An Efficient Floating Point Arithmetic Unit Using Parallel Prefix Adder

[1] Rinu Susan Babu, [2] Sukanya Sundaresh
[1] PG Student [VLSI and ES] [2] Assistant professor,
Department of ECE, TKM Institute of Technology, Kollam
[1] susanbaburinu@gmail.com, [2] sukanya_cep@yahoo.co.in

***Abstract*— In fixed point number representation, digits after the decimal point is fixed and it does not provide a high precision value, while in floating point representation the decimal point is not fixed. Based on the concept of floating point number a fused floating point arithmetic unit is designed. Generally alignment, normalization and rounding are the complex process required in floating-point operation, which significantly increase the latency. This work relies on a fused floating-point three-term arithmetic unit, which includes a fused floating point three term adder unit, a fused floating point three term subtractor unit and a fused floating point three term multiplier unit. Here addition is the basic operation used in adder unit, subtractor unit and multipliers unit, which results in decrease or increase of delay. In order to improve the performance of a three term floating point arithmetic unit, carry save adder is replaced by a parallel prefix adder like kogge stone adder. A parallel prefix addition mainly includes a pre-processing stage, a carry generation stage and a post processing stage. This floating point three term arithmetic unit using parallel prefix adder is designed using VHDL language and it is synthesised in Xilinx ISE Design Suit 13.2 and can be simulated in Model SimSE 6.3f.**

***Index Terms*— fixed point number, floating point number, fused floating point arithmetic unit and parallel prefix adder**

I. INTRODUCTION

As the arithmetic applications grow, numbers play an important role. A number representation usually specifies some way of storing a number that may be encoded as a string of digits. There are several ways by which strings of digits can represented as a numbers. In common mathematical notation, a digit string can be of any length and the location of the decimal point in the digit string can be indicated by placing a dot. If the decimal point is omitted then the digit of string will become an integer.

Earlier a real number was usually represented by an approximation to some fixed number of places after the decimal or dot point, it is known as fixed-point representation. The main advantage of this kind of representation is that integer arithmetic can be used for storing small values. The main disadvantage of a fixed-point number is that they are not flexibile and it is having a limited integer range and therefore it is difficult to represent large number in the same representation. To avoid the drawbacks of fixed point number, floating point numbers are introduced. A floating point number represents a real number which supports a wide range of values. The term floating point indicates that a decimal point can float. Hence computer realizations of scientific notation use floating-point representation.

II. LITERATURE REVIEW

Over the years, several different number representations have been used in computers system; however, for the last ten years the most commonly encountered representation is defined by the IEEE 754 Standard. IEEE 754 floating point Standard is the most common floating point representation that is used in modern microprocessors. IEEE representation Divides the number of bits into three groups, a sign bit, exponent bits and the mantissa part. A sign bit can be either one or zero, one for negative or zero for positive. The exponent field has 8 bits in single precision and 11 bits in double precision. Apart from the sign bit and the exponent bits a floating-point number also has a magnitude part which is represented by the significand field. Mostly significand bits are also known as mantissa bits. IEEE 754 standarad has two precision format, ie single precision and double precision. For single-precision the number of mantissa bits is 23 and for double-precision it is 52. This standarad supports arithmetic operations like addition, subtraction and multiplication. Floating point arithmetic unit is designed using the concept of IEEE 754 standarad .

Basically the fundamental operations involved in any digital systems are addition. Addition is an indispensible operation in digital or analog system. The performance of adders in digital system determines the speed and accuracy of operation. In many applications of digital systems several types of adders can be used such as half adder, full adders, ripple carry adders, carry look ahead adders etc. Among all these adders carry look ahead adder has an improved delay compared to half adder, full adder and ripple carry adder. The carry look ahead adder improves the speed by reducing the amount of time required to determine the carry bits. To propagate the carry to the next stages it introduced two new signals called Propagate and Generate (G, P).But the problem with the Carry Look Ahead adder is, as the number of input bits increases the delay of an adder becomes worst. To eliminate this problem of delay a new adders called Parallel prefix adders is designed [1]. The main advantage of the parallel prefix adder design is that the carry tree reduces the number of logic levels by essentially generating the carries in parallel. The parallel prefix adders operation performed in three stages. Parallel prefix adders pre-compute the carry to eliminate the carry propagation problems and reduce the delay of the adder. In the first stage carries are generated using two signals of generate and propagate. In the second stage carries will be parallelizable to the next stages, so carry chain delay is reduced in this stage. In the third stage generates the sum using the previous stage carries and propagates bits. Sum is generated by the Exclusive-OR operation of the previous carries and propagates bits.

The floating point arithmetic unit uses addition as the basic fundamental operation in adder, subtraction and multiplication unit. This floating point arithmetic unit is designed using a carry save adder. The addition operation using this carry save adder results in decrease or increase of the delay. In order to improve the performance of floating point arithmetic unit, here addition operation is done by replacing carry save adder by a parallel prefix adder like kogge stone adder .

III. PROPOSED SYSTEM

In many DSP applications, multiple floating-point additions are executed consecutively. The floating-point multi-term adder takes multiple input and executes multiple additions with an operation to generate a sum. Generally floating-point multi-term adder can be represented by the floating-point three-term adder designs. There are two approaches to design the floating-point three-term adder [2]. used ie, discrete floating-point three- term and fused floating-point three-term. A direct way to design the floating-point three-term adder is to execute two floating-point additions which is referred to as a discrete floating-point three-term adder. In the case of discrete floating point three term adder, the first floating-point adder takes two inputs and computes an intermediate sum. Then, the second floating-point adder takes the intermediate sum and the third input and computes the final sum. Where as in the case of fused floating-point multi-term adder it takes multiple input and generate a sum.

The main idea of a fused floating point arithmetic unit is to design a 32 bit floating point unit , which includes a three term adder unit, a three term subtractor unit and a three term multiplier unit. Basic block diagram for a fused floating point arithmetic unit is shown in fig 1. An arithmetic unit design aims to minimize time complexity for achieving high speed.

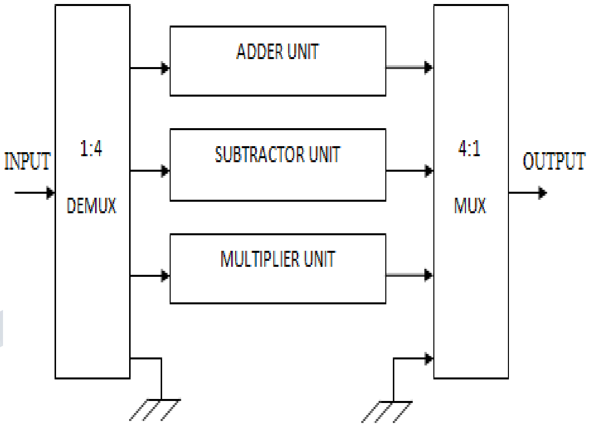


Fig. 1. Block diagram of floating point arithmetic unit

The basic operation involved in a fused floating point arithmetic unit are sign compare, exponent compare, significand alignment, mantissa addition, leading zero detector , normalisation and rounding. Here addition operation can carried out by a 3:2 carry save adder, which takes three input and produces two output , as a result delay increases.This delay introduced by carry save adder can be avoided by, replacing the addition operation using a parallel prefix adder like kogge stone adder [1]. Kogge stone adder is considered as the fastest adder and hence it is widely used in the industry for high performance arithmetic circuits. In kogge stone adder, carry are generated in parallel at the cost of increased area.The complete functioning of Kogge stone adder can be done in three distinct parts:

A. Pre processing stage:

It involves the computation of generate and propagate signals corresponding to each pair of bits in A and B. The logic equations for generate and propagate signals are given below:

$$pi = Ai \text{ xor } Bi \quad (1)$$

$$gi = Ai \text{ and } Bi \quad (2)$$

B. Carry generation stage:

Carry generation involves the computation of carry corresponding to each bit. It uses group propagate and generate as intermediate signals. The logic equations are shown below:

Pi:j = Pi:k+1 and Pk:j (3)

Gi:j = Gi:k+1 or (Pi:k+1 and Gk:j) (4)

C. Post processing stage:

It is the final step involed in Kogge stone adder is the post processing stage. It is common to all adders in carry look ahead family. It involves computation of sum bits. Sum bits are computed by the logic given below:

Si = pi xor Ci-1 (5)

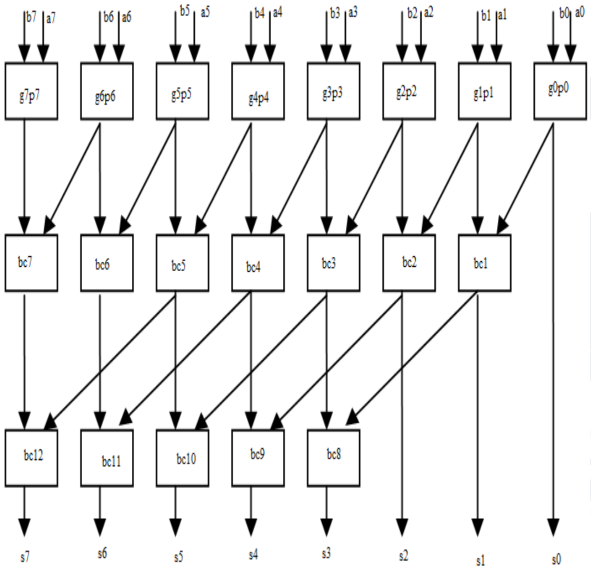


Fig. 2. 8-bit Kogge-Stone Adder

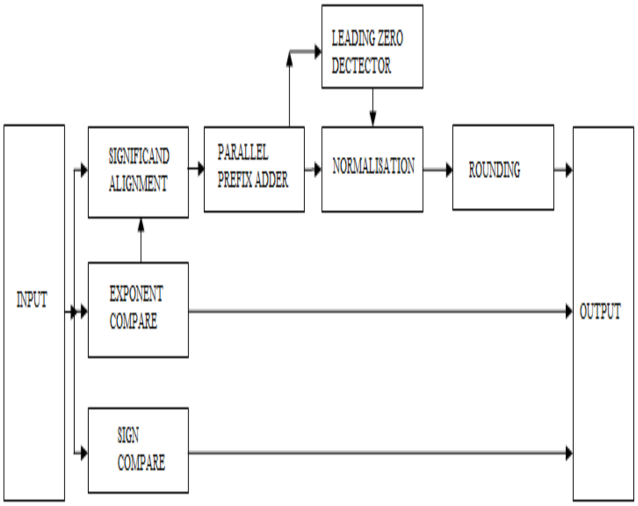


Fig. 3. Block diagram for fused floating point adder

Based on the concept of kogge stone adder three 24bit mantissa addition for floating point adder and floating point subtractor is done, for floating point multiplier an 8bit exponent addition is done using kogge stone adder. Fig 2 shows an 8-bit Kogge-Stone Adder.

D. Fused floating point adder

Floating point addition is the most commonly used floating point operation. The block of a fused floating point adder unit is shown in fig 3, which includes a sign compare, exponent compare, significand alignment, a parallel prefix adder , leading zero anticipation , normalisation and rounding [2].The procedure of executing fused floating-point three-term adder is :

i) The exponent compare logic determines the max exponent among the three exponents and computes the differences between the max exponent and each exponent. Then the corresponding three significands bits are shifted by the amount of the corresponding exponent differences.

ii) The effective operations are determined based on the three sign bit. Then, the significands are passed to a kogge stone adder.

iii) The significand addition is performed. The leading zero detection is performed in parallel with the significand addition operation and the significand sum is shifted by the amount of the leading zero detection result.

iv) The sign logic determines the sign of the sum result.

v) The normalized significands are rounded.

E. Fused floating point subtractor

Addition and subtraction are the basic operations in computer arithmetic. Fast adders or subtractors are desirable not only for speeding up fundamental operations like addition and subtraction in arithmetic operation, but also for accelerating the multiplication and division which involve massive addition and subtraction.Floating point subtractor is similar to that of floating point addition. The block diagram for a fused floating point subtractor is shown in fig 4, which includes a sign compare, exponent compare, significand alignment, invert, a parallel prefix adder adder, leading zero anticipation, normalisation and rounding [2].

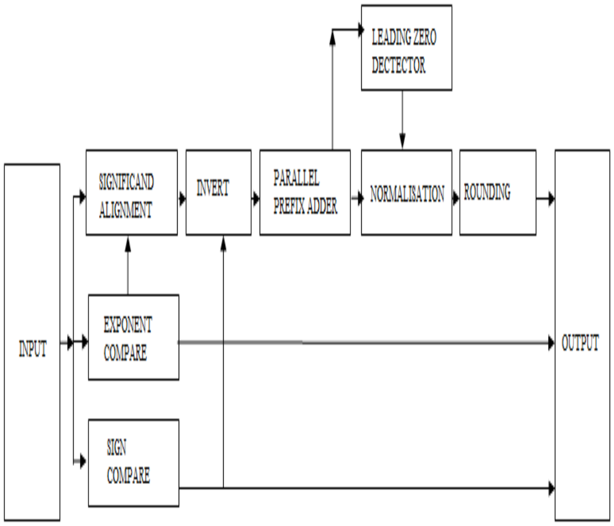


Fig. 4. Block diagram of fused floating point subtractor

The procedure of executing fused floating-point three-term subtractor is :
i) The exponent compare logic determines the largest exponent among the three exponents and computes the differences between the max exponent and each exponent. The three significands are shifted by the amount of the corresponding exponent differences.
ii) The effective operations are determined based on the three sign bits. Then, the significands are passed to a kogge stone adder.
iii) If the operation is negative the the corresponding significand are inverted by taking two's complement operation. Then the significand addition is performed by taking twos complement. The leading zero detection is performed in parallel with the significand addition and the significand sum is shifted by the amount of the leading zero detection result.
iv) The sign logic determines the sign of the sum result.
v) Finally the normalized significands are rounded.

F. *Fused floating point multiplier*

With the constant growth of computer applications ,fastest arithmetic operation especially multipliers are increasingly required. Multipliers are used in digital signal processing operations, such as correlations, convolution, filtering and frequency analysis.

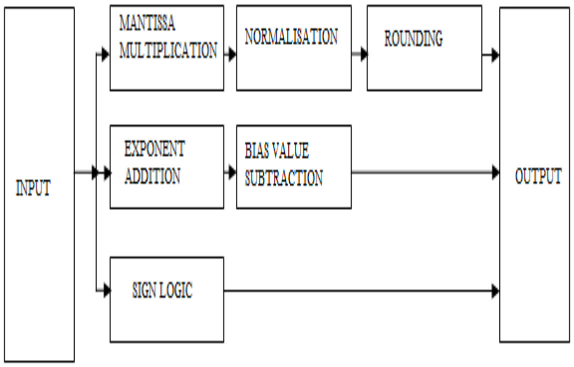


Fig. 5. Block diagram of fused floating point multiplier

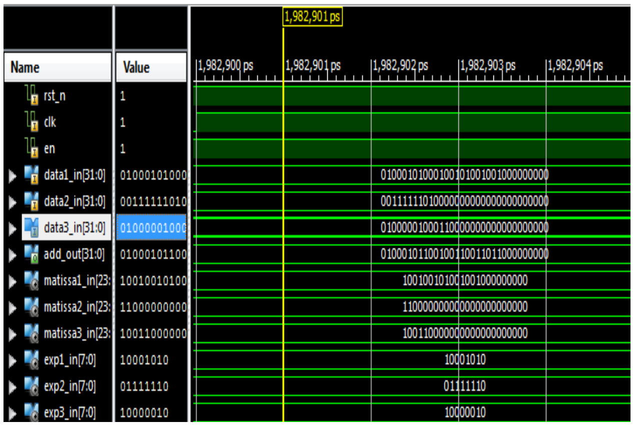
A fused floating point three term multiplier includes a sign logic, exponent addition using a parallel prefix adder ,bias value subtraction, signficand multiplication , normalisation and rounding [4]. Fig 5 shows the basic block diagram of fused floating point multiplier unit.

The procedure of executing fused floating-point three term multiplier is given below:

i) Compute the sign of the result by taking Xs xor Ys xor Zs.
ii) Exponent are calculated by adding the biased exponents of the three numbers, them subtracting with the bias value. The bias value is 127 for single precision and and 1023 double precision based on the IEEE data format respectively es1 + es2 + es3 – bias.
iii) Multiply the three significands as: Xm * Ym * Zm.
iv) Normalize the product if MSB of the product is 1, by shifting the product.
v) Result are finally rounded to the allowed number of mantissa bits.

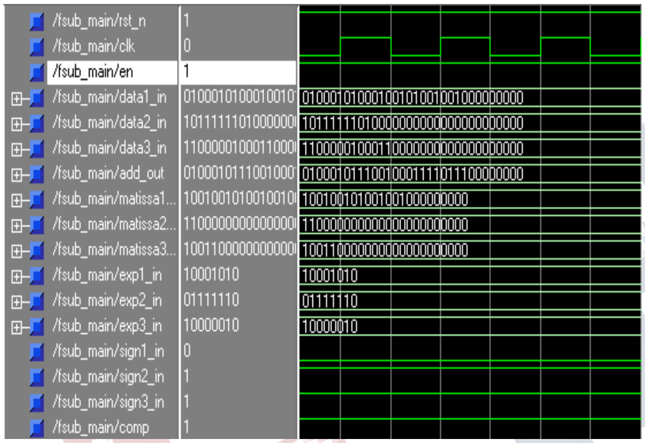
IV. EXPERIMENTAL RESULTS

The modules are designed using VHDL language in Xilinx ISE Design Suite 13.2 and the simulation of the design is performed using ModelSim SE 6.2f . Here a fused floating point arithmetic unit is designed which includes a fused floating point three term adder is shown in fig 6, a fused floating point three term subtractor is shown in fig 7,and a fused floating point three term multiplier is shown in fig 8.



Name	Value	
rst_n	1	
clk	1	
en	1	
data1_in[31:0]	01000101000	01000101000100101001001000000000
data2_in[31:0]	00111111010	00111111010000000000000000000000
data3_in[31:0]	01000001000	01000001000110000000000000000000
add_out[31:0]	01000101100	01000101100100110011011000000000
matissa1_in[23:	10010010100	100100101001001000000000
matissa2_in[23:	11000000000	110000000000000000000000
matissa3_in[23:	10011000000	100110000000000000000000
exp1_in[7:0]	10001010	10001010
exp2_in[7:0]	01111110	01111110
exp3_in[7:0]	10000010	10000010

Fig. 6. Simulation result of floating point three term adder

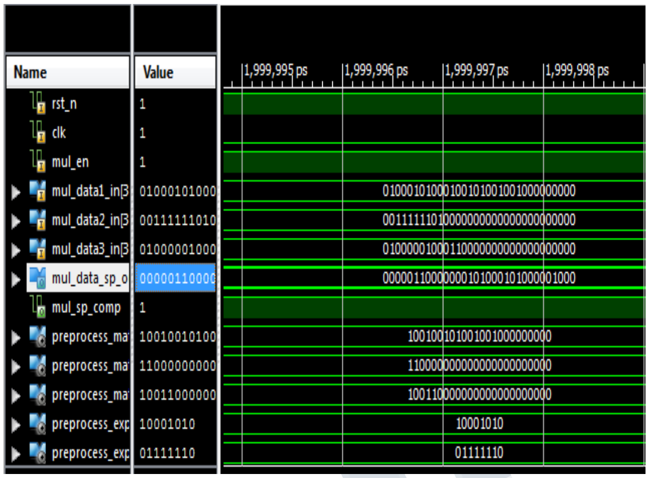


/fsub_main/rst_n	1	
/fsub_main/clk	0	
/fsub_main/en	1	
/fsub_main/data1_in	0100010100010010	01000101000100101001001000000000
/fsub_main/data2_in	1011111101000000	10111111010000000000000000000000
/fsub_main/data3_in	1100000100011000	11000001000110000000000000000000
/fsub_main/add_out	0100010111001000	01000101110010001110111000000000
/fsub_main/matissa1...	1001001010010010	100100101001001000000000
/fsub_main/matissa2...	1100000000000000	110000000000000000000000
/fsub_main/matissa3...	1001100000000000	100110000000000000000000
/fsub_main/exp1_in	10001010	10001010
/fsub_main/exp2_in	01111110	01111110
/fsub_main/exp3_in	10000010	10000010
/fsub_main/sign1_in	0	
/fsub_main/sign2_in	1	
/fsub_main/sign3_in	1	
/fsub_main/comp	1	

Fig. 7. Simulation result of floating point three term subtractor

TABLE I
Comparison of fused floating point arithmetic unit using carry save and kogge stone adder.

DELAY	USING CARRY SAVE ADDER	USING KOGGE STONE ADDER
FLOATING POINT ADDER	24.231ns	23.171ns
FLOATING POINT SUBTRACTOR	23.575ns	21.606ns
FLOATING POINT MULTIPLIER	74.649ns	73.463ns



Name	Value	
rst_n	1	
clk	1	
mul_en	1	
mul_data1_in[3	01000101000	01000101000100101001001000000000
mul_data2_in[3	00111111010	00111111010000000000000000000000
mul_data3_in[3	01000001000	01000001000110000000000000000000
mul_data_sp_o	00000110000	00000110000000101000101000001000
mul_sp_comp	1	
preprocess_ma	10010010100	100100101001001000000000
preprocess_ma	11000000000	110000000000000000000000
preprocess_ma	10011000000	100110000000000000000000
preprocess_exp	10001010	10001010
preprocess_exp	01111110	01111110

Fig. 8. Simulation result of floating point three term multiplier

While comparing a fused floating point arithmetic unit using carry save adder and fused floating point using kogge stone adder, we can see that fused floating point arithmetic unit using kogge stone adder is more efficient in terms of its delay than that of fused floating point arithmetic unit using carry save adder which is shown in table I.

V. CONCLUSION

Based on floating point number a single precision fused floating-point three term arithmetic unit in the IEEE 754 standard format has been designed . A floating point three term arithmetic unit takes three input each of 32 bit. A fused floating point three term arithmetic unit includes an adder unit, subtractor unit and multiplier unit. Addition operation done using carry save adder increases the delay. Inorder to avoid this delay using carry save adder, addition operation is done by a kogge stone adder which further reduces the delay. The whole architecture is designed using VHDL language and synthesized in Xilinx ISE Design Suite 13.2 and simulated in ModelSim SE 6.2f.

REFERENCES

[1] Sunil M, Ankith R D, Manjunatha G D and Premananda B S , “Design and implementation of faster parallel prefix Kogge Stone adder,” International Journal of Electrical and Electronic, Engineering & Telecommunications, Vol. 3, January 2014.

[2] Earl E. Swartzlander , Jongwook Sohn,“ A Fused Floating Point Three Term Adder,” IEEE Transactions on circuits and systems , vol. 61,no.10, October 2014.

[3] Gayathiri K, Prakash S P, Valarmathy S and Jaibalaji K R,“ Implementation Of Floating point Adder Unit,” International Journal of Advanced Research in Electronics and Communication Engineering , vol. 3 , November 2014.

[4] B. Sreenivasa Ganesh, J. E. N. Abhilash and G. Rajesh Kumar “ Design and Implementation of Floating Point Multiplier for Better Timing Performance,” International Journal of Advanced Research in Computer Engineering and Technology , vol. 1, September 2012.

[5] Y.Tao ,G.Deyuan ,F.Xiaoya and R.Xianglong,“Three-operand floating point adder,” in Pro 12th IEEE International Conference on Computer and Information Technology , 2012.

[6] Dhiraj Sangwan and Mahesh K. Yadav “Design and Implementation of Adder / Subtractor and Multiplication Units for Floating Point Arithmetic,” International Journal of Electronics Engineering ,Vol.2, 2010.

[7] A. Tenca,“ Multi operand floating point addition,” in Proc.21st Symp. Computer Arithmetic, 2009.