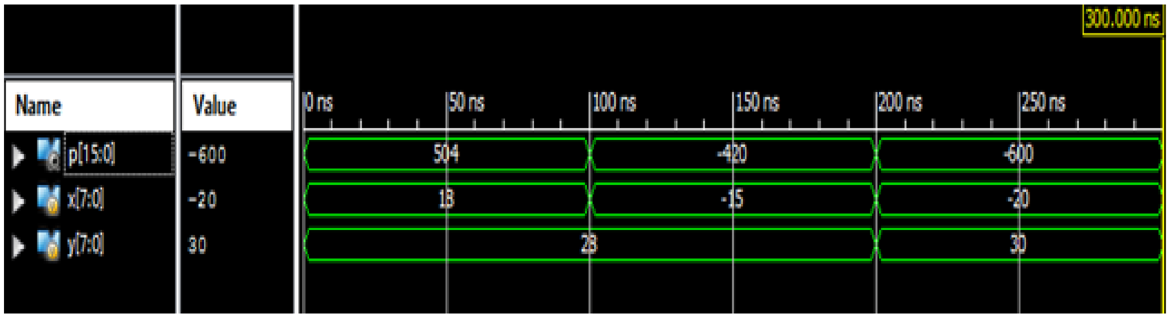
On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	156	9312	2
Signals	0.000	146	---	---
IOs	0.000	32	232	14
Leakage	0.081			
Total	0.081			

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	26.1	82.9	27.1

Supply Summary Source	Voltage	Total Current (A)	Dynamic Current (A)	Quiescent Current (A)
Vccint	1.200	0.026	0.000	0.026
Vccaux	2.500	0.018	0.000	0.018
Vcco25	2.500	0.002	0.000	0.002

	Total	Dynamic	Quiescent
Supply Power (W)	0.081	0.000	0.081

Simulation results



Comparison

Parameters	Existing	Proposed
Area	144 Luts	130 Luts
Power (w)	0.143	0.065
Delay	8.806 ns	5.232ns

CONCLUSION

We proposed a high-speed yet energy efficient approximate multiplier called RoBA multiplier. The proposed multiplier, which had high accuracy, was based on rounding of the inputs in the form of 2n. In this way, the computational intensive part of the multiplication was omitted improving speed and energy consumption at the price of a small error. The proposed approach was applicable to both signed and unsigned multiplications.

FUTURE SCOPE

An Aging-aware variable-latency multiplier design with the AHL. The multiplier is able to adjust the AHL to mitigate performance degradation due to increased delay. The experimental results show that our proposed architecture from 4bit to 16bit multiplication with Booth as last stage instead of Normal RCA adder it will decrease the delay and improve the performance compared with previous designs.

REFERENCES

1. Hao Zhang and Seok-Bum Ko , ―Design of Power Efficient Posit Multiplier,‖ IEEE on Circuits and Systems—ii: express briefs, vol. 67, no. 5, May 2020.Transactions
2. J. L. Gustafson and I. Yonemoto, ―Beating floating point at its own game: Posit arithmetic,‖ Supercomput. Front. Innovat. Int. J., vol. 4, no. 2, pp. 71– 86, Jun. 2017.
3. Sneha Singh and Prachi Singh, ―Implementation of Radix-4 Booth Multiplier by VHDL,‖ IJRREEE, vol 4, issue 1, pp 1-11, Jan 2017.
4. Sukhmeet Kaur, Suman and Manpreet Singh Manna, ―Implementation of Modified Booth Algorithm(radix-4) and its comparison with Booth Algorithm(radix-2),‖ Advances Electric and Electronic Engineering, ISSN 2231-1297, vol

3, no 6,pp 683-690, 2013.

5. Bikash Chandra Sahoo, Sanjay Kumar Samant, ―Design and Power Estimation of Booth Multiplier Using Different Adder Architectures,‖ 2013.
6. S.Shafiulla Basha, Syed, Jahangir Badashah, ―Design and Implementation of Radix-4 Based High Speed Multiplier for ALU‘s using Minimal Partial Products,‖IJAET, ISSN 2231-1963,vol 4, issue 1, pp 314-325.
7. D. R. Kelly, B. J. Phillips, and S. Al-Sarawi, ―Approximate signed binary integer multipliers for arithmetic data value speculation,‖ in Proc. Conf. Design Archit. Signal Image Process., 2009, pp. 97–104.
8. K. Y. Kyaw, W. L. Goh, and K. S. Yeo, ―Low-power high-speed multiplier for error-tolerant application,‖ in Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits (EDSSC), Dec.2010, pp. 1–4.
9. A. Momeni, J. Han, P. Montuschi, and F. Lombardi, ―Design and analysis of approximate compressors for multiplication,‖ IEEE Trans. Comput., vol. 64, no. 4, pp. 984–994, Apr. 2015.
10. K. Bhardwaj and P. S. Mane, ―ACMA: Accuracy-configurable multiplier architecture for error-resilient system-on-chip,‖ in Proc. 8th Int. Workshop Reconfigurable Commun.-Centric Syst.-Chip, 2013, pp. 1–6.
11. K. Bhardwaj, P. S. Mane, and J. Henkel, ―Power- and area-efficient approximate wallace tree multiplier for error-resilient systems,‖ in Proc. 15th Int. Symp. Quality Electron. Design (ISQED), 2014, pp. 263–269.
12. J. N. Mitchell, ―Computer multiplication and division using binary logarithms,‖ IRE Trans. Electron. Comput., vol. EC-11, no. 4, pp. 512–517, Aug. 1962.
13. V. Mahalingam and N. Ranganathan, ―Improving accuracy in Mitchell‘s logarithmic multiplication using operand decomposition,‖ IEEE Trans. Comput., vol. 55, no. 12, pp. 1523– 1535, Dec. 2006.
14. Nangate 45nm Open Cell Library, accessed on 2010. [Online]. Available:http://www.nangate.com/
15. H. R. Myler and A. R. Weeks, The Pocket Handbook of Image Processing Algorithms in C.Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.