International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 02 | Feb 2020 www.irjet.net p-ISSN: 2395-0072

DESIGN AND IMPLEMENTATION OF DOUBLE PRECISION FPU FOR OPTIMISED SPEED

R.Bhuvanapriya[1], Dr.Menakadevi.T[2]

[1] P.G. Student, Dept. of ECE, Adhiyamaan College of Engineering, Hosur, Tamil Nadu, India
[2]Associate Professor, Dept. of ECE, Adhiyamaan College of Engineering, Hosur, Tamil Nadu, India

---***---

Abstract - *Currently, each CPU has one or additional Floating Point Units (FPUs) integrated inside it. It is usually utilized in math wide-ranging applications, such as digital signal processing. It is found in places be established in engineering, medical and military fields in adding along to in different fields requiring audio, image or video handling. A high-speed and energy-efficient floating point unit is naturally needed in the electronics diligence as an arithmetic unit in microprocessors. The most operations accounting 95% of conformist FPU are multiplication and addition. Many applications need the speedy execution of arithmetic operations. In the existing system, the FPM(Floating Point Multiplication) and FPA(Floating Point Addition) have more delay and fewer speed and fewer throughput. The demand for high speed and throughput intended to design the multiplier and adder blocks within the FPM (Floating point multiplication)and FPA(Floating Point Addition) in a format of double-precision floating point operation is internally pipelined to achieve high throughput and these are supported by the IEEE 754 standard floating point representations. This is designed with the Verilog code using Xilinx ISE 14.5 software tool is employed to code and verify the ensuing waveforms of the designed code.*

Keywords: **FPU, FPM, FPA, IEEE 754,Xilinx ISE.**

1. INTRODUCTION

Floating point units (FPUs) are a mathematical coprocessor that's just for floating point numbers. It will handle arithmetic operations. The most newest processors through software library routines can also accomplish numerous transcendental functions like exponential or trigonometric calculation. In computer the real numbers can be expressed in numerous ways. Floating-point representation mainly follows the standard IEEE-754 format, which is the foremost common way of representing an approximate to real numbers in computers as it's well handled in most bulky processors. The real numbers represented in binary format are called floating point numbers. These are portrayed computers by a standards Committee which was formed in 1985 by IEEE .The IEEE-754 floating point illustration for binary real numbers contains three parts is shown in figure1.

Fig.1: Floating point representation

SIGN	EXPONENT	MANTISSA

1.1. Floating Point Representation

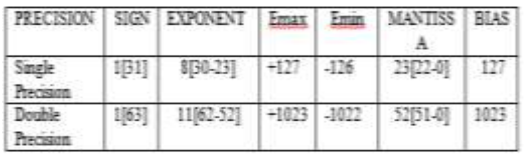
The floating point are often represented in four main sizes such as half (16 bits) ,single(32 bits),double(64 bits) and quadruple(128 bits) precision. The IEEE single precision floating point format has a sign bit, 8 bits of exponent, 24 bits of mantissa and occupies 32 bit. The value of bias is 127, an exponent 0 implies that -127 is kept within the exponent field. The exponents -127 (all 0s) and +128 (all 1s) .The mantissa/significand is 24 bits long -23 bits of the mantissa that is stored in the memory and an implied '1' as the most significant'24th'bit.Thus, a (32 bit) single precision floating point number representation in the IEEE standard is given by equations(1)&(2).

$$X=(-1^{S})\ \square 2^{(E\text{-}Bias)}\ \square (1.M) \qquad (1)$$

$$\text{Value} = (-1^{\text{Sign bit}})\ \square 2^{(\text{Exponent-127})}\ \square (1.\text{Mantissa}) \qquad (2)$$

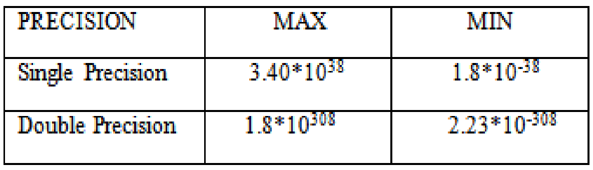
The detailed range of the bit ranges of single precision and double precision is represented in Table I. The IEEE double precision floating point format consist of a sign bit,11 bits of exponent ,53 bits of mantissa and occupies 64 bits on the whole. The exponent uses a biased representation with a bias of 1023. Thus, 64 bit double precision floating point number representation in the IEEE standard is given by equations (3).

$$\text{Value} = (-1^{\text{Sign bit}})\ \square 2^{(\text{Exponent-1023})}\ \square (1.\text{Ma ntissa}) \quad (3)$$



PRECISION	SIGN	EXPONENT	Emax	Emin	MANTISSA	BIAS
Single Precision	1[31]	8[30-23]	+127	-126	23[22-0]	127
Double Precision	1[63]	11[62-52]	+1023	-1022	52[51-0]	1023

Table I: Bit Range



PRECISION	MAX	MIN
Single Precision	$3.40*10^{38}$	$1.8*10^{-38}$
Double Precision	$1.8*10^{308}$	$2.23*10^{-308}$

Table II :IEEE 754 Standard Ranges

The IEEE 754 Standard range of maximum and minimum values that the single (32 bit) and double (64 bit) precision that can accommodate is shown in Table II. If the values exceeds the maximum value then its overflow case and when the values is below the minimum range then its underflow case .

In FPM and FPA internal adders and multipliers constitute an essential element as it performs 95% of operations ,so with extensive research is focused on improving its performance and delays. Basically arithmetic operations like addition and multiplications to be implemented in digital computers. Amongst every arithmetic operation once the implementation of addition completed then it is easy to achieve multiplication.

This paper is ordered as: Section 2 describes work related to implementation ,the system architecture is given in Section 3, section 4 presents double precision floating point, section 5 presents pipelining technique ,section 6 implementation of pipelined 64 bit FPM and section 7 presents the implementation of pipelined 64 bit FPA and section 8 presents simulation results and section 9 presents conclusion and section 10 presents the future scope .

2. RELATED WORK

Several researchers have worked on implementing floating point arithmetic in their own way. The idea of floating point numbers given and three fields includes sign, fraction and an exponent field and extremely small to huge numbers with a varying level of precision [1]. The IEEE-754 Standard defining numerous floating point number format and the sizes of the fields [2].The well known CLA is used in implementation in this paper [3].The concept of double precision floating point addition and subtraction for arithmetic unit was designed using the IEEE 754 format [4].The concept of pipelining is referred [5].This paper states with comparison of other adders CLA is better [8].The anatomy of floating point number concept referred [9]. The concept of representation of floating point number for single and double precision is stated.[10].

3. ARCHITECTURE OF THE SYSTEM

The floating point unit is an IC which is intended to govern all the arithmetic operations based on floating point numbers or fractions. Figure 2, shows the architecture of proposed FPU, this unit shows that there are pre-normaliser block for add and multiplication units and post normaliser block for both add and multiplication and the later exceptions unit is present to handle the expectations unit The fpu_op helps in choosing the operation based on the inputs the operation is performed and the op_a and op_b are the operands and these are the floating point numbers will perform the appropriate functions based on the fpu_op operation . The fundamental microprocessor isn't capable of manipulating floating number quickly so a separate specialized floating point unit is designed as co-processor. This unit is meant to perform basically five operations like addition, subtraction, multiplication, division and square root. The proposed FPU corresponds to two major units in the unit which is represented in the top view of FPU is shown in Figure 3.

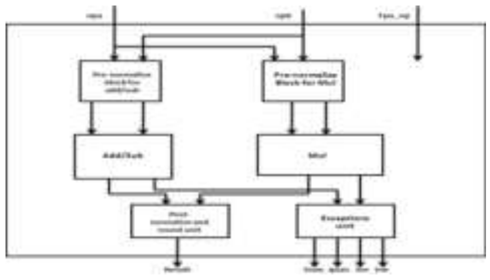


Fig 2: Architecture of proposed FPU

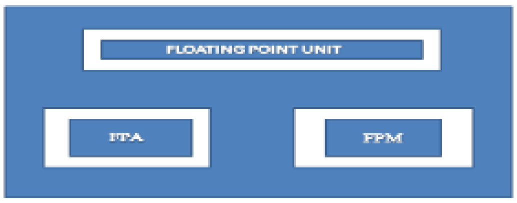


Fig 3: Top view of proposed FPU

4. DOUBLE PRECISION FLOATING POINT

The double precision floating point is ' 1' or '0' in sign bit. "1" in sign bit indicates the negative and "0" indicates the positive. The total number of exponent is 255 here the ranges depends on the exponent value in accordance 11 bits of exponent is accommodated and it has a bias of 1027 .There are totally 53 bits of mantissa but last 1 bit is hidden so it's always represented as 52 bit of mantissa as per IEEE-754 standard and occupies 64 bit.

5. PIPELINING TECHNIQUE

Pipelining is one of the mainstream strategies to acknowledge high performance computing stage. It was first introduced by IBM in 1954 in Project Stretch. In a pipelined unit a new task is started before the previous is finished. This is feasible because most of the adders, high speed multipliers are combinational circuits. Formerly a output has been set it remains fixed for the remainder of the operation. Pipelining is a technique to provide additional increase in bandwidth by allowing simultaneous execution of many tasks. To achieve pipelining input processes must be subdivided into a sequence of subtasks, each of which can be executed by dedicated hardware stage that operates along with other stages in the pipeline. The adder design pipelines the steps shifting, addition and normalization to achieve a summing up every clock cycle. Each pipeline stage perform operations autonomous of others. Input data to the added endlessly streams in. Speed of operation of pipelined add has been established 3 times more than the non pipelined addition and multiplication.

6. IMPLEMENTATION OF FLOATING POINT MULTIPLICATION WITH PIPELINING

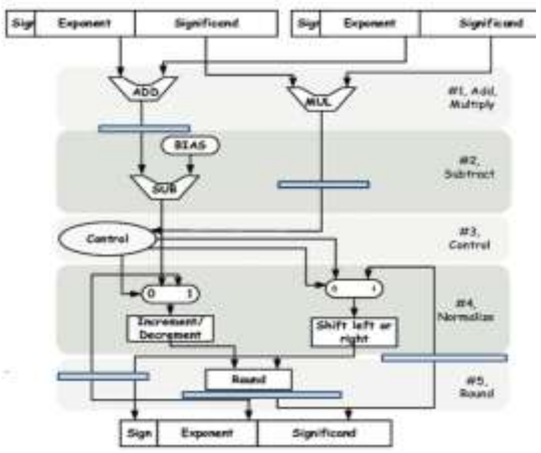
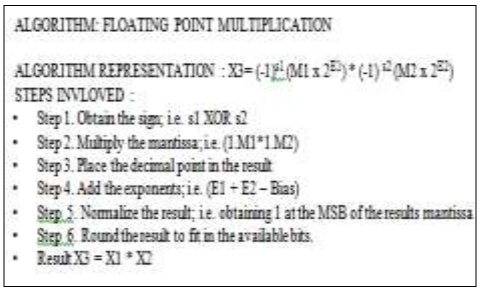


Fig 4 : Pipelined Floating Point Multiplication Architecture



ALGORITHM: FLOATING POINT MULTIPLICATION

ALGORITHM REPRESENTATION : X3= $(-1)^{s1}(M1 \times 2^{E1}) * (-1)^{s2}(M2 \times 2^{E2})$

STEPS INVLOVED :

- Step 1. Obtain the sign; i.e. s1 XOR s2
- Step 2. Multiply the mantissa; i.e. (1.M1*1.M2)
- Step 3. Place the decimal point in the result
- Step 4. Add the exponents; i.e. (E1 + E2 – Bias)
- Step 5. Normalize the result; i.e. obtaining 1 at the MSB of the results mantissa
- Step 6. Round the result to fit in the available bits.
- Result X3 = X1 * X2

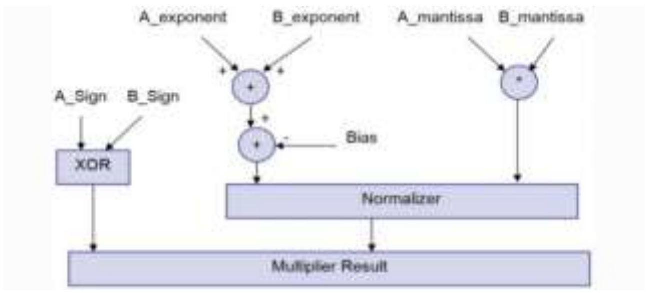


Fig 5: Block Diagram of Multiplication unit

The proposed pipelined floating point multiplication architecture is shown figure 4.The pipelined register are placed in the task to be performed. The block diagram of multiplication unit is shown in figure 5.Consider two operands A and B , the XOR involves in SIGN CALCULATION : if the number is positive then its "0" else "1.

EXPONENT CALCULATION : In this design exponent addition is calculated with two 11 bit exponent and the block is proposed with 11 bit carry look ahead adder (CLA) as shown in figure 6 ,it enhance speed by dropping the time requisite to validate carry bits. It calculates one or more carry bits prior to the sum, this reduces the stay time to conclude the result of the larger-value bits of the adder. The consequence is a reduced carry propagation time. It consist of propagate, sum and carry generator. It adds the exponents of the two operands and then it bias (1023) is subtracted from results obtained from the resultant result of exponent i.e.EA+EB-bias .

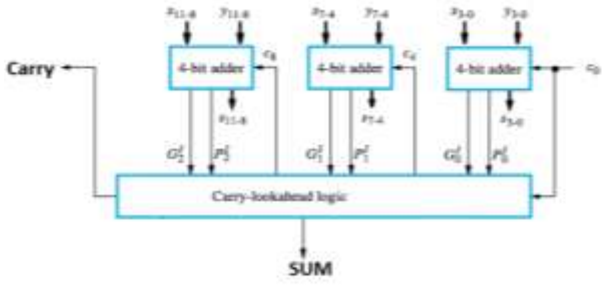


Fig 6 : 11 Bit Carry Look Ahead For Exponent

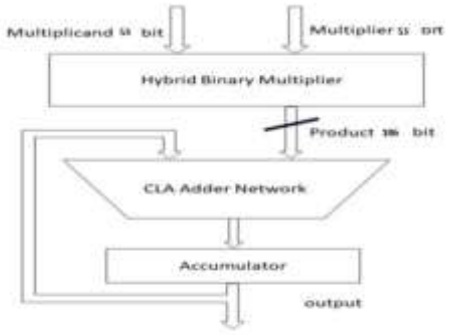


Fig 7: 53 Bit Hybrid Multiplier For Mantissa

MANTISSA CALCULATION: In this design operands A and B mantissa/significand are to be multiplied. As performance of mantissa multiplier dominates overall performance of the floating point multiplication, proposed 53 bit hybrid multiplier is designed as shown in figure7.In,53 bit operands A and B is feed into hybrid multiplier the partial products are given to CLA adder network then those values gets stored in the accumulator then final result of the calculated values is results as the output of 106 bits .Finally, all the resultants of the values of the sign calculation and mantissa calculation and exponent calculation are all formed into a standard IEEE-754 format representation.

7. IMPLEMENTATION OF FLOATING POINT ADDITION WITH PIPELINING

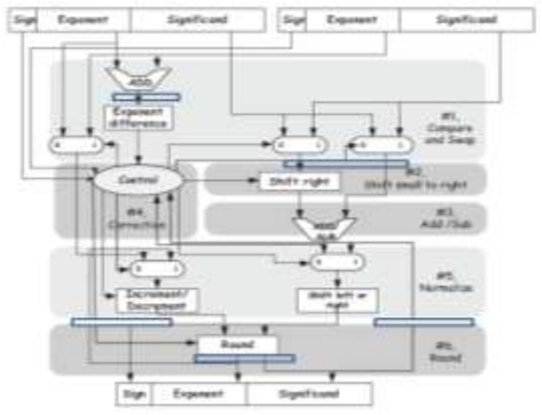


Fig 8 : Pipelined Floating Point Addition Architecture

ALGORITHM: FLOATING POINT ADDITION

ALGORITHM REPRESENTATION: X3= (M1 x 2^{E1})+/- (M2 x 2^{E2})

STEPS INVLOVED :

- Step 1: Check for the absolute value (Abs (X1) > Abs (X2))
- Step 2: Calculate exponent difference Exp_diff= (E1-E2)
- Step 3 : Shift decimal point
- Step 4: Obtain the sign; i.e. s1 XOR s2
- Step 5: Add the mantissa 1.M3=1.M1 + M2 (aligned M2)
- Step 6: Normalize the result; i.e. obtaining 1 at the MSB of the results mantissa.
- Step 7: Round the result to fit in the available bits.
- Result : X3= Added resultant of eR , sR , mR

The proposed pipelined floating point addition architecture is shown figure 8 .The pipelined register are placed in the task to be performed. The block diagram of multiplier is shown in figure 9.Consider two operands A and B , the XOR involves in SIGN CALCULATION : if the number is positive then its "0" else "1.

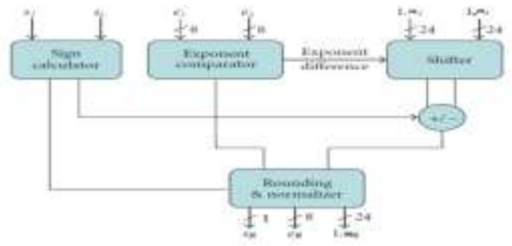


Fig 9 : Block Diagram Of Addition Unit

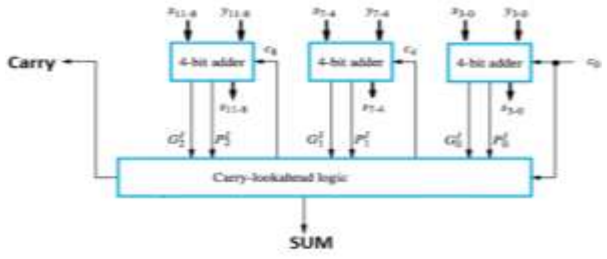


Fig 10 : 11 Bit Carry Look Ahead For Exponent

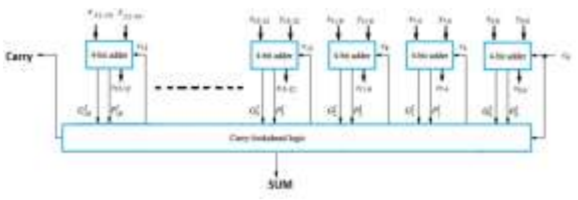


Fig 11: 53 Bit Carry Look Ahead For Mantissa

EXPONENT CALCULATION: In this design exponent addition is completed with two 11 bit exponent and the block is proposed with 11 bit carry look ahead adder (CLA) as shown in figure 10 , it helps in reducing the sum of time necessary to conclude carry bits. To enumerate one or supplementary carry bits forward of the sum it , the result of the huge-value bits of the adder conclude that its less wait of time. It adds the resultant exponents A and B then it subtracted from bias 127 i.e.EA+EB-bias.

MANTISSA CALCULATION: In this design is mantissa / significand of two operands are added. As performance of mantissa adder dominates overall performance of the floating point addition, proposed 53 bit CLA is design is shown in figure 11.The 53 bit operands A and B is feed into CLA then final result of the calculated values is results as the output of 106 bits .Finally, all the resultants of the values of the sign calculation and mantissa calculation and exponent calculation is followed by a standard IEEE-754 format representation.

8. SIMULATION RESULTS

The simulation of double precision FPM and FPA of internal adder and multiplier has been done to calculate the high throughput. This section includes comparative results of hybrid multiplier and existing multipliers and also existing adders and proposed adder. The input values are converted by a tool [6][7].Table III shows the comparison of multipliers. Table IV shows the comparison of adders and Table V shows device summary of proposed double precision FPM.Table VI shows performance analysis of proposed double precision FPM. Table VII shows device summary of proposed double precision FPA.Table VIII shows performance analysis of proposed double precision FPA is implemented by using Verilog language and its simulated in Xilinx ISE 14.5i on Spartan 6.

1.DOUBLE PRECISION OF FPM

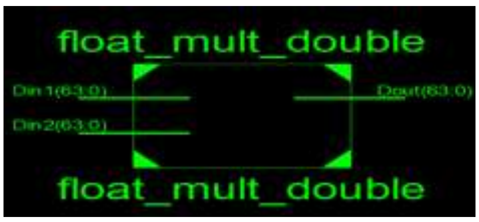


Fig 12: RTL design of double precision FPM

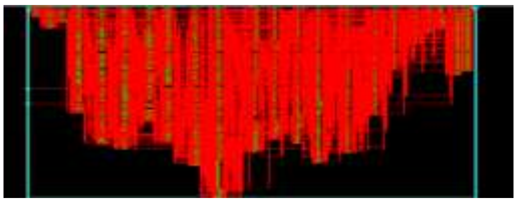


Fig 13: Technology schematic of double precision FPM

The proposed double precision FPM is designed. The proposed FPM is integrated internally with the proposed adder and hybrid multiplier and RTL design and technology schematic view is shown in figure 12,13.The simulation results of the calculated values is shown in figure 14.

Fig 14: Simulation results of double precision FPM

Double Precision FPM: VALUE APPLIED :0.5625*9.750 =5.484375
Din1-0.5625- 0_01111111110_0010_ 00000000_ 00000000 _00000000_ 00000000_ 00000000_00000000
Din2-9.750- 0_10000000010_0011_10000000_ 00000000 _00000000 _00000000_ 00000000_00000000
Dout-5.484375= 0_10000000001_01011111_000000000000000000000000 0000000000000000000

1(a). INTERNAL ADDER IN FPM FOR EXPONENT

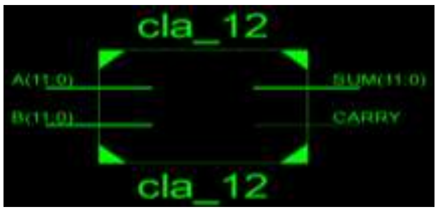


Fig 15 : RTL design 12 bit CLA adder for exponent

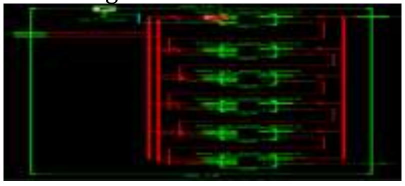


Fig 16: Technology schematic 12 bit CLA adder for exponent

The proposed double precision FPM is internally designed with CLA adder of 12 bit for exponent part calculation . The RTL design and technology schematic view of the proposed CLA 12 bit is shown in figure 15,16. The simulation results of the calculated values is shown in figure 17, here value applied in input a is 26 and input b is 92 is added and output sum value is 118.

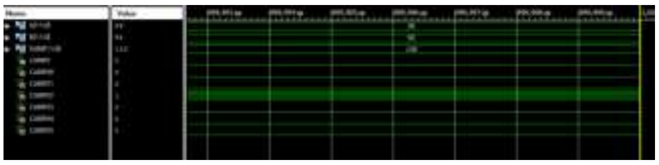


Fig 17: Simulation results of 12 bit CLA adder for exponent

1(b). INTERNAL MULTIPLIER IN FPM FOR MANTISSA

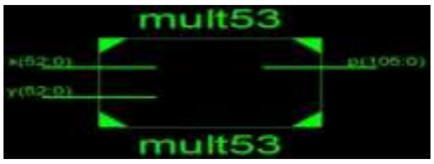


Fig 18: RTL Design 53 bit Hybrid multiplier for mantissa

Fig 19: Technology schematic of 53 bit Hybrid multiplier for mantissa

The proposed double precision FPM is internally designed with hybrid multiplier of 53 bit for mantissa part calculation . The RTL design and technology schematic view of the proposed hybrid multiplier of 53 bit is shown in figure 18,19. The simulation results of the calculated values is shown in figure 20, here value applied in input a is 33 and input b is 10 is multiplied and the output dout value is 330.

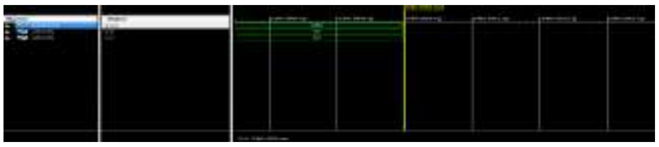


Fig 20: Simulation results of 53 bit Hybrid multiplier for mantissa

2.DOUBLE PRECISION OF FPA

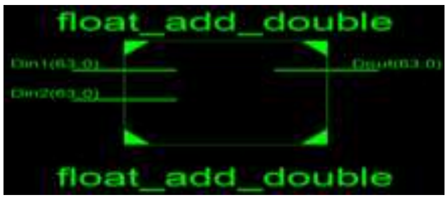


Fig 21: RTL design of double precision FPA

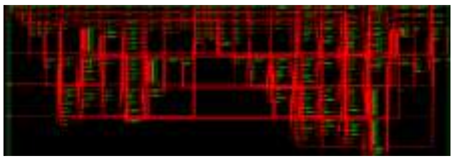


Fig 22: Technology schematic of double precision FPA

The proposed double precision FPA is designed. The proposed FPA is integrated internally with the proposed adder and RTL design and technology schematic view is shown in figure 21,22.The simulation results of the calculated values is shown in figure 23.

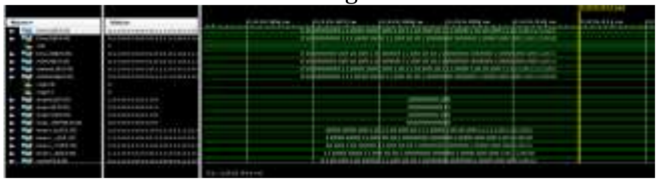


Fig 23: Simulation results of double precision FPA

Double Precision FPA: VALUE APPLIED:
24.538+28.278=52.816
Din1-24.538=0100000001001000 10001001 10111010 01011110 0011010100111111101111101
Din2-28.278=-0100000000111100 01000111 00101011 00000010 00001100 0100100110111010
Dout-52.816=01000000010010100110100001110010101100000010000001100010010011100

2(a).INTERNAL ADDER IN FPA FOR EXPONENT

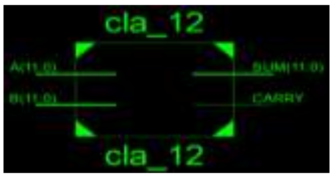


Fig 24: RTL design 12 bit CLA adder for exponent

The proposed double precision FPA is internally designed with CLA adder of 12 bit for exponent part calculation . The RTL design and technology schematic view of the proposed CLA 12 bit is shown in figure 24,25. The simulation results of the calculated values is shown in figure 26, here value applied in input a is 26 and input b is 92 is added and output sum value is 118.

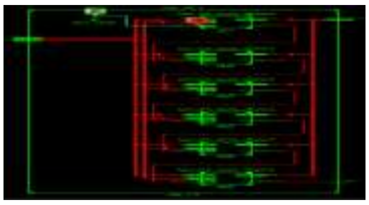


Fig 25: Technology schematic 12 bit CLA adder for exponent

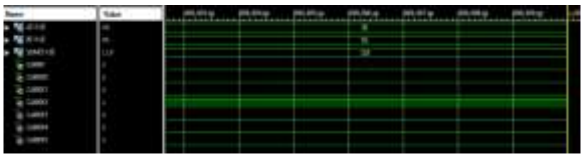


Fig 26: Simulation results of 12 bit CLA adder for exponent

2(b).INTERNAL ADDER IN FPA FOR MANTISSA

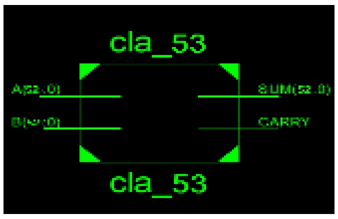


Fig 27: RTL Design 53 bit CLA for mantissa

Fig 28: Technology schematic 53 bit CLA for mantissa

The proposed double precision FPA is internally designed with CLA 53 bit for mantissa part calculation. The RTL design and technology schematic view of the proposed CLA 53 bit is shown in figure 27,28. The simulation results of the calculated values is shown in figure 29, here value applied in input a is 500 and input b is 250 is added and the output dout value is 750.

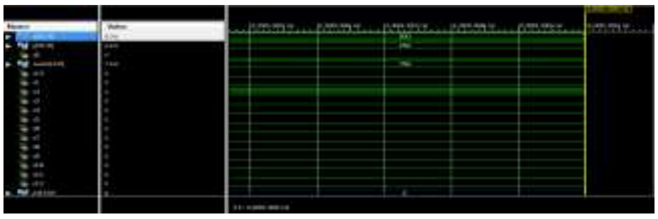


Fig 29: Simulation results of 53 bit CLA for mantissa

TABLE III: Comparison of multipliers

MULTIPLIERS	DELAY
ARRAY MULTIPLIER	12.150ns
WALLACE MULTIPLIER	7.815ns
BOOTHS	7.583ns
PROPOSED - HYBRID MULTIPLIER	7.045ns

TABLE IV: Comparison of adders

ADDERS	DELAY
CARRY SELECT ADDER(CSLA)	3.504ns
CARRY SKIP ADDER(CSA)	2.925ns
PROPOSED - CARRY LOOK AHEAD ADDER(CLA)	2.527ns

TABLE V: Device summary of proposed double precision FPM

PRECSION	TARGET FPGA	NO OF SLICE FLIP FLOPS	NO OF SLICES	NO OF 4 INPUT LUTS
DOUBLE PRECISION	SPARTAN 6	200	400	740

TABLE VI: Performance analysis of proposed double precision FPM

PRECISION	TARGET FPGA	FREQUENCY	DELAY
DOUBLE PRECISION	SPARTAN 6	82.5 MHz	16.12ns

TABLE VII: Device summary of proposed double precision FPA

PRECSION	TARGET FPGA	NO OF SLICE FLIP FLOPS	NO OF SLICES	NO OF 4 INPUT LUTS
DOUBLE PRECISION	SPARTAN 6	100	320	640

TABLE VIII: Performance analysis of proposed double precision FPA

PRECISION	TARGET FPGA	FREQUENCY	DELAY
DOUBLE PRECISION	SPARTAN 6	86.5 MHz	20.10ns

9. CONCLUSION

The demand for high performance and throughput and less delay Floating Point Multiplier (FPM) and Floating Point addition (FPA) unit has been on the rise during the recent years. So the study and assessment of different adders and multipliers were considered. To overcome the challenges such as more delay and less throughput, the proposed with hybrid multiplier and adder are designed and integrated into internal modules of Floating Point multiplier(FPM) and Floating Point Addition (FPA) of FPU of double precision formats.

10. FUTURE SCOPE

The proposed FPM and FPA module of FPU is designed for single precision floating point numbers.It can be further extended for 128 bit and can also be validated in additional modules respectively. Even it can be designed and tried in hardware for even better analysis. In fact by combining hybrid multipliers and adders and additional blocks that can achieve less delay and high throughput.

REFERENCES

[1] https://www.itu.dk/~sestoft/bachelor/IEEE754_article.pdf.
[2] https://www.ece.ucsb.edu/Faculty/Parhami/pubs_folder/parh00-text-arith-im-v2-w02.pdf.
[3] T.Menakadevi," Direct Digital Synthesizer using Pipelined CORDIC Algorithm for Software Defined Radio", International Journal of Science and Technology, Volume 2 No.6, June 2012,pp.372-378.
[4] Prerna ," VHDL Implementation of Addition and Subtraction unit for Floating Point Arithmetic Unit ",International Journal for Research in Technological Studies (IJRTS) on 2014,pp.31-34.
[5] https://www.techopedia.com/definition/13051/pipelining
[6] ConversionToolhttps://babbage.cs.qc.cuny.edu/IEEE-754.old/64bit.html
[7] Conversion Tool http://www.binaryconvert.com/convert_float.html
[8] Rashmi Rahul Kulkarni," Comparison among Different Adders", IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) Volume 5, Issue 6, Ver. I (Nov -Dec. 2015), PP 01-06
[9] https://www.johndcook.com/blog/2009/04/06/anatomy-of-a-floating-point-number/
[10] http://www.applied-mathematics.net/miniSSEL1BLAS/float-ieee754.pdf