A Multiplexer Based Normalization Architecture for IEEE-754 Floating Point Numbers

DISSERTATION-III

Submitted in partial fulfillment of the

Requirement for the award of the degree

Of

MASTER OF TECHNOLOGY

VLSI Design

By

CHEPURI PRAVALIKA

(11905340)

Under the Esteemed Guidance

Of

Dr. Rajkumar Sarma

Associate Professor

School of Electronics & Electrical Engineering

Lovely Professional University

Punjab

April 2021

CERTIFICATE

This is to certify that the Dissertation-III titled "**A Multiplexer based Normalization Architecture for IEEE-754 Floating Point Numbers**". That is being submitted by "**Chepuri Pravalika**" in partial fulfillment of the requirements for the award of Master of Technology, is a record of bonafide work done under my guidance. The content of this report, in full or in parts, haven either taken from any other source nor have been submitted to any other Institute or university forward of any degree or diploma and the same is certified.

Dr. Rajkumar Sarma
Supervisor
Associate Professor
(Lovely Professional University)

Objective of the Thesis is satisfactory /unsatisfactory

Examiner I **Examiner II**

ACKNOWLEDGEMENT

I would like to thank **Lovely Professional University** for giving me opportunity to use their resource and work in such a challenging environment. I am grateful to the individuals whom contributed their valuable time towards my thesis.

I wish to express my sincere and heart full gratitude to my guide "**Dr. Rajkumar Sarma** "Associate professor, who guides me to take-up this thesis in sync with global trends in scientific approach.

I owe my heartiest thanks to my parents & all those guideposts who really acted as lightening pillars to enlighten my way throughout this project that has led to successful and satisfactory completion of my Dissertation-III.

Last but not least, I would like to thank God for the strength that keeps me standing and for the hope that keeps me believing that this report would be possible. I would also thank the staff members of the department of Electronics and Communication Engineering who have been very patient and co-operative with us.

DECLARATION

I, Chepuri Pravalika, student of Master of Technology under Department of Electronics and Communication Engineering of Lovely Professional University, Punjab, hereby declare that all the information furnished in this dissertation-III report is based on my own intensive research and is genuine. This dissertation-III, to the best of my knowledge, does not contain any work which is not done by me.

Date:

Chepuri Pravalika

Reg No. 11905340

M.Tech (VLSI Design)
Lovely Professional University

ABSTRACT

Floating point numbers is an extensive data type in computation. Floating point processing is used in numerical applications. It is the process of efficiently organizing data in a database. For Digital signal processors and Digital image processors, the dynamic range and precision of floating point numbers is important to decide between these two processors to accomplish for an application. When the two floating point numbers of equal or identical exponents for subtraction, then in such case normalization process is performed so that it can compatible with IEEE format. To obtain greatest precision, the number is normalized and also to achieve high performance using IEEE-754 standard. Designing of Multiplexer based normalization at transistor level will increase the performance and to gain great accuracy. The cadence virtuoso is used for designing the normalization architecture at transistor level using different technologies like 90nm gpdk or tsmc 130nm.

Table of Contents

LIST OF FIGURES

CHAPTER 1

Introduction:

Normalization refers to a process that makes something more normal or regular. In practical problem, if we use fixed point representation the range may not be sufficient. So, for real numbers in computers, normalized Floating point representation is used. In this 32 bits are divided into two parts i.e., mantissa along with its sign bit and exponent too [1]. The fractions of mantissa represent with a leading bit and exponent power 2 is multiply by mantissa. By using 32 bits, the range can be increases. Binary format of floating point numbers is given as [2]

Sign x mantissa x $2^{\wedge +/- exp}$

S (1 bit)	E (8 bits)	M (23 bits)

Fig no 1: IEEE 754 standard word

1.1 Types of Floating point Representation:

1.1.1 Fixed Floating point representation

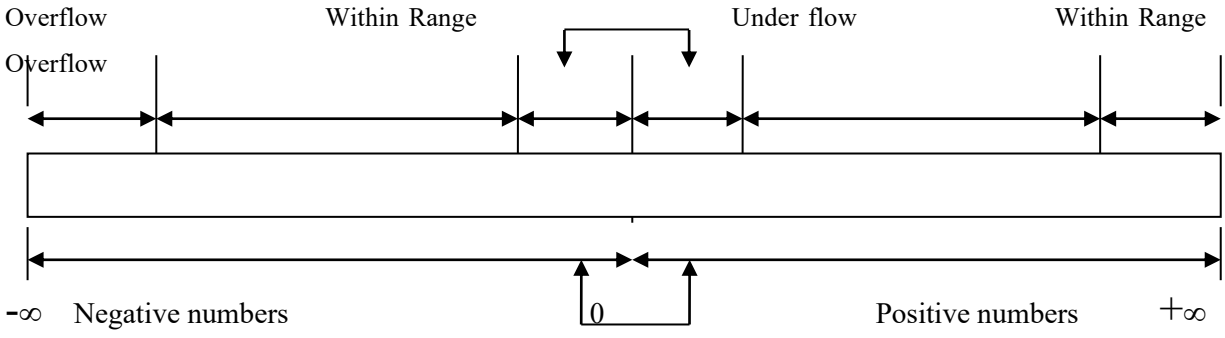
The number of digits is fixed before or after the decimal point.

1.1.2 Floating point representation

The number of digits is not fixed before or after the decimal point. i.e, decimal can float.

Comparing both representations, floating point is more advantageous than fixed point; it is able to handle large range values.

Floating-point Range: It mostly depends on the number of bits in the representation, significand and exponent component [3].



Denormalized

1.2 Multiplication Algorithm:

i. Multiplying (1.M1*M2).

ii. Keeping of decimal point in result.

iii. Exponents Addition; (E1+E2-Bias).

iv. Getting of sign; s1 xor s2.

v. Normalization process

vi. Rounding

vii. Observe overflow & underflow [4].

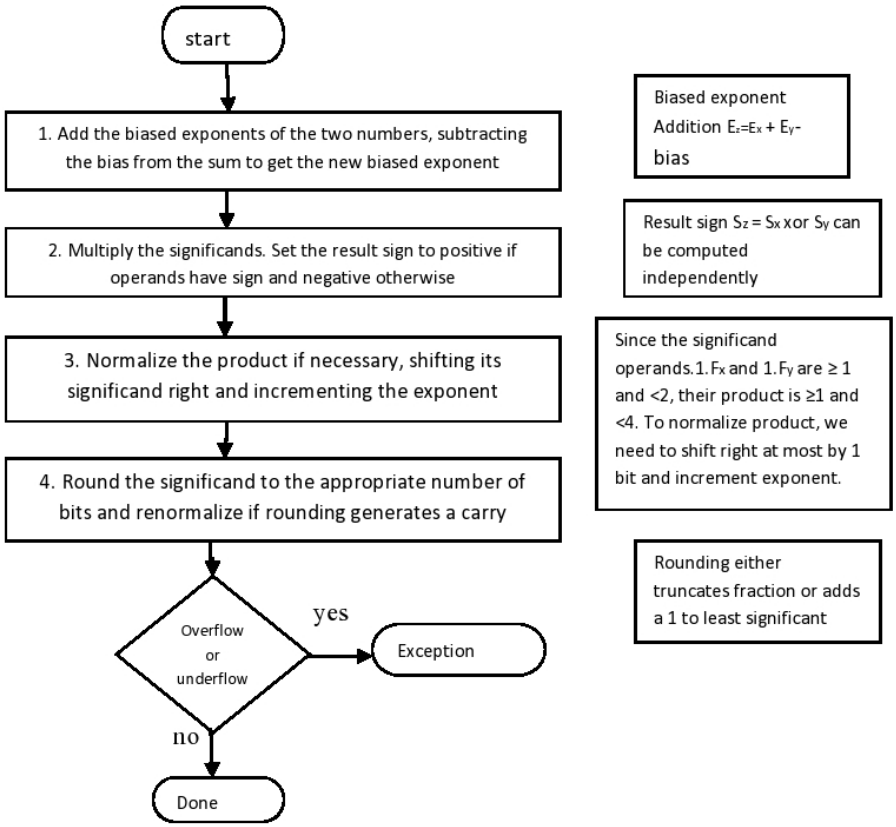


Fig 2: Floating Point Multiplication

Overflow/Underflow means if the E result is large/small for representing. Overflow occur when two exponents are adding and underflow too. This can be compensated by normalization in which the result should be in specific bit size otherwise it is not a normalized one [5].

To increase in area, power and latency, to improve in three-term adder, more techniques are applied such as pipelining, alignment, leading zero anticipation. Add new exponent for comparison [6].

Evaluation is done on area and critical paths and a correction method was introduced in order to give the exact result, do not included a logical levels because to analyze the input operand instead of summation [7].

Using inexact fp adders improves power consumption and delay and processed of dynamic quality of images range [8].

Old designs are applied for all different applications for better accuracy to save circuit parameters by using inexact circuits computation and compared with existing accurate ones [9].

Procedure of Normalization is:

1) Counting of maximum zeroes.
2) Shifting with counted number.

It majorly consists of Leading zero counter and shifter. To count the number of zeros, LZC is used. The shifter moves the MSB to left side according to result [10].

For designing of Single and Double precision the key points has to be consider that are maximum frequency, hardware resources and clock latency. An operation like High precision requires high dynamic range leads to requirement of efficient processors. There are vast Digital signal processing applications using basic floating point addition and subtraction [11].

In embedded, especially in older processors do not have hardware support. Almost every language has a data type; compiling takes place at a time which responds to certain exceptions of respective standards such as overflow [12].

1.3 Standard IEEE-754 Formats:

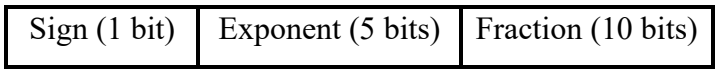
In 1985, the IEEE introduces the standard binary floating –point format which includes different types such as single and double precision, rounding mechanism, arithmetic operations, etc. It represents as by the given equation. Z = (-1s)*2(exp-bias)*(1*M). The different standards defined in the IEEE 754 format are shown in table

Name	Common name	Base	Digits	Emin	Emax
Binary 32	Single Precision	2	23+1	-126	+127
Binary 64	Double Precision	2	52+1	-1022	+1023
Binary 128	Quadruple Precision	2	112+1	-16382	+16383
Decimal 64		10	16	-383	+384
Decimal 128		10	34	-6143	+6144

Table 1: IEEE-754 format Standards

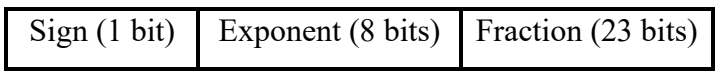
For Binary floating point numbers, the first three formats of 32,64 and 128 are used and remaining are used for Decimal Binary floating point numbers of 64 and 128 bits.

IEEE Half-Precision format: It consists of 16 bits for representation of a floating point number. The format is divided into three fields. The first field comprises of sign bit for fraction part, 5bits comprises of exponent field and ten bits for fractional part. The sign bit represents the number either positive or negative number. If the sign bit is 0 then it is positive number and 1 for negative number. To represent the number in IEEE format, first it has to be converted into a normalized scientific notation containing before binary point should be one bit simultaneously adjusting of exponent.



IEEE Half-Precision floating point format

IEEE Single-Precision format: It consists of 32 bits for representation. One bit for sign bit, 8bits for exponent part and 23 bits for fractional part.



IEEE Single-Precision floating point format

CHAPTER 2

Problem Background:

The main purpose of binary number is to make normalize by shifting numbers either left or right. The shifted number of bit-positions is treated as singed integer. For the negative int is said to be encoding of exponent. It can hold all the “significant bits” of number.

Numbers must have minimum storage requirement and arithmetic operations should be effective in operation.

Algorithms which appears impractical put into practice now became easy to realize with desirable parameters tends to upcoming designs can be incorporated and to represent large value numbers, range is required so as to integers would represent using IEEE 754 standard.

In present scenario, precision plays a vital role and goal is to have a less area in ALU architecture. One of the problems is to identify the position of nonzero digit. Generally, the delay of normalization block more than shifter because it is having carry propagation. Research has been focused for reduction of LZC block.

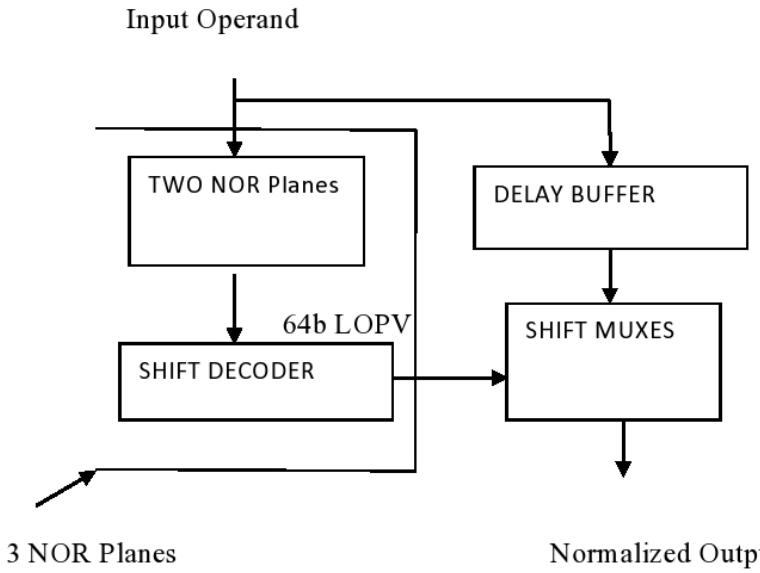


Fig 3: Proposed Normalization scheme for 64b operand

CHAPTER 3

Scope and Objectives of the Study:

- To incorporate IEEE 754 format with existing Normalization Architecture.
- The detailed analysis of the IEEE 754 will be obtained in the process of designing normalization architecture.
- The normalization takes place in order to make this number compatible with the IEEE floating point format.
- To implement a Novel Multiplexer of Normalization Architecture capable to incorporate IEEE 754.
- A comparison will be made for the designed Normalization architectures with different technologies are gpdk 45nm, 90nm and tsmc 130nm.
- Design of architecture and analysis with IEEE 754 will give better understanding of precision.

CHAPTER 4

Review of Literature:

Kyung-Nam Han et al., 2002: proposed a method for directly shift amount for optimizing LZCs or LZAs. The amount information is decoded from the LOPV using a adding a NOR plane and sent to shifter [13]

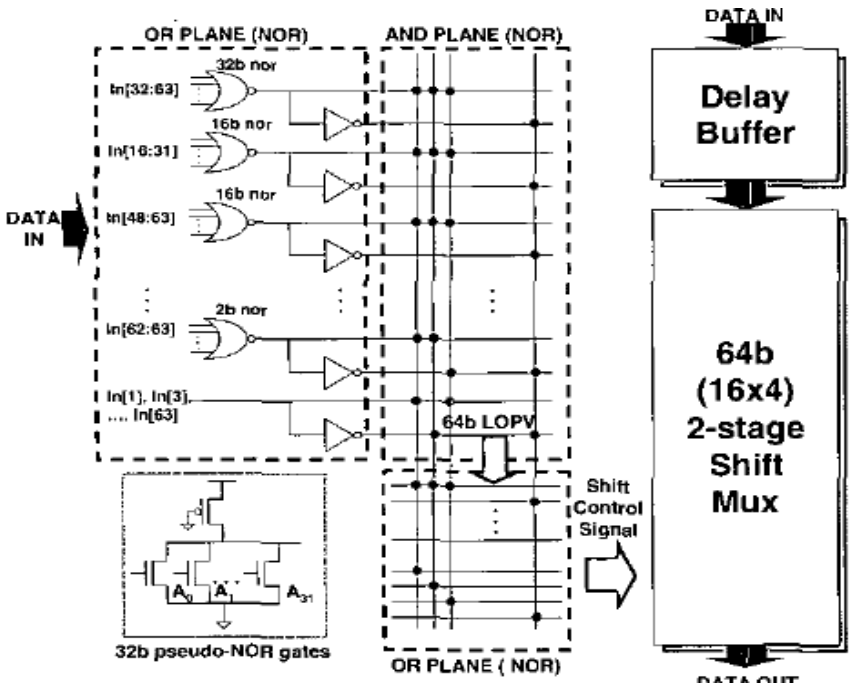


Fig 3.1: Circuit diagram of proposed normalization unit with three NOR planes and a shifter.

Jae Hong Min et al., 2013: compares the fused magnitude unit with conventional discrete units. It computes two 24 bit significant squares in parallel. The magnitude operation used for many graphical and signal applications and to finding complex numbers [14].

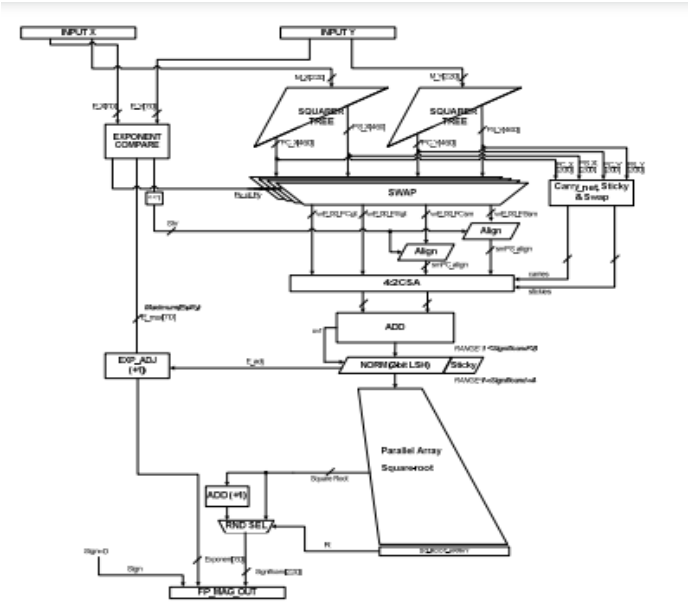


Fig 3.2: Floating point fused Magnitude unit.

Jongwook Sohn et al., 2016: four-term dot product was designed in a single unit to achieve good performance compared to traditional designs. This was implemented and synthesized with CMOS technology [15].

P M Drusya et al., 2016: The proposed system has reduced area than the previous existing adders and it is related to a discrete design [16].

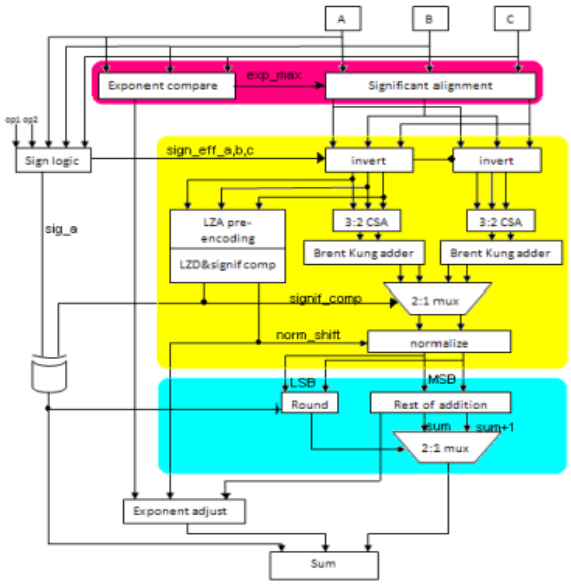


Fig 3.3: Area efficient fused floating point three term adder

Preeti Khobragrade et al., 2016: Introduced a method for reduction in delay. Predicted the error bit of shift amount and proposed for error correction logic to the result obtained [17].

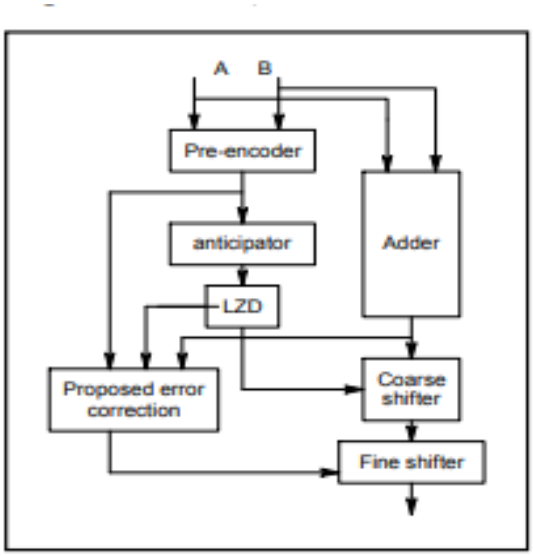


Fig 3.4: Block representation of proposed method

P.M.Seidel et al., 2004: Implemented for low cost and reduces the power dissipation which is completely compliant with IEEE standard of multiplication and division [18].

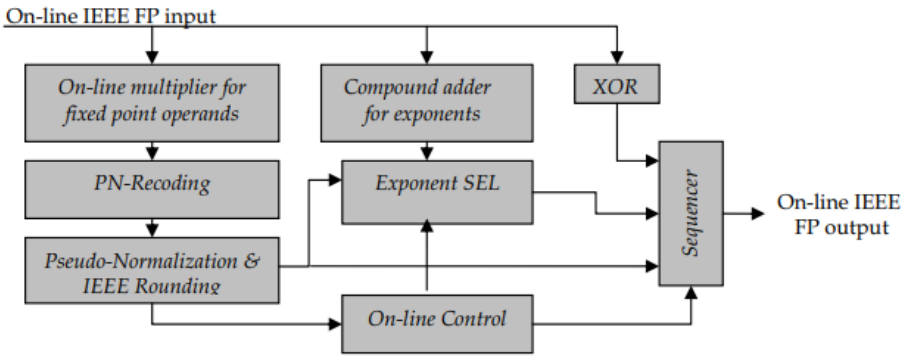


Fig 3.5: Structure of proposed Implementation

Jong wook Sohn et al., 2013: Dual path designs reduce latency compares to dot product unit. By analyzing data flow, three pipeline stages are proposed for design to increase the throughput [19].

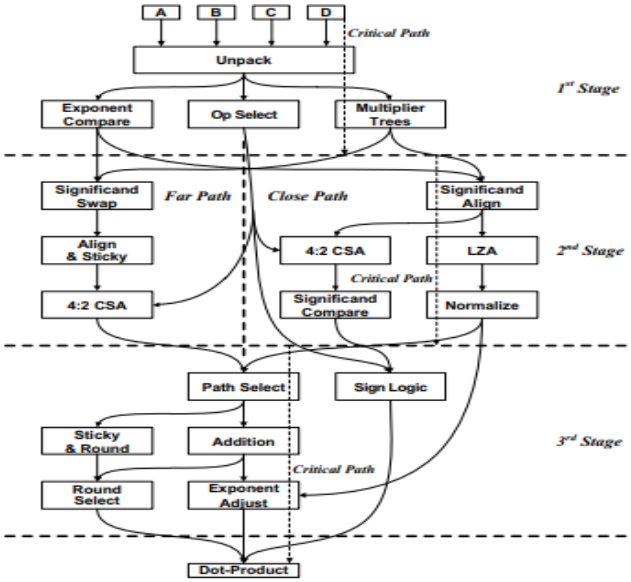


Fig 3.6: Data flow and critical path of a dual path fused dot product unit

John wook Sohn et al., 2014: Increase the throughput and design is implemented for both single and double precisions by improving the fused three term adder [20].

Fig 3.7: Data flow of a pipelined improved fused floating-point three-term adder.

Anatoly I. Grushin et al., 2008: Normalizing and rounding are implemented and presented a lza. An algorithm was presented for implementation of underflow path described in verilog [21].

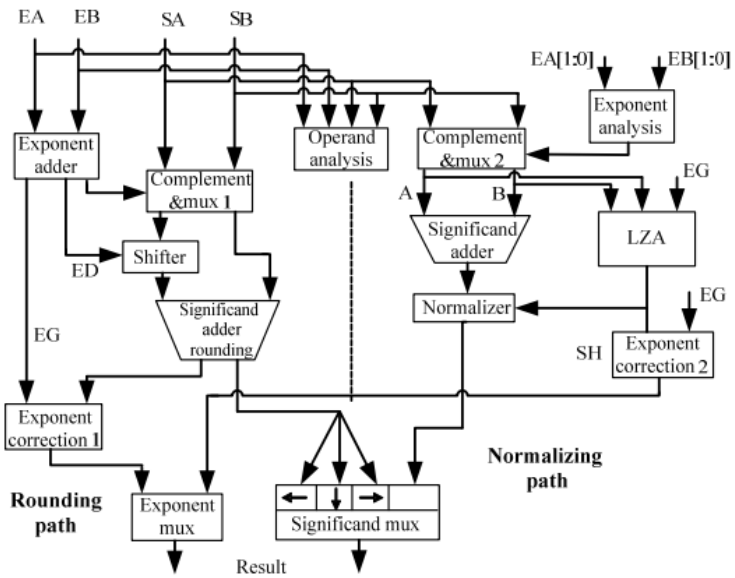


Fig 3.8: FP adder consisting of Normalizing and Rounding paths.

Fang Jianping et al., 2004: To improve the speed in DSPs, it is needed to develop a LOP is of logical way which can apply to many adders. A 32-bit FA operation is fast than the existing LOD algorithm [22].

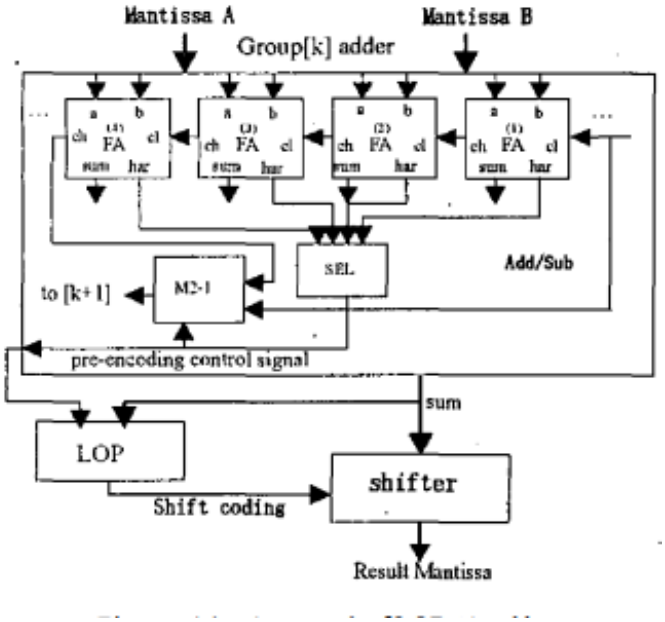


Fig 3.9: Novel approach of LOP algorithm

H. Suzuki et al., 1996: proposed a new logic of LZA to enhance speed. It was defined by Boolean functions consisting of less number of transistors by five stages in circuit and for greater in speed similar as pass transistor logic [23].

Giorgos Dimitrakopolous et al., 2008: Very significant reduction of energy are got by detail mathematical approach and for maintaining error of certain logic a novel technique was used that shows the minimum move in further without limits in shifter [24]

Richard J. Zaccone et al., 1987: For unnormalized output of digit on-line can cause a problem for divide and square root. To remove this two possibilities are explained for getting correct outcome by combining mentioned techniques in this paper [25].

Nisha Singh et al., 2018: work is done for single precision and modeled in verilog HDL for efficient modules of subtraction and addition so as to decrease the count of gates. RTL code is synthesized using compiler for 180nm gpdk with suitable constraints [26].

H. Yamada et al., 1995: A high noise tolerant precharge CMOS flow was developed for accelerating paths of critical and time in carry lookahead. Each delay was calculated by circuit simulation and to accomplish the 13.3ns execution time [27].

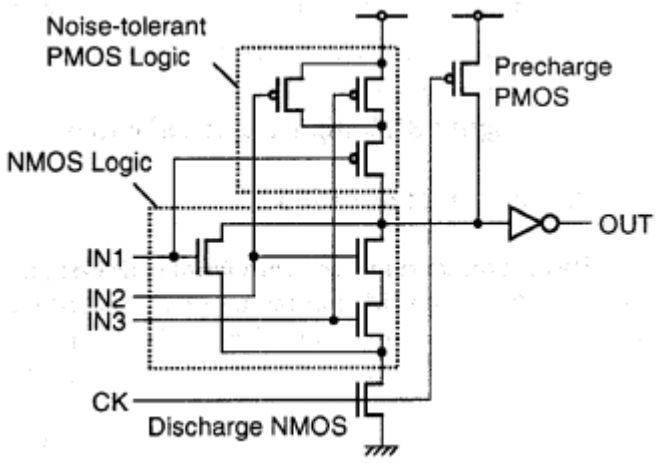


Fig 3.10: NTP circuit block diagram

E. Antelo et al., 1998: Introduced for effectively usage over complex numbers and CORDIC processors by a conventional two operand normalization to suppress the need of comparing of both operands leading 0's [28].

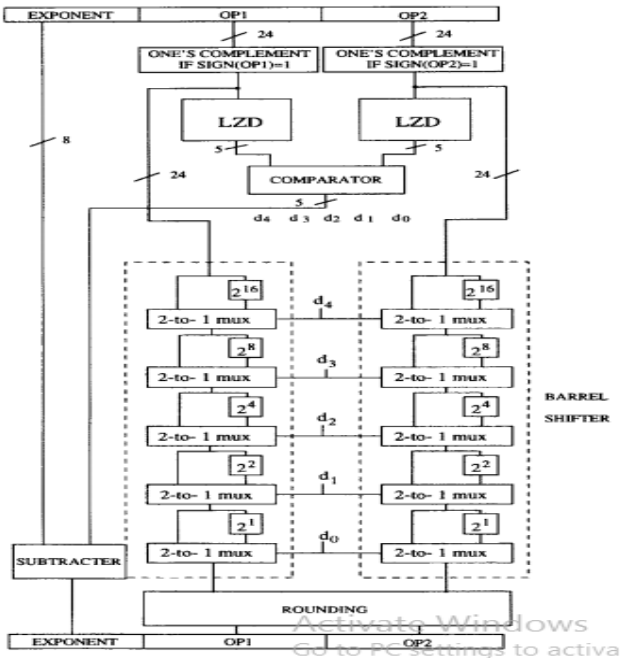


Fig 3.11: Conventional two operand normalization

Peter Korner et al., 2009: To determine the location in non-redundant representation. When determining the shift is done along with converting into specified represent for correction. In this paper, gives a algorithm to find the approximate place [29].

T.Lang et al., 2004: This work introduces the MAF such that Normalization takes place before addition taken other parts like LZA and dual adder for performing this and allows to balance the require parameters [30].

J.D Brugera et al., 2005: The main point is latency reduction of addition in the sense not to get alignment parallelly with multiplier and later multiplication was bypassed in the block of addition and estimation is done on delay, compared with old single data path MAF [31].

A. Beaumont-smith et al., 1999: modified a version to use a less quantity of hardware in parallel node of processor. This was achieved by providing feedback to un-normalized. The only additional hardware is require here for 2-cycle fp adder was compared [32].

Suresh Srinivasan et al., 2013: Demonstrated the double precision FPMAC which gives the best outperformance known for IBM6 power in timing and sponsoring extra power benefits due to split handling. Simulation is done on individual blocks; this innovative design helps the microprocessors with good convergence [33].

J. Eldon et al., 1982: In this paper everything which involves in main process has been explained individually and for respective format TRW develops a chip 10 MHz for arithmetic operations. Signal processing is needed for handling various signals as well as for noise, not indulging overflow [34].

Weiqiang Liu et al., 2015: For image applications of dynamic range, the proposed designs of small area and offers high performance than their equivalent exact. This can be modeled in VHDL and outputs are analyzed to check for meeting of desired values [35].

D.W.Sweeney., 1965: In this paper basic functionality of overall process was elaborated along with that notation, data gathering, binary radix and other radices; concentrated on the addition part also, for all the worst cases and analyzation on set of problems and some choices are includes for designing [36].

Andre Guntoro et al., 2009: In this paper, different wavelet filters takes into account such as DWT/DWP which supports memory sizes for various applications demands in order to compute forward and inverse transforms [37].

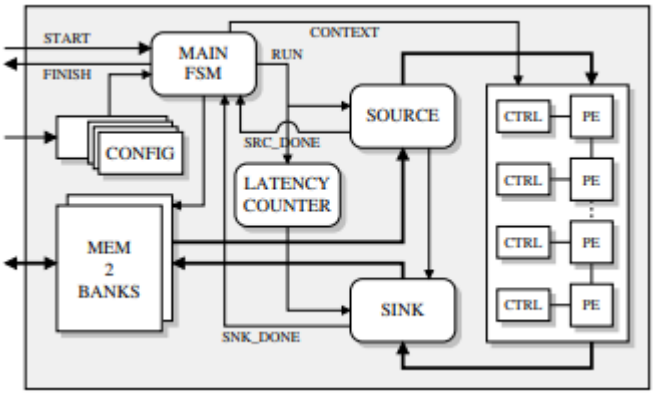


Fig 3.12: Wavelet Processor

CHAPTER 6

Proposed Research Methodology:

In this thesis work, Normalization architecture is designed in the Cadence virtuoso. Each Primary block like adder and comparator was designed using this tool.

- Floating point representation able to retain its resolution and accuracy compared to fixed point representation.
- Cadence is a software and hardware design company. It supplies tools to its customers in the areas of 5G communications and medical applications. The Cadence Virtuoso software is used for the design and simulation is done by using ADEL tool.
- There is a vast range of technologies that can be used for the design in the Cadence tool. For this design out of the available 90nm or 45nm gpdk technology or TSMC 130nm technology we can use any gpdk.
- The required parameters like power and delay for individual blocks calculations are made with the help of ADEL tool.

CHAPTER 6

Expected Research Outcomes:

1. All the contents of architecture like Exponential adder, Exponential comparator and Exponential Shifter were designed at transistor level.
2. A paper based on the thesis work can be published in a good quality journal.
3. An Efficient Contribution on field of Normalization which makes the society in the better precision and to perform IEEE 754.
4. A patent or copyright can be filed based on the novelty of the architecture of the Normalization.
5. This work will contribute in the field of IEEE 754 standard to the society and floating point normalization is helpful for minimization of accuracy problems etc.

CHAPTER 7

Proposed Work plan with Timeline

The main Components in Normalization architecture like Adder, Comparator and Shifter are made in Cadence virtuoso. The architecture has to be designed with the above building blocks to incorporate with IEEE-754 format. The detailed analysis of architecture is done; a research paper will be published in a good quality journal and conference also. The timeline for work is given in the figure.

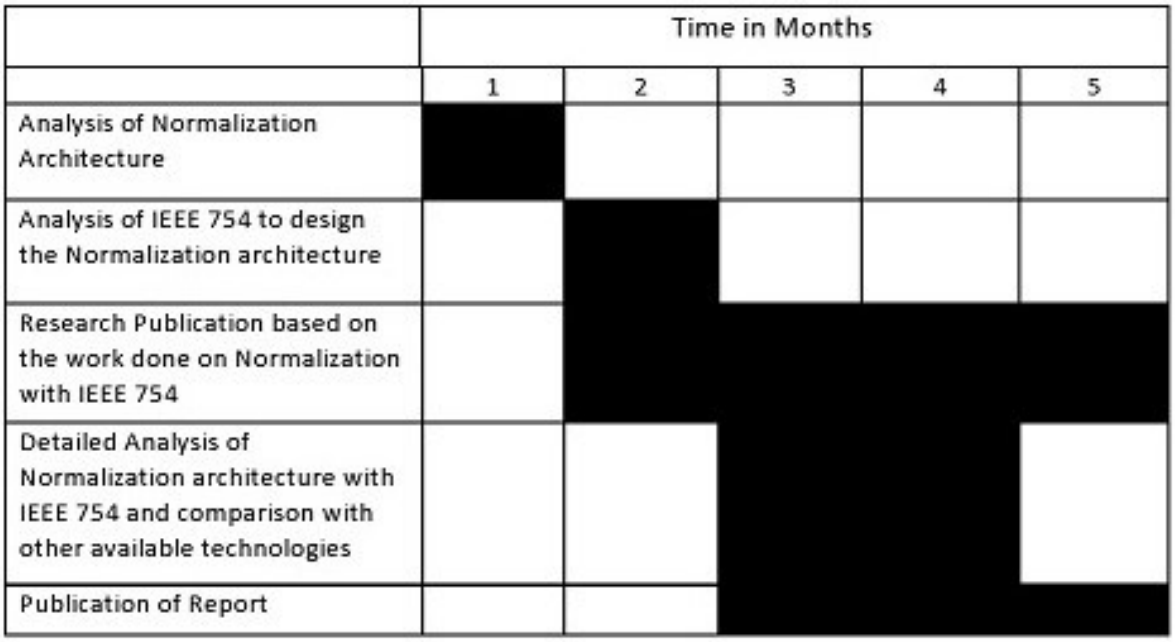


Fig 3.13: Proposed work plan with a time plan

CHAPTER 8

Experimental Work:

The process of ensuring that maximum accuracy of a number for a fixed number of bits is known as 'Normalization'. Normalization ensures that maximum accuracy of a number for a given range of bits. It also ensures that each number has only one possible bit pattern to represent it. Floating point number is used to enhance the range of representation.

In this work, Normalization of floating point numbers process is performed. The inputs of Half Precision are arranged in a format such as given below:

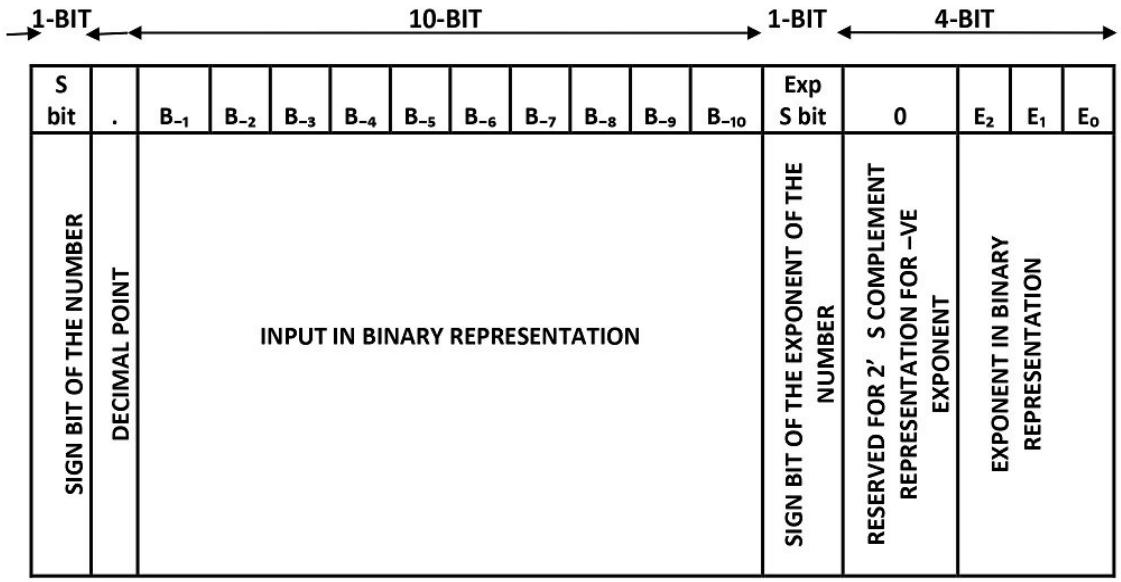


Fig 4: Floating point Input Structure

The size of the input is 16bits, in which two bits are reserved for sign bits of the number and its exponent. The representation of positive or negative number is depends on 0 or 1 of the sign bit, respectively. Ten bits are used for Binary representation and four bit exponent representation in binary. In binary representation the important point is to consider that the 4^{th} bit of the exponent is made default as '0' for representation of 2-bit number in 2's complement form. The range is given by $-(2^{n-1}) to + (2^{n-1} - 1)$ n = number of bits.

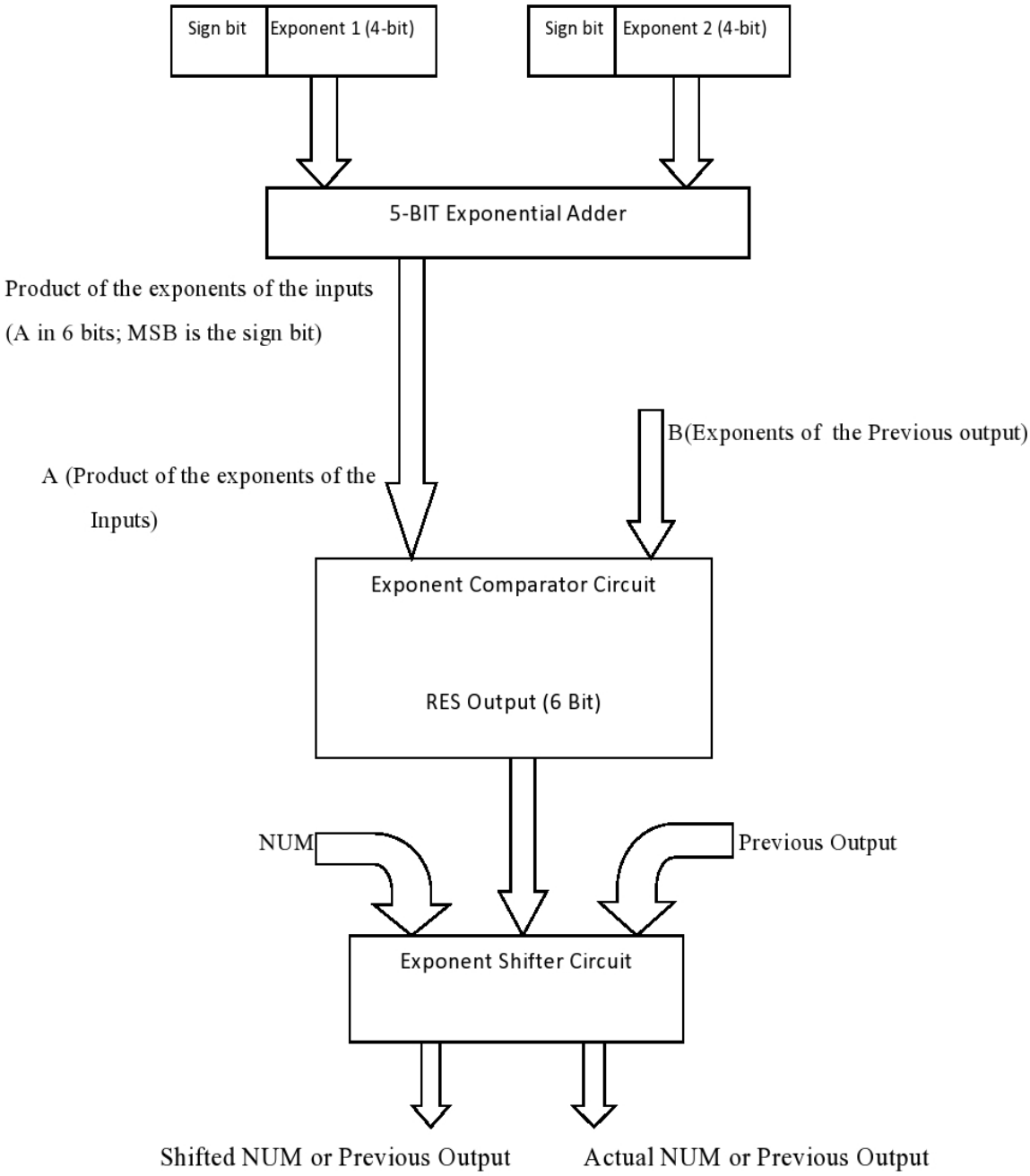


Fig 5: A Multiplexer based Normalization Architecture

In this architecture, the exponent can have range from -8 to +7. The architecture inputs should be entered in decimal point only.

The primary content of the Normalization Architecture are:

- Exponential Adder
- Exponent Comparator Circuit (ECC)
- Exponent Shifter Circuit (ESC) and
- 2:1/4:1 Multiplexers with different sizes

8.1 Exponential Adder (EA):

The EA block performs the addition of inputs exponents. The inputs given to the EA block is of five bits (including one sign bit), it produces the result in 6 bits. Along with that the MSB bit (i.e., 6th bit) is the sign bit of the result. The size of each exponent is 5-bits in which one bit is reserved for sign representation. The following steps are followed in the EA block:

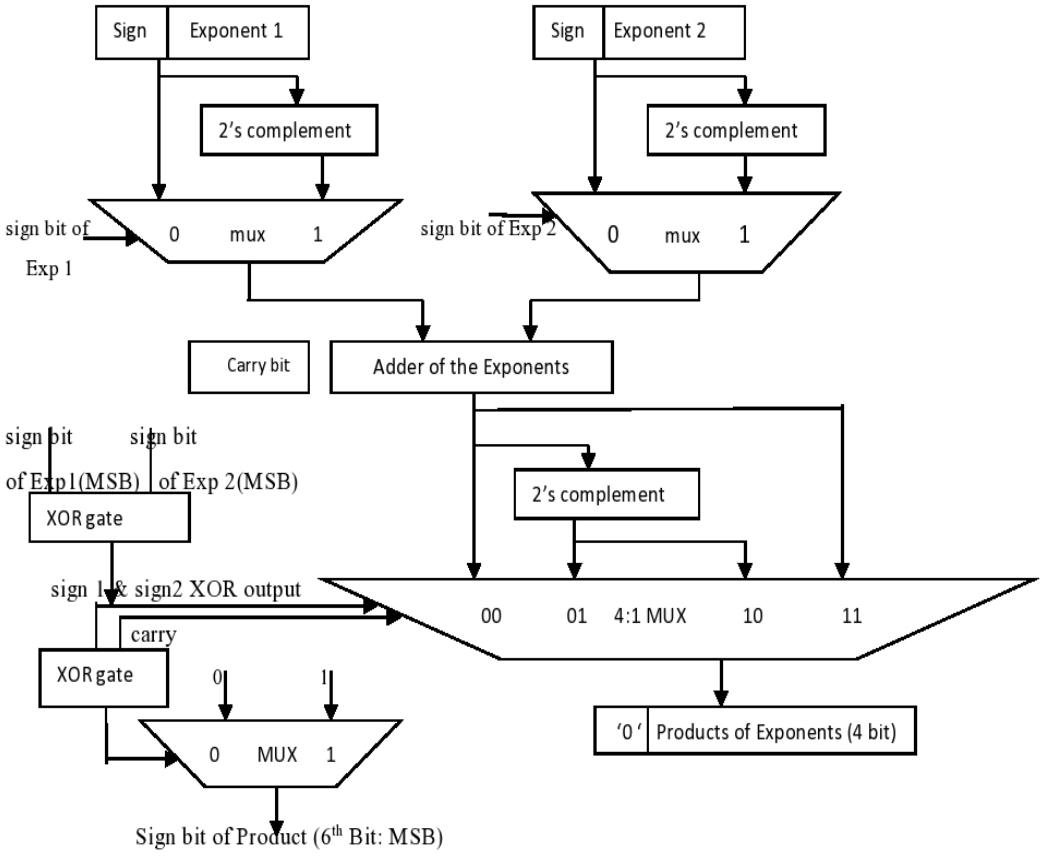


Fig 6: Exponential Adder (EA) Architecture

i) The Exponents are represented in 2's complement form which depends on the sign bit.

ii) Actual/ 2's complement form of both exponents are added using 5-bit adder block.

iii) The Adder block does not provide exact result of sum whereas the inputs to the adder block is of actual /2's complement form. So, the adder output is further processed through a

4-bit 4:1 multiplexer for representation of result in sign magnitude form. The result of the 4-bit 4:1 multiplexer follows the below conditions:

- If both the sign and carry bit of adder block is either ‘00’ or ‘11’ of XOR output then pass, the output of adder is itself is the 4:1 multiplexer output 4-bit.
- If they are ’01’ or ‘10’ then the 4-bit 4:1 multiplexer output is the 2’s complement representation of the adder block.

iv) The EA output is in sign-magnitude form only and the last bit of the EA block depends on the sign bit of exponent 1 and exponent 2 and carry bit of adder block i.e., XOR value.

Exponential Adder was designed and implemented in Cadence-Virtuoso using 90nm gpdk technology.

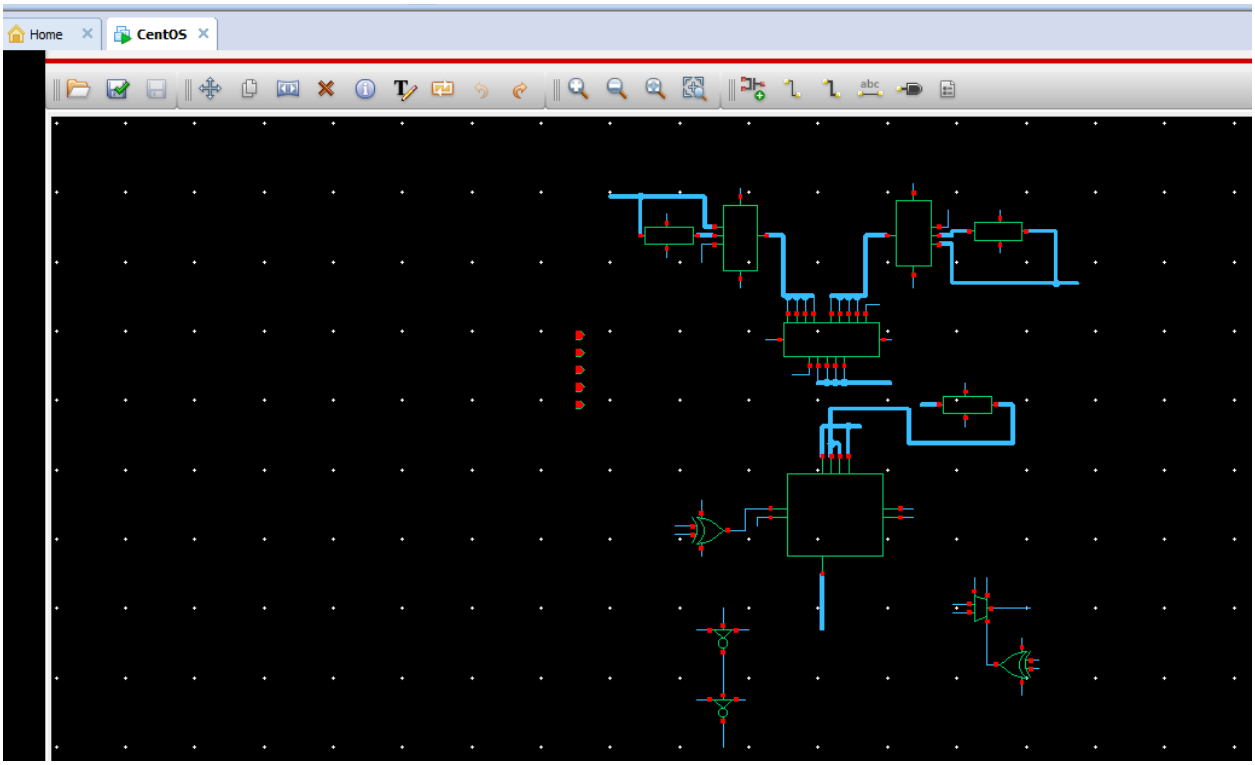


Fig 7: Exponential Adder Architecture designed in Cadence 90nm gpdk

8.2 2’s Complement 5-bit:

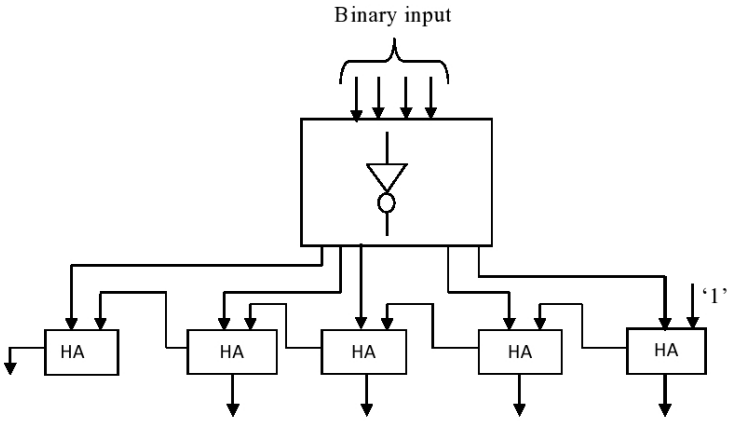


Fig 8: 2’s Complement Block

A 2’s complement block consists of n-bit inverter along with half adders. This block is responsible for representing a negative binary number in 2’s complement form or vice versa. 2’s complement block is designed in Cadence tool using 90nm gpdk technology.

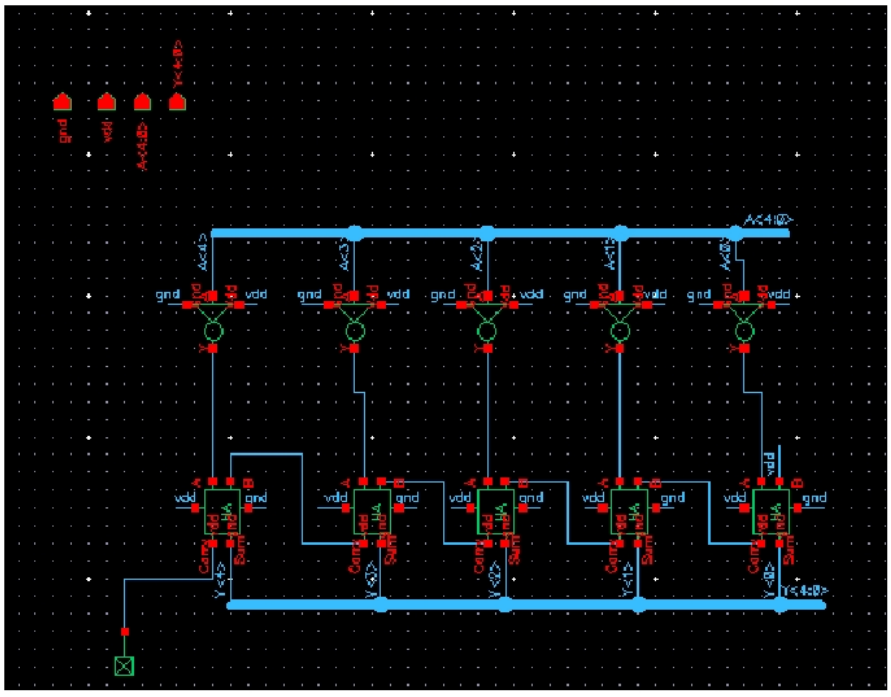


Fig 9: 2’s Complement block 5-bit design

2:1/4:1 Multiplexers of Different Size inputs:

In this work, the algorithm does not use programming code. So, based on the conditions multiplexers of different sizes with multiple or single bits is considered.

8.3 Exponent Comparator Circuit (ECC):

For ECC the inputs are the product of the Exponents (EA output i.e., 6 bit) and output exponent of the previous cycle (6-bit in size). The operation of ECC block is as follows:

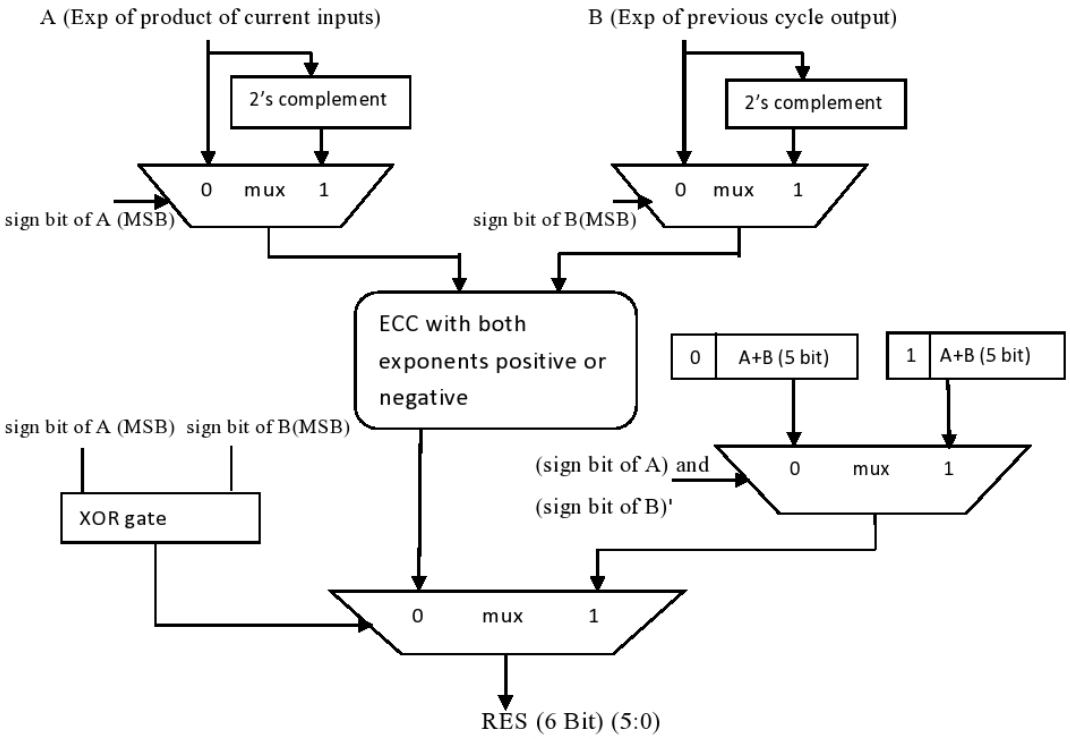


Fig 10: ECC Architecture

i) ECC inputs are represented in 2's complement form based on sign bit.

ii) Based on sign bits of ECC inputs the operation is further precede as follows:

a. If both the sign bits are different, so add the inputs of ECC to produce a 5-bit output (i.e., discard the carry bit) and introduce the 6^{th} bit is '1' if the product of inputs exponents is negative or make the 6^{th} bit is '0' in other circumstances.

b. If they are same, then find the higher input number among two inputs and find the difference between two inputs as following procedure:

   - Compare both numbers bit by bit, starts from MSB to LSB to find the higher number.
   - Use 2's complement block to find the difference. The difference produce a 5-bit output (i.e., discard the borrow bit) introduce the 6^{th} bit as '0' if Exponent

inputs product is higher than the previous cycle exponent. Make the 6th bit as 1 in other cases.

- Multiplexers are used to compare the inputs.

iii) This operation produces a 6-bit output, further used to perform the binary shift.

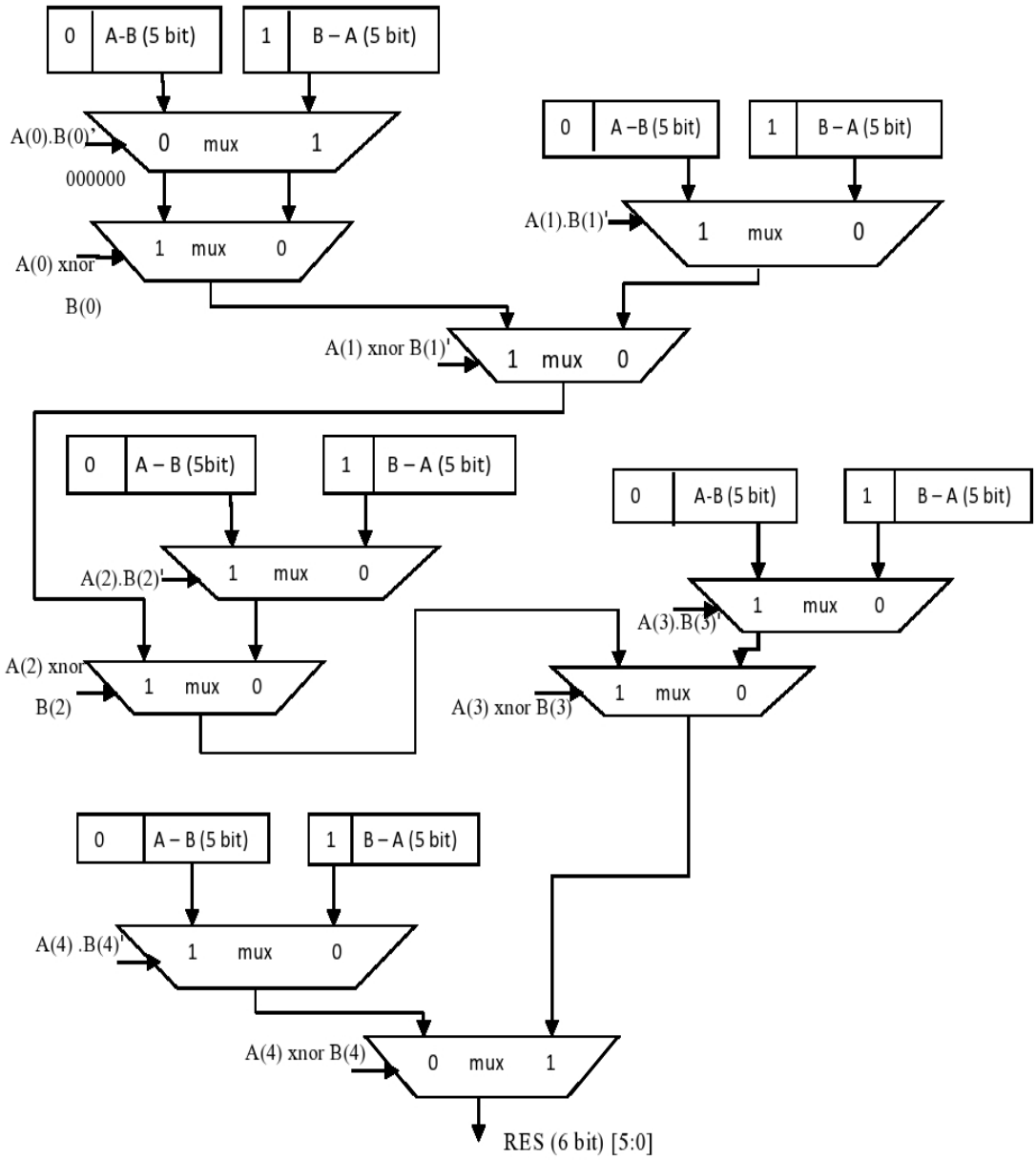


Fig 11: The ECC with same sign bit

Exponent Comparator Circuit was designed in Cadence tool using 90nm gpdk technology

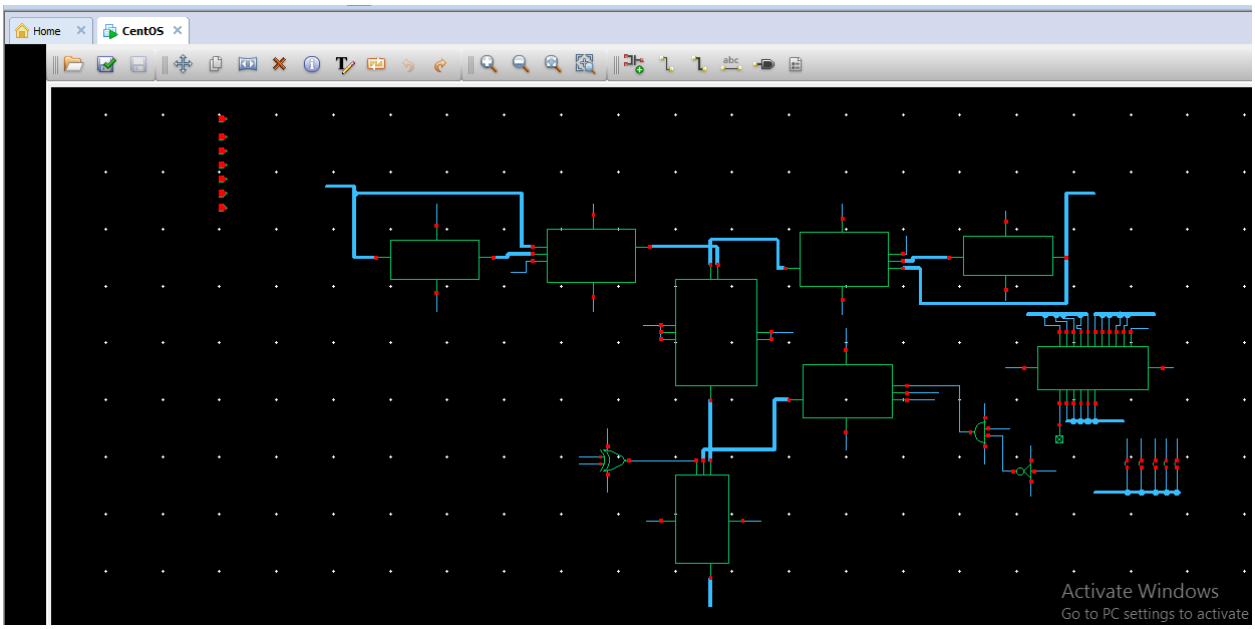


Fig 12: Exponent Comparator Circuit

Fig 13: Exponent Comparator of 5-bit

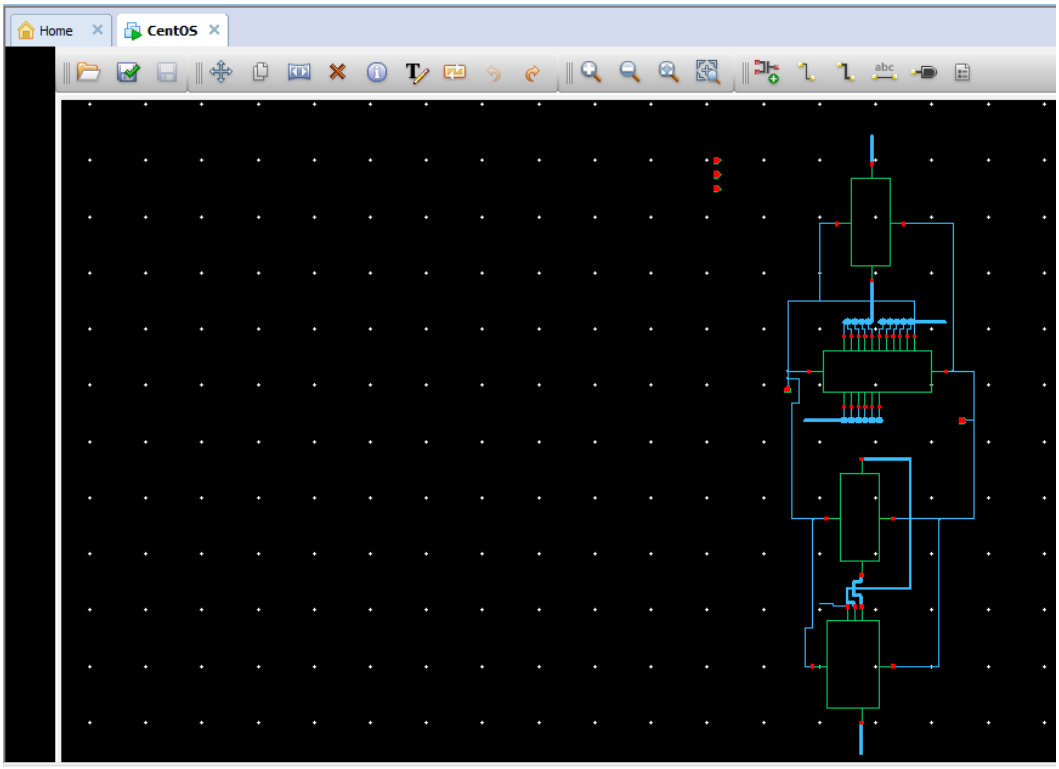


Fig 14: Difference of two inputs (A-B)

8.4 Exponent Shifter Circuit (ESC):

The shifter Circuit is responsible to shift the smaller number by the amount of difference between these two exponents. The inputs to the ESC block are 6-bit output of ECC block, a 32-bit product of the inputs and 32-bit value of the previous cycle output. The process has to be following as:

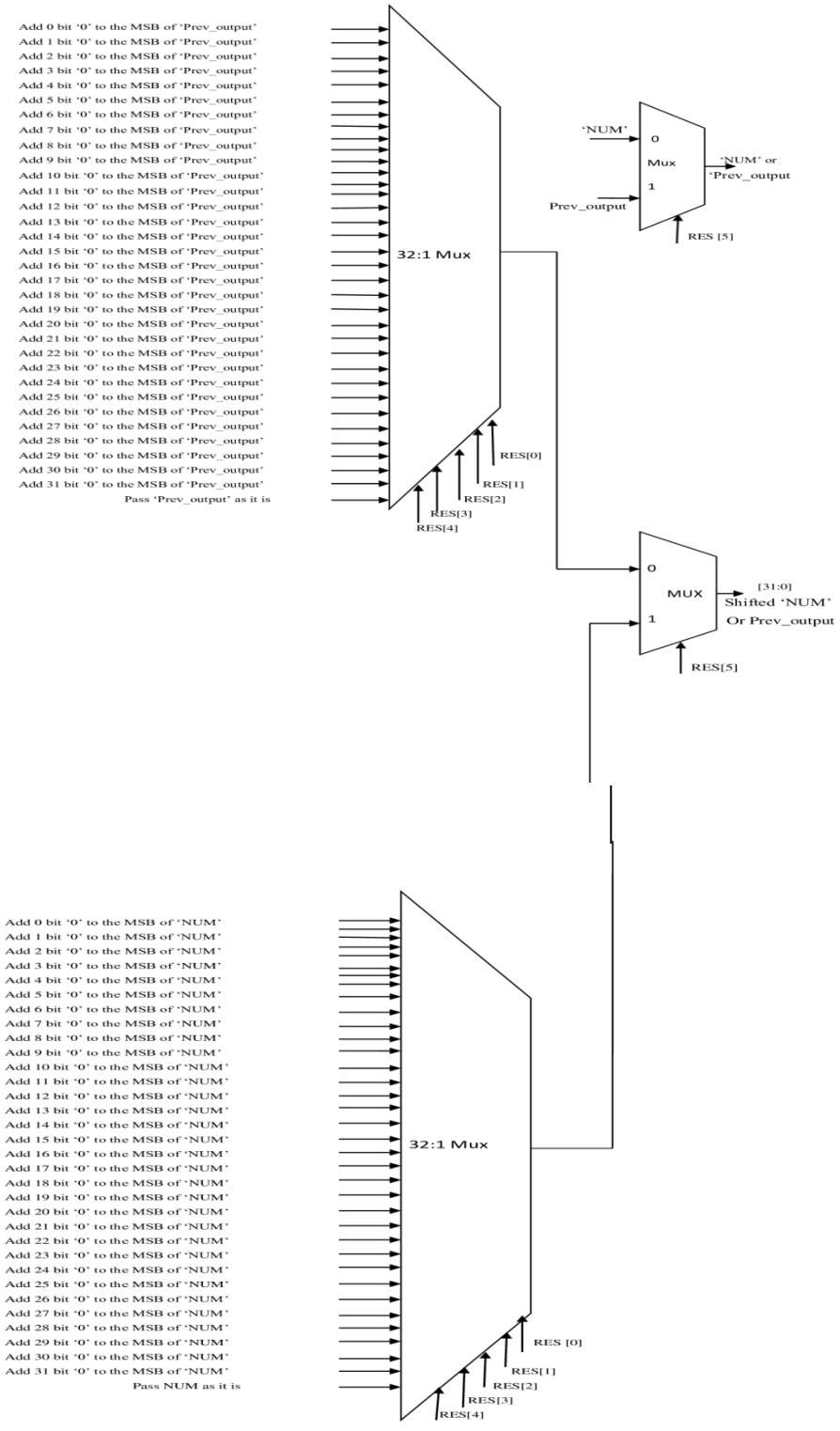


Fig 15: The ESC Architecture

i) As shown in figure, the smaller number identification is made based on the ECC output (6 bits). If the MSB is '1' of ECC block output, then product of inputs is shifted towards the right by the equivalent decimal value of the remaining 5-bit binary of ECC block output. On

the other hand, if the MSB is ‘0’ of ECC block output, then the previous output is shifted towards right by the equal decimal value of the remaining 5-bit binary of ECC block output.

ii) The input to the ESC block, which is needed not to be shifted, is identified by the same MSB of the ECC block output.

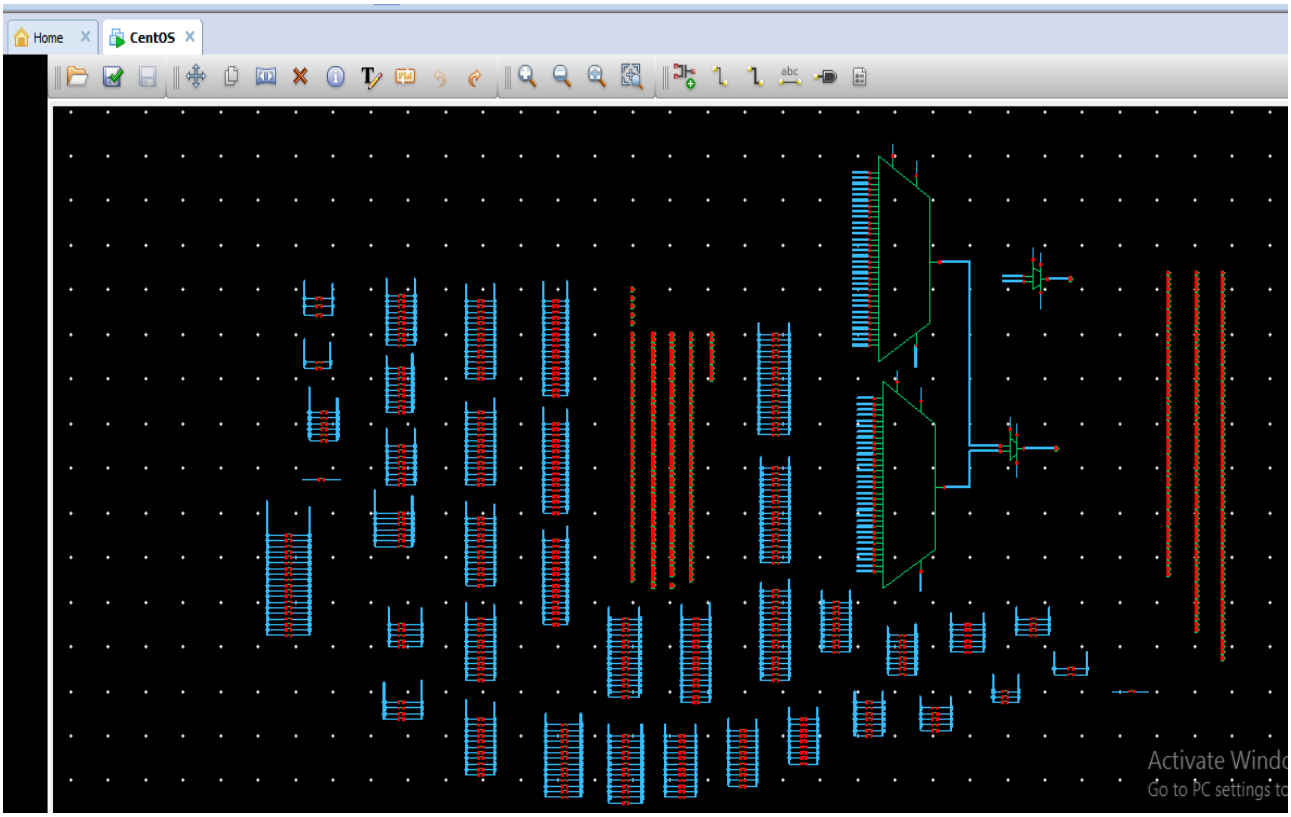


Fig 16: Exponential Shifter Circuit designed in Cadence 90nm gpdk

8.5 SINGLE PRECISION:

Single Precision consists of 32 bit. It is the format proposed by IEEE standard for floating point representation. The inputs of Single Precision are arranged in a format such as given below:

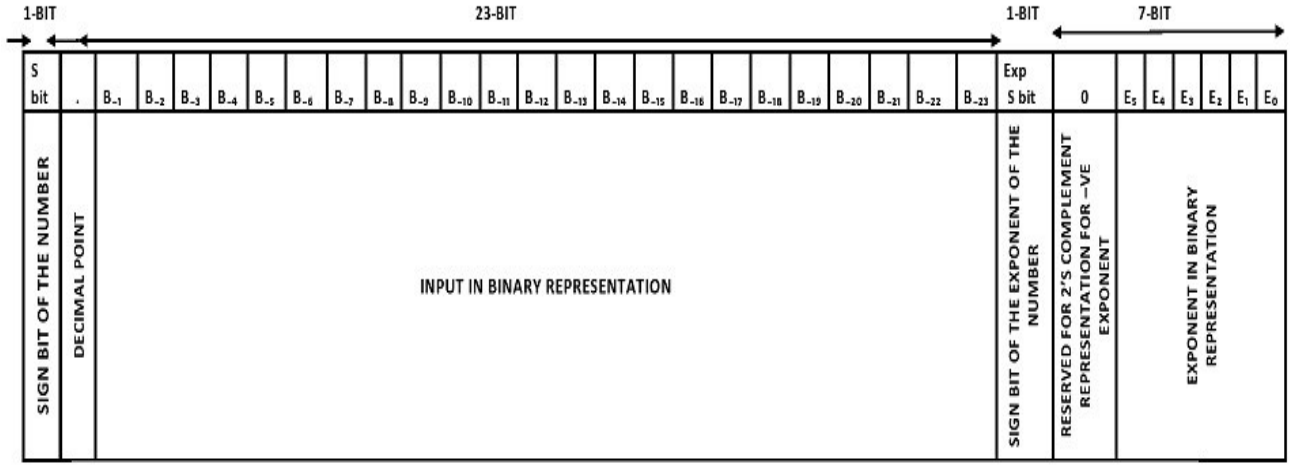


Fig 17: Single precision input floating point structure

The input size is 32 bits, in which two bits are reserved for sign bits of the number and its exponent. The positive or negative number representation is depends on 0 or 1 of the sign bit, respectively. 23 bits are used for binary representation and seven bit exponent represented in binary. In this binary representation the important point is to consider that the 8^{th} bit of the exponent is made default as '0' for 2's complement form. The Normalization architecture for Single precision is same as half precision. The number of bits has to be increase in the implementation process.

8.5.1 Exponential Adder:

The inputs given to the EA block is of 8 bits (including sign bit), it produces the result in 9 bits. Along with that the MSB bit (i.e. 9^{th} bit) is the sign bit of the result. The size of each exponent is 8-bits in which one bit is reserved for sign representation. The output is in sign-magnitude form only and the last bit of EA block depends on the exponent 1 & 2 sign bits and carry bit of adder block i.e., XOR value. Exponential adder was designed and implemented in Cadence- Virtuoso using 90nm gpdk technology.

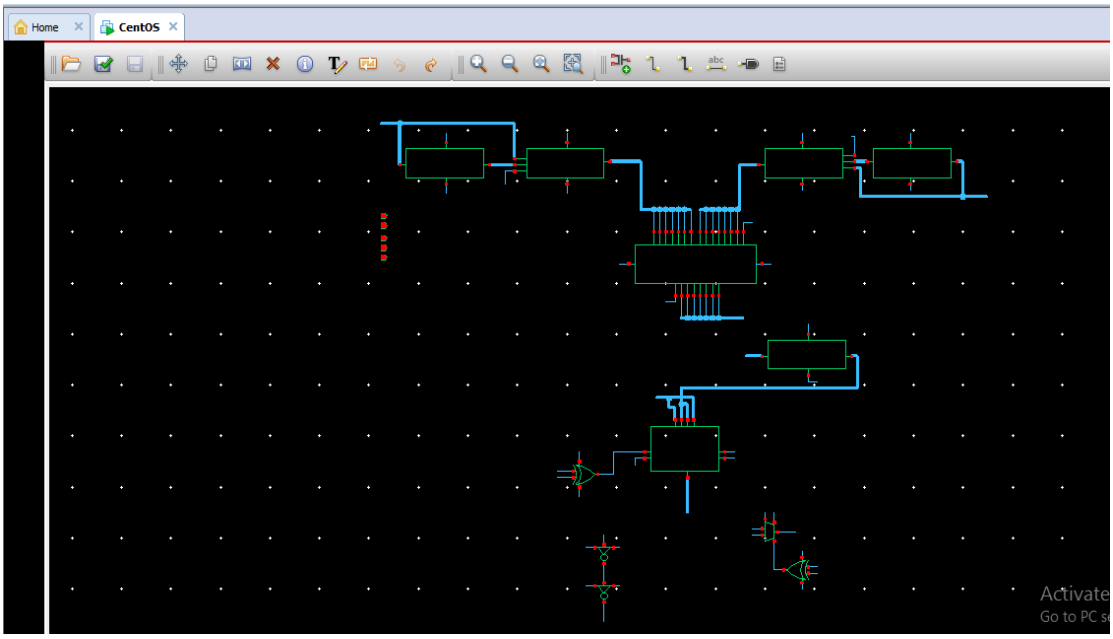


Fig 18: Exponential adder Architecture design

8.5.2 Exponential Comparator Circuit (ECC):

The product of Exponents (EA output i.e., 9 bit) and output of previous cycle exponent (9 bit). Based on sign bits of inputs, it finds the higher input number and difference between two numbers. Comparing bit by bit from MSB to LSB to find the higher number and using 2's complement block to find the difference. Multiplexers are used to compare the inputs.

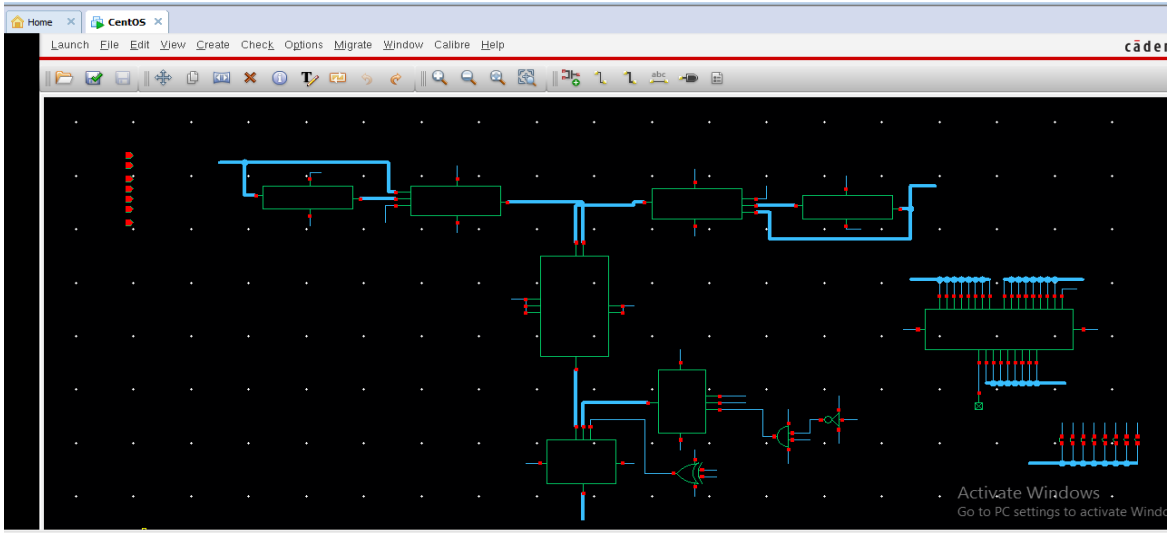


Fig 19: Exponential Comparator Architecture design

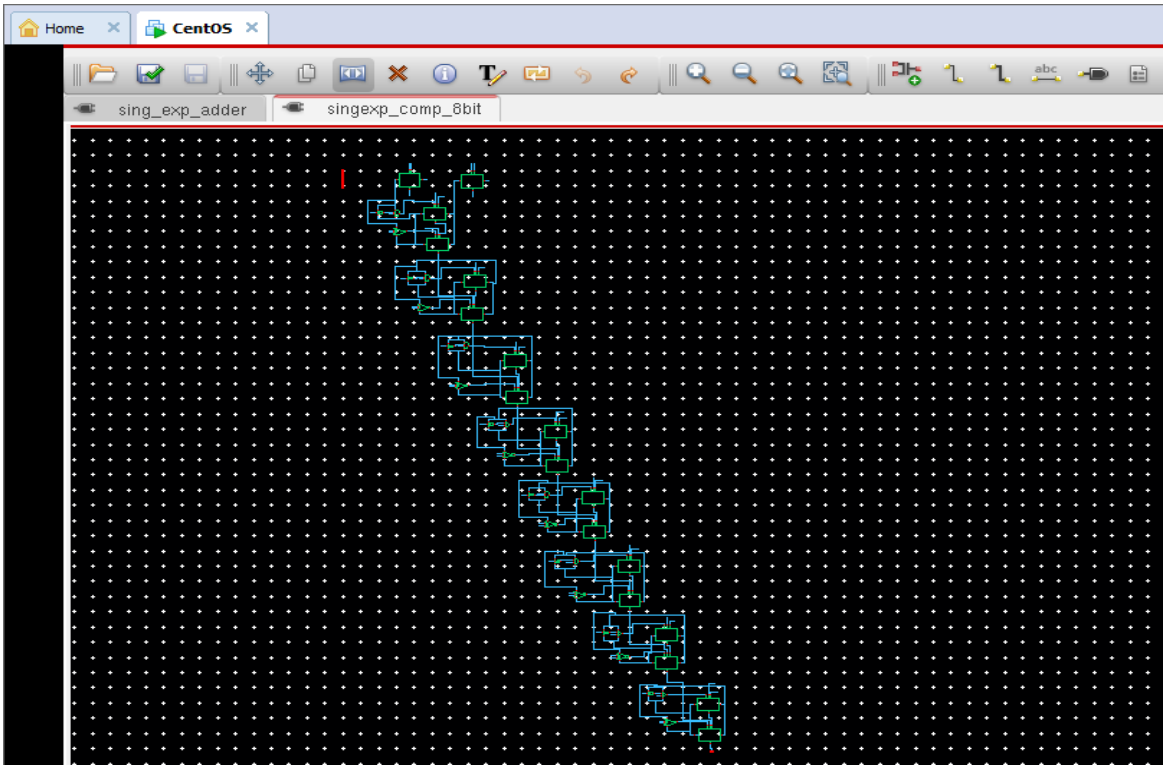


Fig 20: Exponent Comparator of 8-bit.

It compares the bit by bit from MSB to LSB using multiplexer of 9bits.

8.5.3 Exponential Shifter Circuit (ESC):

Single Precision shifter has to be design for shifting of 32 bits using 32 bit multiplexer. The inputs to the shifter are 32 bits of previous output and product of inputs. Based on the ECC output (RES of 8 bits) work as selection lines and input to the shifter. If the RES MSB is 1, then the product of inputs is shifted towards right with equal 8 bit value of ECC. If the RES MSB is 0, then the previous output is shifted to right side. The shifter has to design in such a way that comparing and shifting process will be done simultaneously. Ped-latch pair is to apply at each and every output bit to obtain synchronized output without loss of data.

Simulation Results:

1. Half Precision: It occupies 16 bits in IEEE-754 standard Binary floating point number format, in which one bit for sign, five bits for Exponent and ten bits for Mantissa. The primary blocks of Normalization architecture are Exponential Adder, Exponential Comparator and Exponential Shifter was explained in experimental work. The simulation result is given below:

i. **Exponential Adder:** EA adds the inputs exponents of each 5 bits i.e., including sign bit and it produces the output of 6 bits with sign bit.

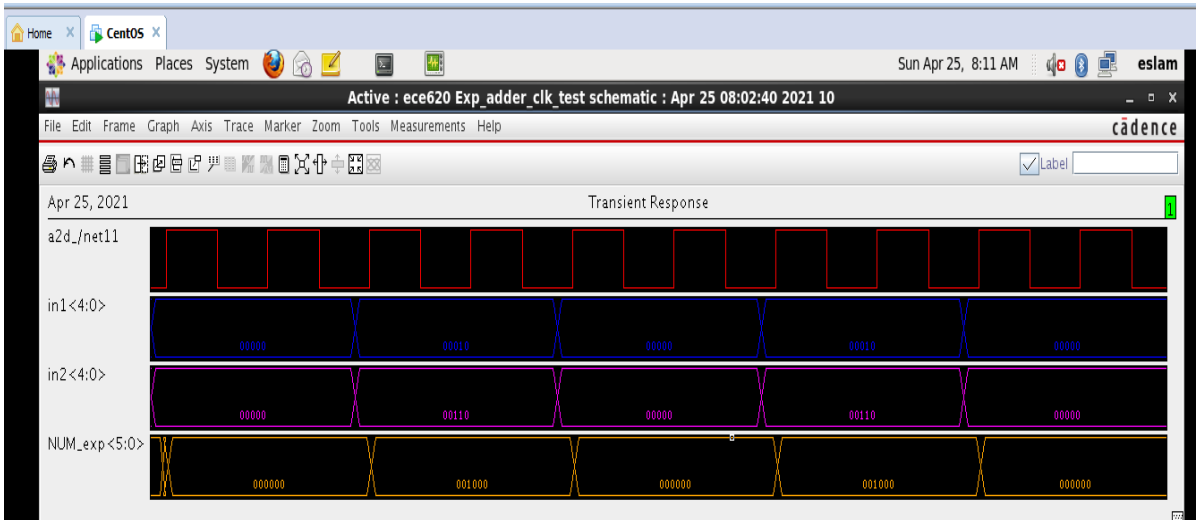


Fig 21: Exponential Adder Output Waveform

ii. **Exponential Comparator Circuit:** The inputs given to Comparator are EA 6-bit output and previous cycle of 6-bit. Comparator compares both the inputs and produces the result of 6 bit i.e., comparing bit by bit of both inputs for finding higher number and by using 2's complement block to find difference.

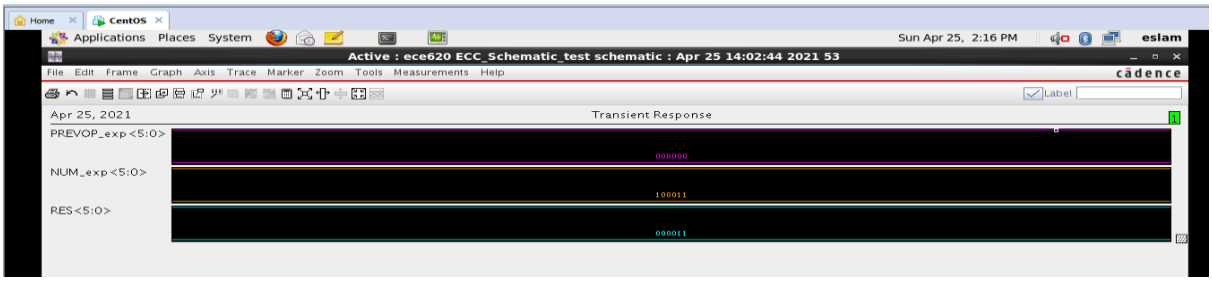


Fig 22: Exponential Comparator Output Waveform

iii. **Exponential Shifter Circuit:** It is responsible to shift the smaller number based on difference value of two exponents. It identifies the smaller number is done based on the ECC output. Shifter shifts the number towards the right based on the 6-bit Comparator output MSB (1 or 0). The simulation is done combining both comparator output and shifter inputs so as to perform the shifting process simultaneously.

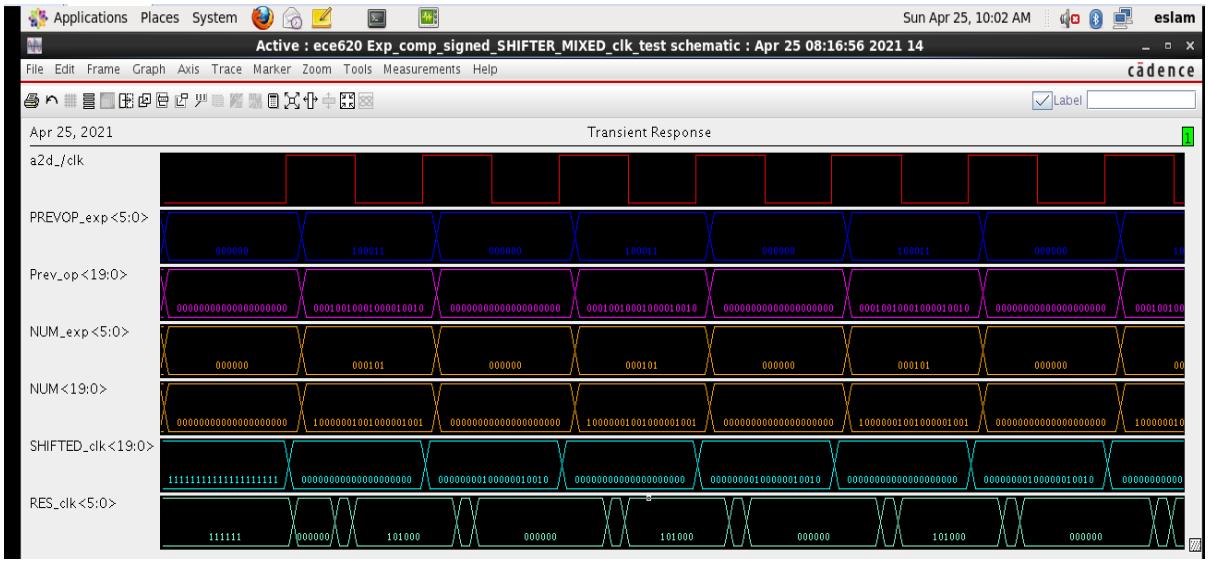


Fig 23: Exponential Shifter Output Waveform

(i) In a system, simulation refers to check and verify the functionality of building blocks of design. A Multiplexer based Normalization of Half Precision is done for IEEE-754 floating point numbers using EA, ECC and ESC circuits. The EA block majorly consists of multiplexer, adder and 2's complement block. In this the EA performs addition process of two input exponents of 5bits with different signs. For negative sign of 4bits input, the 2's complement bock is responsible to represent it as a negative binary number in 2's complement form and adder adds the inputs using full adders and produces a result of 4bit output along with carry. The obtained output is not exactly result. Further it was given to 4 bit 4:1 mux followed by conditions; it represented the result in sign-magnitude form. EA last bit depends on the input sign bits and adder carry bit. The EA produced the 6bit result in the simulated waveform is represented as NUM_exp. For synchronization of EA, PED-latch pair and clock is applied to each output bit to get synchronized EA output.

(ii) The EA output of 6 bits and previous cycle output of 6 bits are given to comparator. Based on sign bit the inputs are represented in 2's complement form. Here the important point is to consider that if the both inputs signs are different then addition has to perform to produce 5-bit output, discarding the carry bit. Otherwise if both signs are same, it finds the difference between and highest input number by comparing both numbers bit by bit from MSB to LSB. If Exponent input is higher than previous cycle exponent, then make the 6^{th} bit as 1. For comparing the inputs, Multiplexers are used. In the simulated output waveform the 6 bit output is RES on comparison of both inputs which is used to shifting process further.

(iii) To shift the 20 bits of half precision based on the ECC output RES. 20 bits of previous output and product of inputs has given as inputs to 32 bit mux. In that 20 bits are shifted for half precision. If the ECC output MSB is 1, then the product of inputs is shifted to right side with equal decimal value of 5-bit ECC output. If MSB is 0, then previous output is shifted to right side. If there is no shift needed for ESC input, also determined by the ECC MSB. Previous output is passed without shifting, if the MSB is 1 and if MSB is 0, the product of inputs is passed without shifting. It is designed such that comparing and shifting process is done simultaneously and to obtain synchronized output, PED-latch pair and clock is applied to each output bit.

2. Single Precision: It occupies 32 bits in IEEE-754 standard in which one bit for sign, eight bits for exponent and 23 bits for Mantissa. The increment of number of bits has done with respect to input structure.

i. Exponential Adder: It adds the two input exponents of each 8bits including sign bit produced the result of 9 bits with sign bit. The obtained result is without synchronization. So ped-latch pair is used at each output bit of EA along with clock to get synchronized output as NUM_exp.

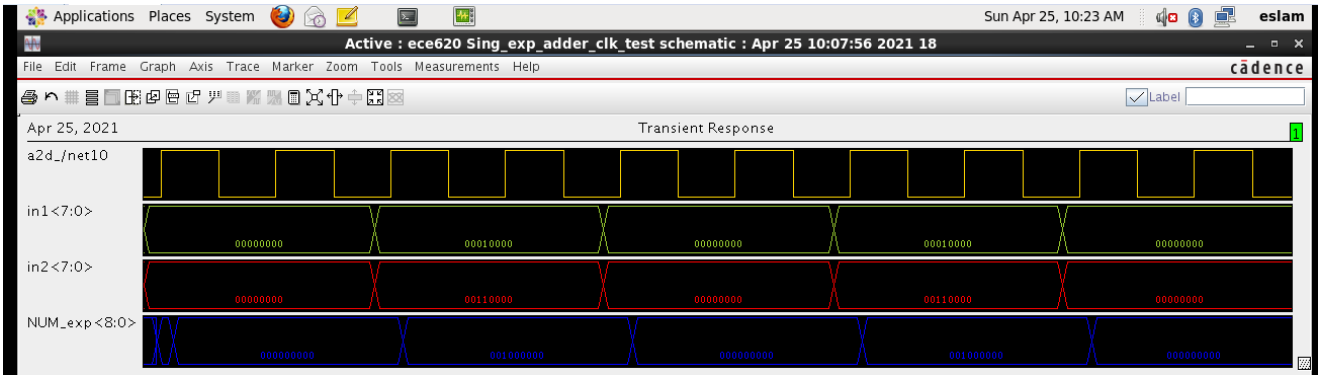


Fig 24: Exponential Adder output waveform

ii. Exponential Comparator: The EA output of 9bits and previous cycle output of 9bits are given as inputs to comparator. It compares the both inputs bit by bit from MSB to LSB to find the higher number and using 2's complement block to find difference.

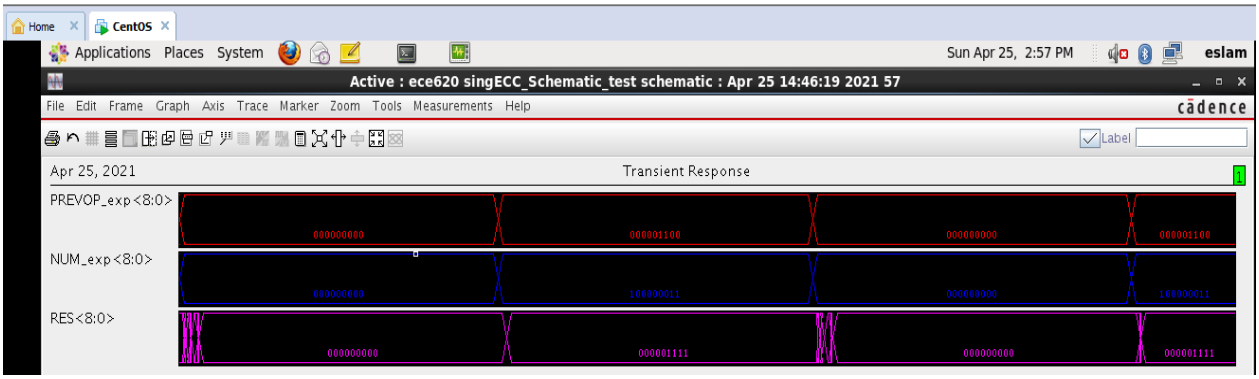


Fig 25: Exponential Comparator output waveform

Explanation of Multiplexer based Normalization Architecture using binary values:

Let us take for an example in detailed to elaborate on proposed architecture.

The input numbers are given as follows:

Input1 = **0**01001011, Input1 exponent = **0**0010

Input2 = **1**01010001, Input2 exponent = **0**0101

The MSB is highlighted for both inputs i.e., (9^{th} bit). In the same way, the MSB of both input exponents is highlighted. For this example, Input1 and Input1 exponent are positive and Input2 & Input2 exponent are positive. Other side, the previous output and its exponent is

positive i.e., 00000000000000000 and its exponent 00000. The process steps for the example are described as follows:

1. The two input exponents of Input1 & Input2 are 00010 and 00110 are given. It produced the result of NUM_exp represented in 6 bits (**0**01000).The addition of 5 bit numbers produces a result in 6 bits including sign bit. The NUM_exp is in sign-magnitude form.

2. NUM_exp and previous cycle output of 6 bits (001000 & 000000) are given to comparator. Comparator compares both the NUM_exp and previous cycle output. Here, NUM_exp is higher number compared to previous cycle output exponent; it produced the output as RES (001000). So sign bit is represented as '0'.

3. The RES output is given to Shifter as input selection line.32bit multiplexer is designed to shift the 20bit numbers. Product of inputs (10000001001000001001) and previous output (00010010001000010010) of 20bits has given as input. The shifter shifts the 20bits in accordance with RES value. Here, the RES value is 001000, by this equivalent decimal value the product of inputs is shifted as (00000000100000010010) which is synchronized output.

Table 2: Power delay of internal blocks of Normalization Architecture

Block	Static power	Average power	Dynamic power	delay	Inference
EA_Adder block (Half Precision)	6.2mw	5.2mw	-1mw	58.51ps	The maximum delay from input clock to output MSB (NUM_exponent).
ESC block along with ECC block (Half Precision)	131.42mw	137.7mw	6.28mw	58.16ps	Delay from clock to output MSB of shifted clock i.e., with same sign bits (either positive or negative) in ECC block to maximum bit of ESC block.
EA_clk block (Single Precision)	14mw	9.437mw	-4.563mw	280.1ps	Delay from clock to output MSB of (NUM_exponent).

10. Summary and Conclusion

Normalization process of two floating point numbers for half precision is taken such that Exponential adder, Exponential Comparator Circuit and Shifter Circuit was designed and implementation is done on Cadence virtuoso technology. Exponential shifter of single precision yet to be implement. Single precision of 32 bits occupies more memory, large range and more accuracy. Half precision occupies half of the memory space and in some situations for floating point numbers it gives greater performance better than 32 bits. Half and Single precision of Normalization process gives the high efficiency and power. Ped-latch pair of pipelining process, the power will be reduced. The normalization of floating point numbers in the IEEE-754 standard has the advantage of large range of values with better accuracy and precision. Power and delay are the most important factors in CMOS circuits. The step wise procedure for normalization architecture of Exponential adder, Exponential Comparator Circuit and Exponential Shifter Circuit was elaborated using different sizes of 2x1/4x1 multiplexers. For design and implementation, the Cadence tool of gpdk 90nm, 45nm technologies can be used. The multiplexer based normalization architecture of floating point representation can be used in all Normalization based applications and also in all higher demand engineering applications of wide range of data with greater accuracy, precision and high speed.

REFERENCES:

[1] E. Hokenek, R. Montoye and P. W. Cook, “Second-Generation RISC Floating Point with Multiply-Add Fused”, *IEEE Journal of Solid-State Circuits*, Vol. 25, pp. 1207-1213, 1990.

[2]A. Beaumont-Smith, M. Liebelt, C. Lim, K. To and W. Marwood, “A digital signal multi-processor for matrix applications", *Proc. 14th IREE Australian Microelectronics Conference*, pp. 245-250, 1997-Sep.

[3] Mohammed Al Ashrafy, Asharf Salem and Wagdy Anis, "An efficient implementation of floating point multiplier", *Electronics Communications and Photonics Conference* (SIECPC*)*, 2012.

[4] Jean-Pierre Deschamps, Gery Jean, Antoine Bioul and Gustavo D. Sutter, "Floating Point Unit" in *Synthesis of Arithmetic Circuit, Hoboken, New Jersey: A john Wiley & Sons*, inc., publication, pp. 513-548, 2006.

[5] Meenu, S. Ravi and Ajit Saraf, "Analysis and study of different multipliers to design floating point MAC units for digital signal processing applications", *International Journal of Research inAdvent Technology (IJRAT),* vol. 2, no. 3, pp. 264-267, March 2014, ISSN 2321-9637.

[6] G. Dimitrakopoulos, K. Galanopoulos, C. Mavrokefalidis and D. Nikolos, "Low-power leading-zero counting and anticipation logic for high-speed floating point units", *Very Large Scale Integration (VLSI) Systems IEEE Transactions on*, vol. 16, no. 7, pp. 837-850, 2008.

[7] A. Malik, D. Chen, Y. Choi, M. H. Lee and S. B. Ko, "Design tradeoff analysis of floating-point adders in FPGAs", *Electrical and Computer Engineering Canadian Journal*, vol. 33, no. 3/4, pp. 169-175, 2008.

[8] Kunapareddy S. Nagendra and D. Suresh, "Improved fused floating point add-subtract unit", *International Journal of Research in Computer and Communication Technology*, vol. 2, no. 9, Sep. 2013.

[9] J. M. Yohe, "Rounding in floating point arithmetic", *IEEE Trans. Computers*., vol. C-22, pp. 577-586, June 1973.

[10] P. R. Gray and R. G. Meyer, “Analysis and Design of Analog Integrated Circuits”, 1984.

[11] V. Oklobdzija, "Comments on leading-zero anticipatory logic for high-speed floating point addition", *IEEE. Solid State Circuits*, pp. 292-293, Feb. 1997.

[12] H. H. Saleh and E. E. Swartzlander, Jr., “A Floating-point Fused Dot-Product Unit”, *IEEE International Conference on Computer Design*, ICCD, pp. 427-431, 2008.

[13] Kyung-Nam Hanand Sang-Wook Han,"A new floating-point normalization scheme by bit parallel operation of leading one position value", *IEEE Asia-Pacific Conference on ASIC*, 8-8 Aug. 2002

[14]Jae Hong Min, Earl E. Swartzlander," Fused floating- point Magnitude Unit", *IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 4-7 Aug. 2013.

[15]Jongwook Sohn ; Earl E. Swartzlander," A Fused Floating Point Four-Term Dot Product Unit", *IEEE Transactions on Circuits and Systems*,vol.63,24 February 2016.

[16] P M Drusya ; Vinodkumar Jacob," Area efficient fused floating point three term adder", *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*,3-5 March 2016.

[17]Preeti Khobragade and Prachi Palsodkar," Floating point unit using error correction scheme and modified anticipator", *Online International Conference on Green Engineering and Technologies (IC-GET)*,19-19 Nov. 2016.

[18] P.M. Seidel," On-line IEEE floating point multiplication and division for reduced power dissipation", *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, vol.1,7-10 Nov. 2004.

[19] Jongwook Sohn ; Earl E. Swartzlander," Improved Architectures for a Floating point Fused Dot Product Unit",*IEEE21st Symposium on Computer Arithmetic*,7-10 April 2013.

[20]Jongwook Sohn ; Earl E. Swartzlander," A Fused Floating point Three-Term Adder", *IEEE Transactions on Circuits and Systems*,vol.61,22 July 2014.

[21]Anatoly I. Grushin ; Maxim L. Remizov, "Fast Result Normalization in FP Adder", *IEEE25th Convention of Electrical and Electronics Engineers in Israel*,3-5Dec. 2008.

[22] Fang Jianping ; HaoYue ; Che DeLiang," A novel LOP to improve the normalization of the FP adder in DSPs",*7th International Conference on Solid-State and Integrated Circuits Technology,*vol.3,18-21 Oct. 2004.

[23]H. Suzuki ; H. Morinaka ; H. Makino ; Y. Nakase ; K. Mashiko ; T. Sumi," Leading-zero anticipatory logic for high-speed Floating point addition", *IEEE Journal of Solid-StateCircuits*,vol.31,Aug 1996.

[24]Giorgos Dimitrakopoulos and Kostas Galanopoulos "Low-Power Leading-Zero Counting and Anticipation Logic for High-Speed Floating Point units", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*,vol.16, 27 June 2008.

[25]Richard J. Zaccone ; Jesse L. Barlow," Eliminating the Normalization Problem in Digit on Line Arithmetic", *IEEE Transaction on Computers*,vol.C-36,Jan. 1987.

[26]Nisha Singh; R Dhanabal," Design of Single Precision Floating point Arithmetic Logic Unit",*4th International Conference on Electrical Energy Systems (ICEES)*,7-9 Feb. 2018.

[27] H. Yamada ; T. Hotta ; T. Nishiyama ; F. Murabayashi ; T. Yamauchi ; H. Sawamoto," A 13.3ns double-precision floating point ALU and multiplier",*95 International Conference on Computer Design. VLSI in Computers and Processors,*2-4 Oct. 1995.

[28] E. Antelo ; M. Boo ; J.D. Bruguera ; E.L. Zapata," A novel design of a two operand Normalization Circuit", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*,vol.6,March 1998.

[29]Peter Kornerup," Correcting the Normalization Shift of Redundant Binary Representations", *IEEE Transactions on Computers*, vol.58, 06 March 2009.

[30]T. Lang ; J.D. Bruguera," Floating point multiply-add-fused with reduced latency", *IEEE Transaction on Computers*,vol.53,21 June 2004.

[31] J.D. Bruguera ; T. Lang," Floating point fused multiply-add: reduced latency for floating point addition",*17th IEEE Symposium on Computer Arithmetic (ARITH'05)*,27-29 June 2005.

[32]A. Beaumont-Smith ; N. Burgess ; S. Lefrere ; C.C. Lim," Reduced latency IEEE floating point standard adder architectures",*14th IEEE Symposium on Computer Arithmetic (Cat. No.99CB36336)*, 14-16 April 1999.

[33]Suresh Srinivasan ; Ketan Bhudiya ; Rajaraman Ramanarayanan ; P. Sahit Babu ; Tiju Jacob ; Sanu K. Mathew ; Ram Krishnamurthy ; Vasantha Errgauntla," Split-Path Fused floating point Multiply Accumulate (FPMAC)",*IEEE21st Symposium on Computer Arithmetic*,7-10 April 2013.

[34]J. Eldon ; C. Robertson," A Floating point format for signal processing", *IEEE International Conference on Acoustics, Speech, and Signal Processing*,vol.7, 3-5 May 1982.

[35] Weiqiang Liu ; Linbin Chen ; Chenghua Wang ; Máire O'Neill ,"Design and Analysis of Inexact Floating point Adders", *IEEE Transactions on Computers*,vol.65, 27 March 2015.

[36]D. W. Sweeney," An analysis of floating point addition", *IEEE Transactions on Circuits and Systems*, vol.4, 1965.

[37] Andre Guntoro ; Manfred Glesner," A flexible Floating point wavelet transform and wavelet packet processor", *Design, Automation & Test in Europe Conference & Exhibition*,20-24 April 2009.