A Noval Approch to Design and Implement the High Efficiency Multiplier

Neha Jassi[1] Er. Arti Goel[2]
[1]M. Tech Student [2]Assistant Professor
[1,2]Department of Electronics Communication Engineering
[1,2]Swami Devi Dyal Institute of Engineering And Technology,(Affiliated To Kurukshetra University) Barwala,Panchkula

Abstract— This paper describes the single precision floating point multiplier. Floating-point numbers are widely adopted in many applications due to their dynamic representation capabilities. Floating point numbers represents real numbers in binary format. This paper presents a high speed binary floating point multiplier based on dadda algorithm with brent kung adder. To consume less area addition is done using brent kung adder replacing ripple carry adder. The design achieves frequency of 775.705 MHz with 862 slice area compared to existing floating point multipliers. The floating point multiplier is developed to handle the underflow and overflow cases. The multiplier is implemented using Verilog HDL and it is targeted for Xilinx Virtex-5 FPGA and design is simulated using Modelsim.

Key words: DADDA Multiplier, Brent Kung Adder, Ripple Carry Adder, Single Precision

I. INTRODUCTION

Floating Point (FP) number is widely used in representation of real number which have wide range of values. Multiplication is one of the general arithmetic operations in these computations. Also the need of high speed multiplier is increasing which do arithmetic operations in very less time with greater accuracy. Higher throughput arithmetic operations are important to achieve the desired performance in many real time signal, electronic industry, 3 D technology and image processing applications. One of the key arithmetic operations in such applications is multiplication and the development of fast multiplier circuit. Also reducing the delay and power consumption, maximize the accuracy and speed of multiplication are very essential requirements for many applications.

Floating point numbers represents real numbers in binary format. There are two main types of IEEE 754 floating point number formats. They are single precision and double precision floating point format. This thesis presents a single precision floating point format. Floating point numbers attempt to represent real number with uniform accuracy. It consists of a 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 the significand.

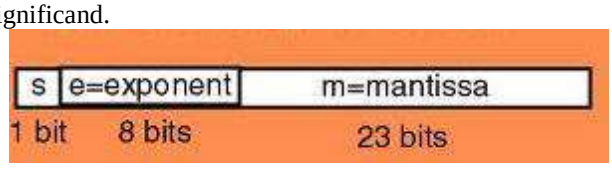


Fig. 1: IEEE Single Precision Floating Point Format

II. FLOATING POINT MULTIPLICATION ALGORITHM

The normalized floating point numbers have the form of Z = (-1S) * 2 (E - Bias) * (1.M). The following algorithm isused to multiply two floating point numbers.

1) Multiplying the significand; i.e. (1.M1*1.M2)
2) Placing the decimal point in the result.
3) Adding the exponents; i.e. (E1 + E2 – Bias)
4) 4. Obtaining the sign of the final result i.e.: (s1 xor s2)
5) Normalizing the result; i.e. obtaining 1 at the MSB of the results significant multiplication result.
6) Rounding the result to fit in the given floating point number format.
7) Checking for underflow or overflow occurrence.

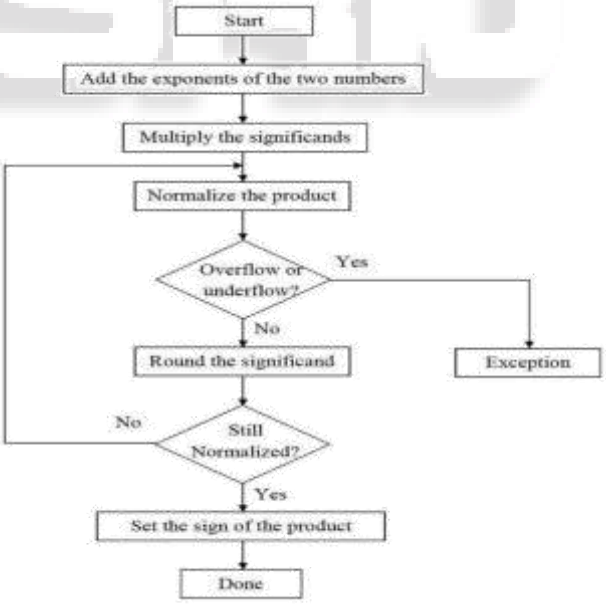


Fig. 2: Floating Point Multiplier Flow Chart

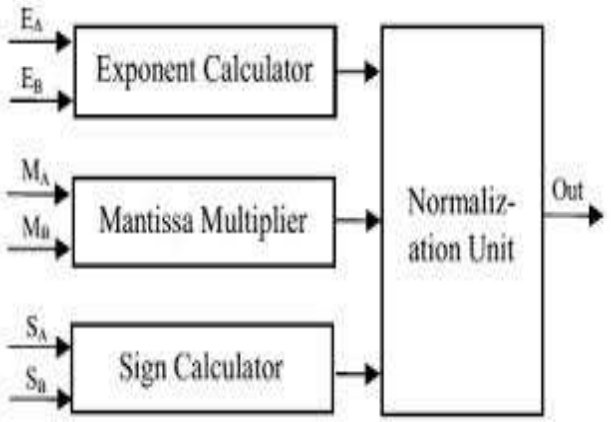


Fig. 3: Floating Point Multiplier Block Diagram

III. EXISTING SYSTEM:

A. Floating Point Multiplier using Dadda Algorithm with Ripple carry adder

Dadda algorithm for multiplication is similar to Wallace algorithm. But it is faster and requires few gates than Wallace algorithm. Dadda projected a sequence of matrix heights that are prearranged to give the minimum number of reduction stages. Due to this, Dadda multiplier has less expensive reduction phase. To reduce the N by N partial product matrix, dada multiplier uses the half adder and full adder. In order to realize the minimum number of reduction stages, the height of each intermediate matrix is limited to the least integer that is no more than 1.5 times the height of its successor. The following procedure is used to reduce partial product matrix in to two rows.

1. Consider dj is the minimum reduced height i.e.: d1=2 and dj+1 = [1.5*dj], where dj is the matrix height for the jth stage. Repeat this calculation until the largest height is calculated. Find the smallest j such that at least one column of the original partial product matrix has more than dj bits.
2. In the jth stage from LSB column, employ full adder and half adder or both adder a reduced matrix with no more than dj bits in any column.
3. Let j = j-1 and repeat step 2 until a height of each column reduced with only two rows.

This method of reduction is called a column compression technique because it attempts to compress each column. Dadda multipliers utilize the least amount number of full adder. So the number of intermediate stages is set in terms of lower bounds: 2, 3, 4, 6, 9 . . . original matrix of Dadda multipliers has N2 bits whereas final two row matrix has the 4.N-3 bits. Each full adder or (3, 2) counter take three inputs and produces two outputs, the number of bits in the matrix is decreased by one with each applied (3, 2) counter therefore, the full number of full adder is N2 – 4.N+3 the length of the carry propagation adder is CPA length = 2.N–2.

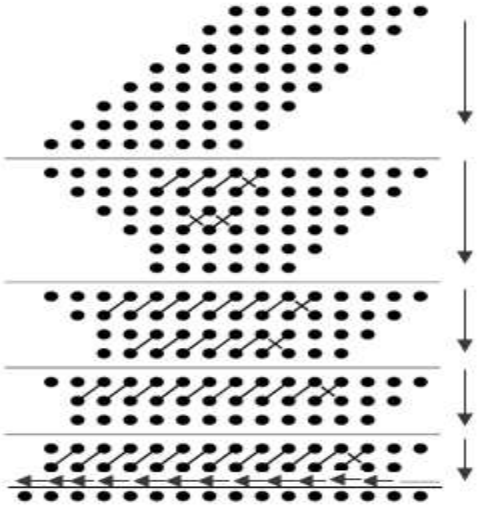


Fig. 4: Dot Diagram For 8 By 8 Dadda Multiplier

IV. PROPOSED SYSTEM

Floating Point Multiplier using Dadda Algorithm with Brent Kung Adder

A. Parallel Prefix Adders.

The construction of parallel prefix adder involves three stages

1) Pre-processing stage
2) Carry generation network
3) Post processing

B. Pre-Possessing Stage:

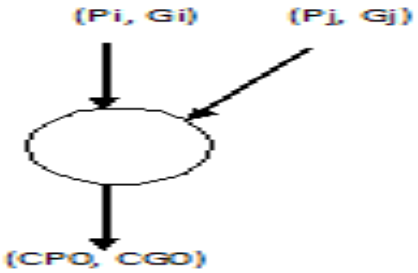
In this stage we compute, generate and propagate signals to each pair of inputs A and B. These signals are given by the logic equations 1&2:

Pi=Ai xor Bi..... (1)
Gi=Ai and Bi..... (2)

C. Carry Generation Network:

In this stage we compute carries corresponding to each bit. Execution of these operations is carried out in parallel. After the computation of carries in parallel they are segmented into smaller pieces. It uses carry propagate and generate as intermediate signals which are given by the logic equations 3&4:

CP i:j=Pi:k+1 and Pk:j.....(3)
CG i:j=Gi:k+1or(Pi:k+1 and G k:j).....(4)



The operations involved in this figure are given as:

CP0=Pi and Pj..... (3(i))
CG0= (Pi and Gj) or Gi..... (3(ii))

D. Post Processing:

This is the final step to compute the summation of input bits. It is common for all adders and the sum bits are computed by logic equation 5&6:

Ci-1=(Pi and Cin) or Gi..... (4)

Si=Pi xor Ci-1..... (5)

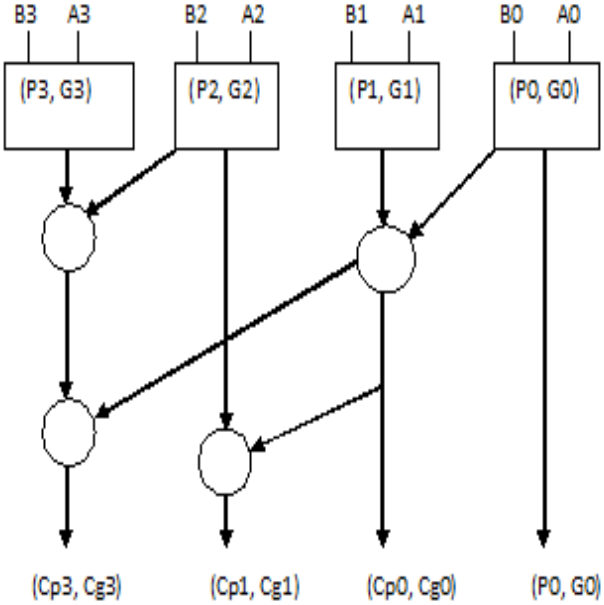


Fig. 4: Bit Brent-Kung Adder.

V. EXPERIMENTAL RESULTS

This section shows the experimental results. We implemented our algorithm in verilog language. Table-2 shows design of floating point multiplier by using dadda algorithm with brent keng adder.

The comparison of utilization of slice, LUT FF pairs and frequency is given in table 5.1. Our design achieve maximum frequency of 775.705 MHz with no. of slice occupied 861 in virtex 6 family. As compared to previous multiplier, the proposed multiplier has better result in terms of speed and slice area.

	PROPOSED FLOATING POINT MULTIPLIER	EXISTING FLOATING POINT MULTIPLIER
LUT & FF Pairs Used	862	1146
CLB Slices	861	1149
Max Frequency (MHz)	775.705	526.857

Table 2: Device Utilization

The modelsim open in a new windows Fig 2:we take two 32 bit random floating point numbers as input and it gives 32 bit output in single clock cycle.

Inputs: a = 32'HC20C0000; = 0 10000001 01100 = 5.5

b = 32'H40B00000; = 1 10000100 00011 = -35

Output: result = 32'HC3408000; = 1 10000110 10000

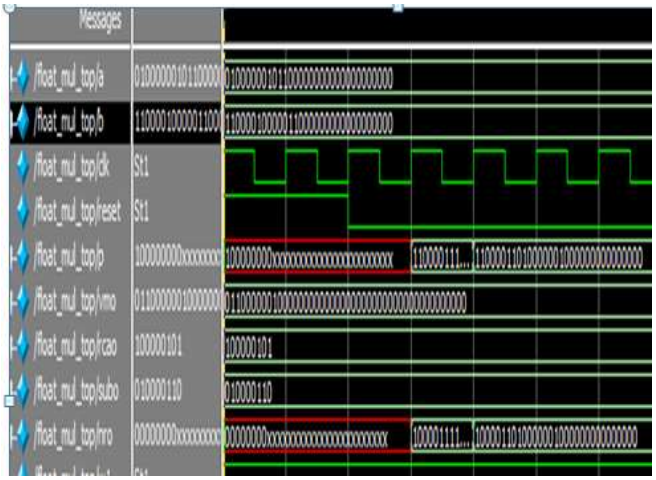


Fig. 5: Simulation Waveform

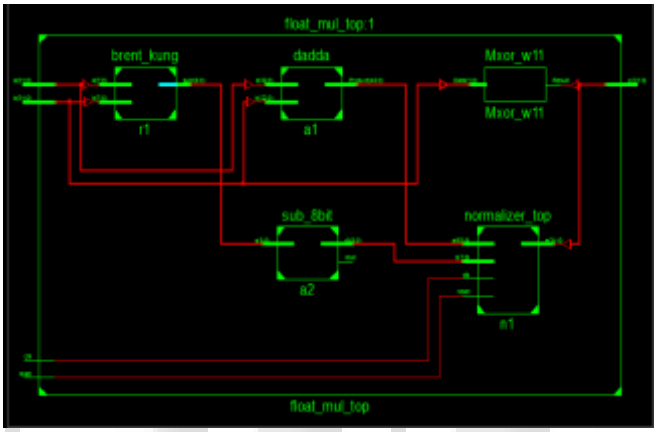


Fig. 4: Internal diagram of floating Point Multiplier

The no. of LUTs and FF pairs and CLB slices decreases by using dadda algorithm with Brent kung adder by replacing ripple carry adder. The no. of LUT and FF pairs used in existing system (1164) is more compared tom proposed system (862). CLB slices used in existing system are 1164 where as in proposed system its amount decreases up to 862. Frequency associated with proposed system is less than the existing multiplier.

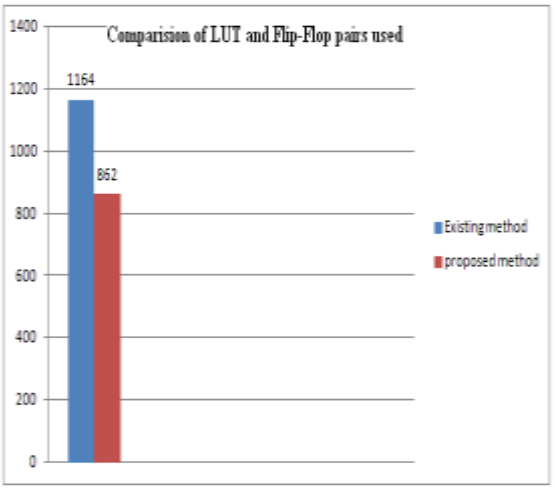


Fig. 5: Comparison of LUT and FF Pairs Used

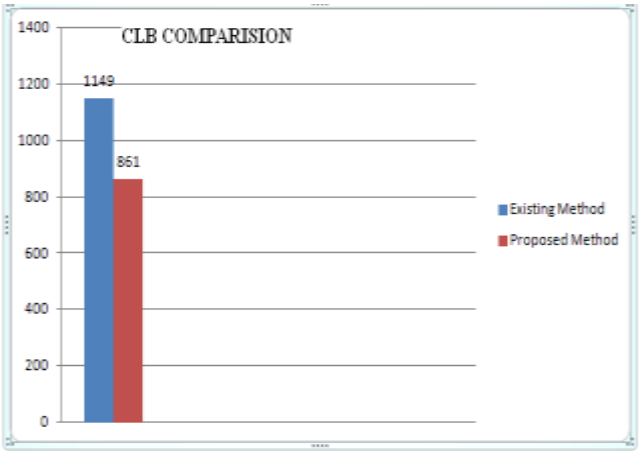


Fig. 6: Comparison of CLB

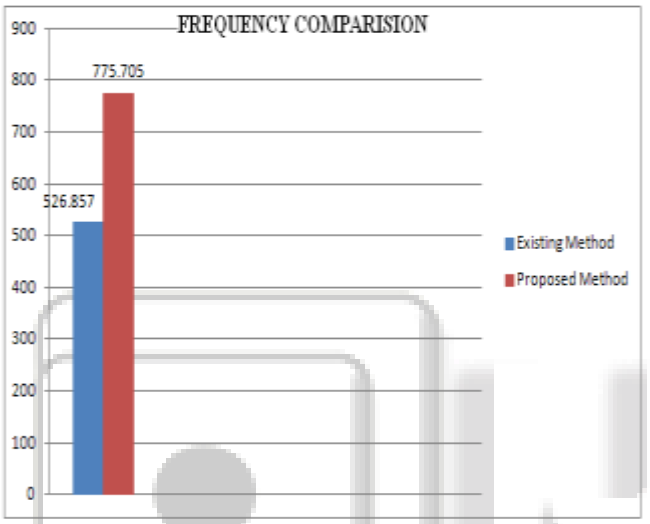


Fig. 7: Comparison of Frequency

VI. CONCLUSION

This paper describes an implementation of a floating point multiplier using dadda multilplier with brent kung adder that supports the IEEE 754 binary interchange format. Proposed method is compared with previous designed Dadda algorithm module. Proposed multiplier achieves the maximum frequency of 775.705 mhz with 861 slice area. thjis method is used to increase the speed of floating point multiplier.

REFERENCES

[1] Vincent, R Anju, S.L. "Decimal floating point format based on commonly used precision for embedded system applications "Emerging Research Areas and 2013 International Conference on Microelectronics, Communications and Renewable Energy (AICERA/ICMiCR), 2013 Annual International Conference on DOI: 10.1109/AICERA-ICMiCR.2013.6575957 Publication Year: 2013 , Page(s): 1- 4

[2] Renteria-Mejia, C.P. Trujillo-Olaya, V. ; Velasco-Medina, J. "design of 8192-bit Montgomery multipliers based on radix-8 Booth encoding and coded-digit " Circuits and Systems (LASCAS), 2013 IEEE Fourth Latin American Symposium pp.-1 – 4, 2013

[3] Prashant kumar Sahu and Asst. Prof Nitin Meena, "Comparative Study Of Different Multiplier Architecture", IJETT, Volume 4 Issue ,pp. 4293,10-Oct 2013

[4] Rakesh Kumar, Neelam Sharma "Design of Fast Pipelined Multiplier using Modified Redundant Adder" I.J. Intelligent Systems published in MECS and Applications, 47-53,2012

[5] Mohamed Al-Ashrfy, Ashraf Salem and Wagdy Anis "An Efficient implementation of Floating Point Multiplier" IEEE Transaction on VLSI 978-1-4577-0069- 9/11@2011 IEEE.

[6] Nachtigal,; Thapliyal, H. ; Ranganathan, N. Design of a reversible single precision floating point multiplier based on operand decomposition Nanotechnology (IEEE-NANO), 2010 10th IEEE Conference on DOI: 10.1109/NANO.2010.5697746 Publication Year: 2010 , Page(s): 233-237

[7] Ron S. Waters, Member, IEEE, and Earl E. Swartz-lander, Jr., Fellow, IEEE," A Reduced Complexity Wallace Multiplier Reduction," IEEE transactions on computers, vol. 59, no. 8, August 2010.

[8] M. Shams et al., "Novel Reversible Multiplier Circuits in Nanotechnology," in World Applied Science Journal, Vol. 3, No. 5, pp, pp. 806- 810, 2008.

[9] W. Shi, Z. Y. Wang, H. G. Ren, T. Cao, W. Chen, B. Su, et.al., "DSS: applying asynchronous techniques to architectures exploiting ILP at compile time," Proc. 28th IEEE International Conference on Computer Design, IEEE Press, 2010, pp. 321-327.

[10] L. B. Huang, L. Shen, K. Dai and Z. Y. Wang, "A new architecture for multiple-precision floating-point multiply-add fused unit design," Proc. 18th IEEE Symposium on Computer Arithmetic, IEEE Press, 2007, pp. 69-76.

[11] J., Langou, et. al., "Exploiting the Performance of 32 bit Floating Point Arithmetic in Obtaining 64 bit Accuracy", University of Tennessee Computer Science Tech Report, UT-CS-06-574, 2006

[12] Huang Z. J., Ercegovac M. D., Cater J., "High-performance Low-Power Left-to-Right Array Multiplier Design", IEEE Transactions on Computers, Vol. 54, No. 3, March, 2005.

[13] 1L.D. Van and C. C. Yang "Generalized low-error area-efficient fixed-width multipliers", IEEE Trans. Circuits Syst. I, vol. 52, no. 8, pp.1608 -1619 2005

[14] Amir K., Kaamran R., Majid A, "A novel multiplier for high speed applications", Proceedings of IEEE International on SOC Conference, 2005, pp.:305 - 308, Sept 25-28, 2005.

