http://www.jctjournals.com

ISSN(ONLINE):2278 - 3814

Volume-12, Issue-07, July 2023

JOURNAL OF COMPUTING TECHNOLOGIES (JCT)

International Journal

Page Number: 01-07

Designing of 32 bit Floating Point Carry Look Ahead Adder Using VHDL

[1]*Ajay Kumar*, [2]*Prof. Nishi Pandey*, [3]*Prof. Abhishek Agwekar*

[1]*M.Tech Scholar*, [2]*Assistant Professor*, [3]*Head of department*

[1,2,3] *TIEIT, Bhopal,*

[1]ajaylion33@gmail.com [2]nishi.pandey@trubainstitute.ac.in, [3]abhiagwekar@gmail.com

***Abstract*— Floating-point arithmetic is by far the most widely used method for approximating real-time operations when performing arithmetic calculations on modern computers. The advantage of floating point notation is that it can support more values than the representation of points and numbers. Addition/subtraction, multiplication and division are basic arithmetic operations on these numbers. Here floating point addition/subtraction is the hardest. This paper uses 32-bit floating point adder with high performance, beautiful space and high performance according to IEEE 754 standard. Use VHDL. The proposed architecture is implemented in the Xilinx ISE simulator. The results of the proposed design were compared with existing architectures and a reduction in space and latency was observed. Also, the project can be measured in terms of area, velocity and power using other types of acceleration.**

***Keywords*— *CLA, FPGA, DSP, Signal and* Xilinx ISE simulator.**

I. INTRODUCTION

Multiplication is the most important arithmetic operation in many applications. As speed is always constraint in the multiplication operation, hence increasing the speed is very important. The speed of the multiplier determines efficiency of the system. All the signal and data processing operations involve multiplication. In any system design, the three main constraints which determine the performance of the system are speed, area and power requirement. A Field Program Gate Array (FPGA) offers an integrated digital circuit plate form to design an integrated circuit using a general purpose computer (GPC) and application specific integrated circuit (ASIC), with the addition of modifying the structure of the design as per user requirement. The major limiting factor is a programmable logic resource on the FPGA board, which were used to configure different integers and bit-level tasks. However, in recent few years, this method offers a wide scientific application to monitor and control on different platform such as Internet of Things, Artificial Intelligence and Machine Automation.

Multiplication of the floating point numbers is a discriminating necessity for DSP application including huge dynamic range. However the design of the floating point multiplier required a number of interconnected (cascaded) processing blocks such as adders and shifters. It includes designing of such a system through synchronous logic, it has many drawbacks such that the simultaneous transition of all clock signals may cause generate the noise, the latency and throughput of the complete system are equal to the slowest element. Also it consumes higher power, because of unnecessary transistor transitions and non-modular-design, complex clock distribution network, combines to form major problems, which can be overcome by using the asynchronous approach called self-timed circuits. This technique observes the state of operation of connected blocks, thus eliminating the need for clock synchronization. The self-timed circuits have shown good adaptability to environmental and operational conditions while removing all the drawbacks of synchronous circuits. Hence we have shown that this technique can also be successfully applied to floating multiplier circuits.

II. LITERATURE REVIEW

Various researches have been carried out in order to design adders and a proposed design of 32 bit multiplier using carry look ahead. Carry Look ahead Adder (CLA) is one of the fastest adder structures that is widely used in the processing circuits.There are two types of circuits, named as synchronous andan asynchronous. The synchronous circuits have a clock signal to synchronize the operations of subsystems, while an asynchronous circuit does not have a clock signal, but its operating states, these states may or may not be output of the cascaded sub-system.

P. Balasubramanian, et.al., "Low power self-timed carry lookahead adders," in Circuits and Systems (MWSCAS),

2013. In this paper authors focused on semi-custom design (direct synthesis) of self-timed CLA adders which are physically implemented using standard cells, with an eye on timing optimization. In this context, authos deals with the design of self-timed CLA adders based on the notion of section-carry, where intra-section carries are allowed to ripple within an adder group, while inter-section carries are generated via lookahead. Moreover authors discuss on efficient implementations of 16-bit full adders on FPGA devices using specialized carry-logic. In this paper author focus on the implementation of 16-bit full adder based on Very High Speed Integrated Circuit (VHSIC) Hardware Description Language. And concluded with the design of Hardware Resources mapped on SPARTAN-3 FPGA, implement, simulate and synthesized using VHDL[1].

Preethi Sudha, et.al." Design Of High Performance IEEE-754 Single Precision (32 bit) Floating Point Adder Using VHDL" In this paper Floating Point arithmetic is by far the most used way of approximating real number arithmetic for performing numerical calculations on modern computers. The advantage of floating-point representation over fixed-point and integer representation is that it can support a much wider range of values. Addition/subtaraction, Multiplication and division are the common arithmetic operations in these computations. Among them Floating point Addition/Subtraction is the most complex one. This paper implements an efficient 32bit floating point adder according to IEEE 754 standard with optimal chip area and high performance using VHDL .The proposed architecture is implemented on Xilinx ISE Simulator. Results of proposed architecture are compared with the existed architecture and have observed reduction in area and delay . Further, this project can be extendable by using any other type of faster adder in terms of area, speed and power [2].

F.-C. Cheng, et.al. "Self-Timed Carry-Look ahead Adders," IEEE TRANSACTIONS ON COMPUTERS, vol. 49, no. 7, pp. 659-671, JULY 2000. A Subsystems, in asynchronous circuits, usually need a subsequent start and completion mechanisms that is to be used to synchronize with one another subsystem. The major advantage an asynchronous circuits is it operate at average rate while synchronous circuits are operate at the worst rates[4].

Behnam Amelifard, et.al. "Closing the Gap between Carry Select Adder and Ripple Carry Adder: A New Class of Low-power High-performance Adders," in Quality of Electronic Design, ISQED 2005. Sixth International Symposium on21-23 March, 2005. In this paper authors discussed on the idea of sharing two adders used in the Carry Select Adder (CSA), aand presented a new design of a low-power high performance adder. Authors compare the speed of a new adder against a Ripple Carry Adder (RCA), and found that new adder is faster than Ripple Carry Adder but slower than a CSA. While in terms of its area and power dissipation new adder occiped a smaller area and lower power than those of a CSA [5].

III. PROBLEM FORMULATION

Floating Point (FP) addition, subtraction and multiplication are widely used in large set of scientific and signal processing computation. As we have already seen that multiplier is one of the key components which perform a complex function in the system due to which delay occurs. This lag causes hindrance in other operations too. Further, while we see that major drawback of a conventional multiplier is its speed; the main reason behind it is the process of multiplication. Suppose if we consider shift and add multiplier we can see that the number of partial products generated are quite high. Due to which more number operations have to be performed. This is one of the major problems which need to be overcome. Secondly, we have observed that multiplier can be made using different types of adder. Speed and efficiency of a multiplier largely depends on which adder is used. Therefore, we had to make a comparative study using different adders. Power and area requirement is of other adders is also large. Hence, we had to select such type of adder which provides optimization of the basic requirements.

Structure of Self-Timed Circuit (Synchronous) We have a synchronous and self-timed circuit of a unit is shown in Figure 4-1 and 4-2 respectively. The control is based on a request/acknowledges protocol.

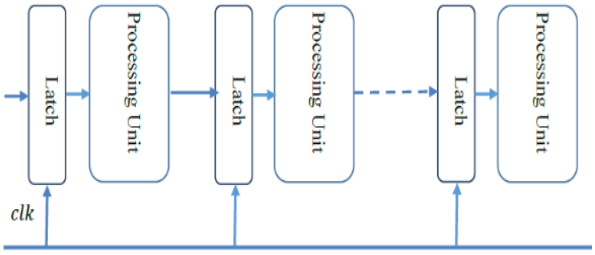

Fig. 1 Block diagram of synchronous cascaded system

In a synchronous cascaded system, processing unit is a pipeline with intermediate latches and get activated through a single clock. That clock helps to create coordination and synchronization among processing units. While in an asynchronous, internally the signals Req, ok, t, $done$, and en are used for intercommunication between control and processing units. As shown in figure 3-5 the input of the second block is raised by the first block, if second block is free then the control unit of the second block registers the data by risingen of the latch, and then the Ack signal is raised to ensure the first block that it has accepted the data for processing. The control block of second block raises the $start$ signal for its processing unit; when the unit completes the given task (after some amount of time) the $done$ signal from processing unit goes high.
In the same way processing progresses till last block reaches. The important and challenging task for the self-timed circuit is to generate the done signal. A systematic way of detecting the end of the computationconsists of using a redundant encoding of the binary signals for different state or incident is as:

Each signal s is represented by a pairs($s1$,$s0$). According to the definition for normal operating conditions $if s1=s, hens0=not(s) for resets1=0, s0=0$.
We can assume different value of signal as $ns1,2,...,sn$.
Each signal si issubstituted by a pair ($1i$,$s0i$). Then the done flag is computed as in eq. 3.

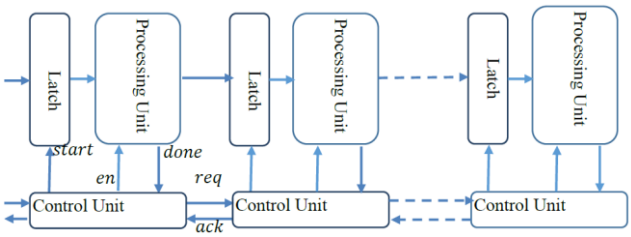

Figure 2 Block diagram of asynchronous (Self-Timed) cascaded system

3.2 Structure of Self-Timed Carry Look Ahead Adder

Self-Timed Carry Look Ahead Adders can be implemented by many ways but in our work we have used the dual-rail signaling [29] in input bits, sum bits, and carry bits, and by using one-hot code in the internal signals. The adder is designed using two basic modules: C and D, connected in a tree-like structure as shown in Figure 3-4. The equations of the C-module are defined as follows:

Equation 2 $$CarryKill = k_i = A_i^0 B_i^0$$

Equation 3 $$\text{Carry Generate} = g_i = A_i^1 B_i^1$$

Equation 4 $$\text{Carry Propagate} = p_i = A_i^0 B_i^1 + A_i^1 B_i^0$$

Equation 5 $$S_i^0 = A_i^0 B_i^0 C_i^0 + A_i^1 B_i^1 C_i^0 + A_i^0 B_i^1 C_i^1 + A_i^1 B_i^0 C_i^1$$

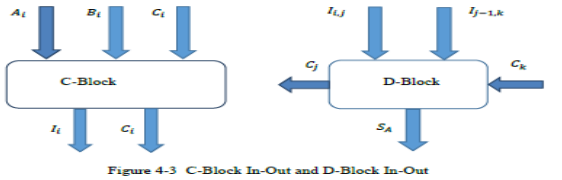

Figure 4-3 C-Block In-Out and D-Block In-Out

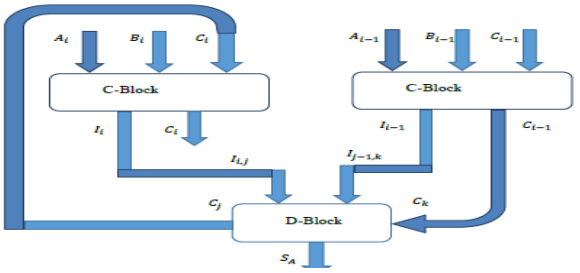

Fig. 3 Tree-Structured Adder using C-Blocks and D-Blocks

IV. SIMULATION AND RESULTS

The VHDL simulation of the proposed multiplier is performed by using Xilinx ISE 14.4,and the simulation waveforms with other important findings are presented in this section.

Fig. 4 RTL of the Proposed Architecture

Fig. 4 represent the Register Transfer Logic (RTL) view of our proposed architecture. It shows combine a view of all units of carry look ahead adder, Exponent adder and shifter and significant or mantissa multiplier.

4.2 RTL of the Floating Point Multiplier

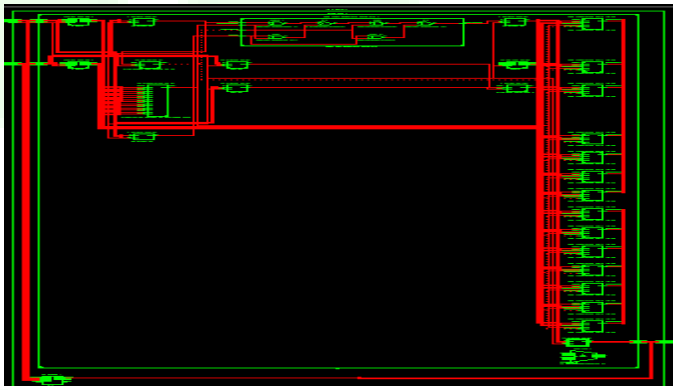

Fig. 5 RTL view Floating Point Multiplier

Fig. 5 represent the RTL view of our proposed Floating Point of multiplier. It illustrate complex connectivity between of individual sub-units for 32-bit configuration. It consist of multiple C-block and D-block unit as per design,

4 .3 RTL of the Exponent and CLA Adder.

Figure 5-3 represent the RTL view of our proposed Exponent and Carry look-ahead Adder. It illustrate the input bit configuration with output bit configuration named as A0 and A1. Here, A0 is used for Exponent adding and shifting with respect to bias bit requirement while A1 is used for Mantissa or Significant Multiplier and result as output.

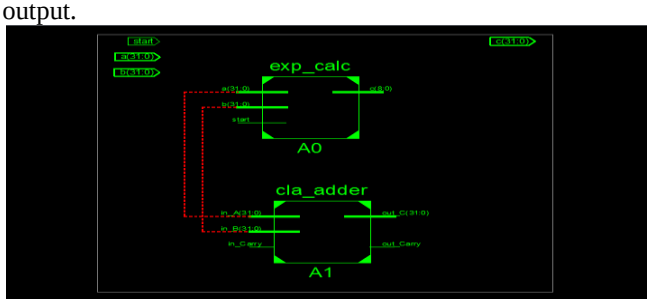

Fig. 6 RTL of Exponent and CLA adder

ISSN(ONLINE):2278 - 3814

4.4 Multiplier Internal View

Figure 5-4 represent the RTL internal view for multiplier. It illustrates a complex network of connection between each unit. As we are working for 32 Bit operation, it creates a huge connection within each sub-units, and furthermore, for 23 bits of mantissa or significant multiplier, individual groups of unit configured to form a network.

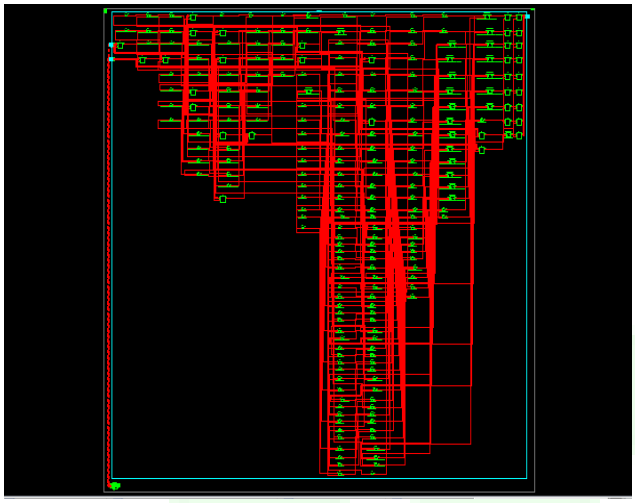

Figure 7 RTL Multiplier -Internal view

4.5 Verification Test-Scenarios with different operand

For verification of results we have performed multiple test with different combination of input. The multiple tests are executed on our simulated proposed architecture, as mentioned below.

5.5.1 Evaluation of Multiplication

x=c1900000=−18.000000

y=41180000=9.500000 z=c32b0000 =−171.000000

Figure 8 Time Analysis for Multiplication

5.5.2 Carry Adder

x=d1900000=−77309411328.000000

y=42131000=36.765625 z=d4257200 =−2842328825856.000000

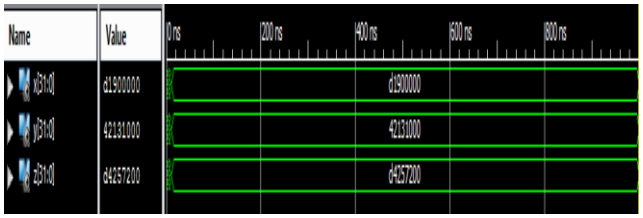

Figure 9 Time Analysis for Carry Adder

5.5.3 Multiplier

x=a3450000

y =111c2000

z=00000000 =0.000000

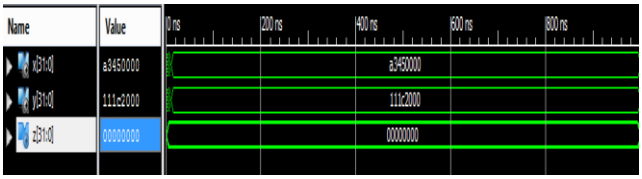

Figure 10 Time Analysis for Multiplier

5.5.4 Floating Point Multiplier

x=72431020=38636205340162287549935643525 12.000000

y =411d2002=9.820314

z=73ef728e

=37941967598990390600068874895360.000000

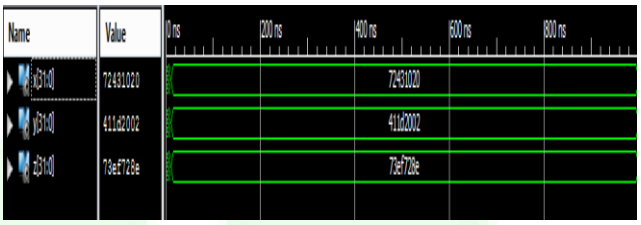

Figure 11 Time Analysis for 32 Bit Floating Point Adder

x=42431020=48.765747 36

y=611d2002=1811528265957229199 36.000000

z=63ef728e =8834052737567199068160.000000

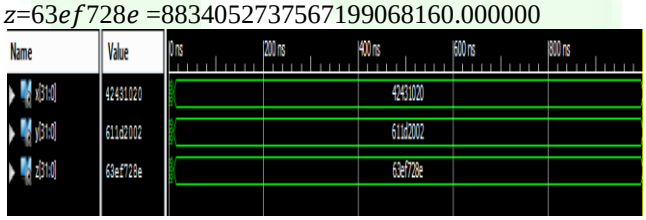

Figure 12 Time Analysis for Floating Point Adder

On the base of the resource utilization numbers of the previous chapter we can estimate how much floating-point operations can be implemented in an FPGA design. For example, a 60 % utilization of a Xilinx 14.4 by pure arithmetic units with 32 bit significands allows for a maximum of unsigned adders or multiple . If all the operators of the target application fit into the FPGA then a direct pipelined approach is the fastest implementation. For formulas with floating-point operations this design style is appropriate.

x=c2431829=−48.773594

y=c1169271=−9.410752

z=43e57f84 =458.996216

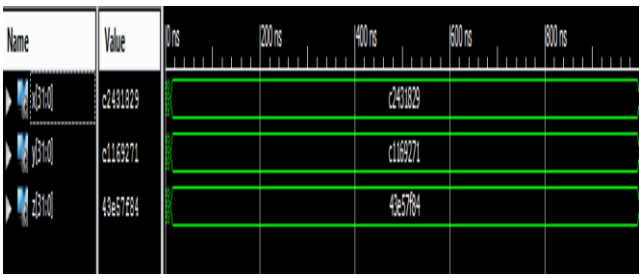

Figure 13 Time Analysis for Sign Bit Floating Point Adder

ISSN(ONLINE):2278 - 3814

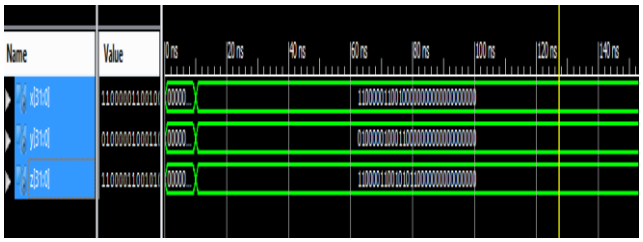

Figure 14 Time Analysis for 32 bit Floating Point Adder

In all the test scenarios, we give input to “x” and “y” and get output at “z”, which match with the desired scenario resultant output.

Simulation results for 32 bit carry look ahead adder are shown below. Here two numbers A and B are taken third is input carry .These two numbers are 32 bit wide. 37 Addition operations is performed on these two numbers which generate the result as sum and carry out.

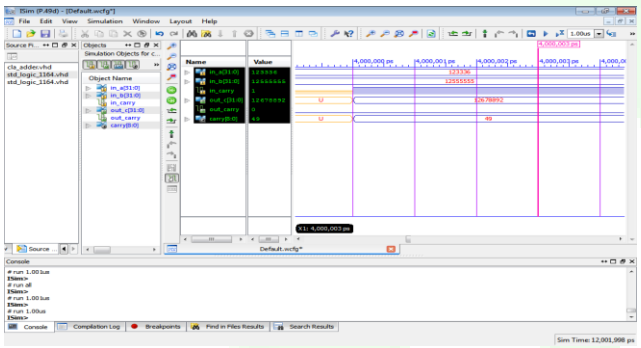

Figure 15 Carry Look Adder

Carry adder

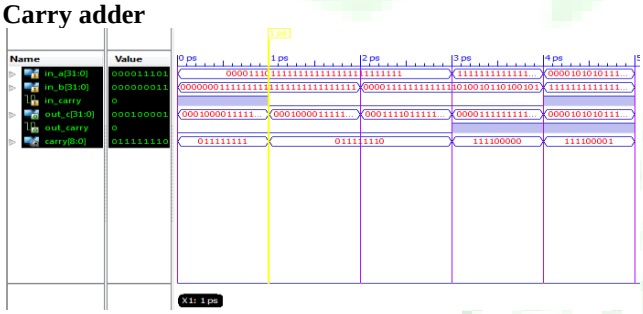

Figure 16 Binary Carry Look Adder Simulation

Other signals like carry generation carry propagation and next carry will be generated. Sum is the addition of three numbers that is A, B and Ci _sum is the addition of two number.

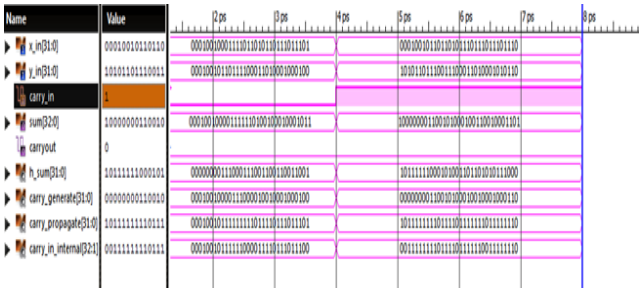

Figure 17 Output simulations of 32 bit carry look ahead adder

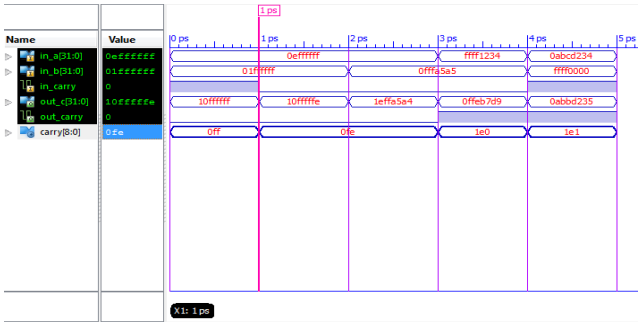

Figure 18 Carry Look Adder in Hexa Simulation

Test Bench b_cla_adder shows the result for the unsigned and signed adder operation

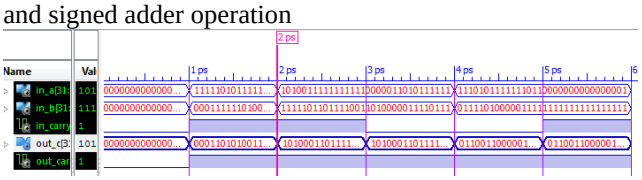

Figure 19 Binary Test Bench Adder

In our test plan successfully produced the expected output frequency testing of the first 32 bit carry adderTb_expo shows the simulation result of 32 bit carry adder

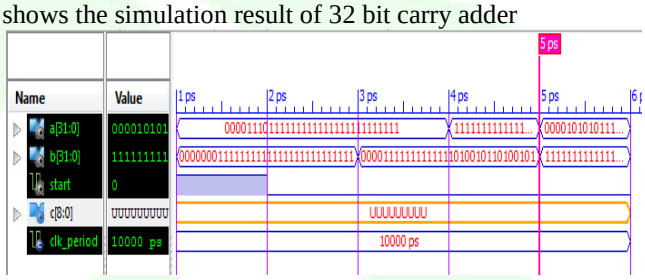

Fig. 20 Exponantial Carry Look Adder

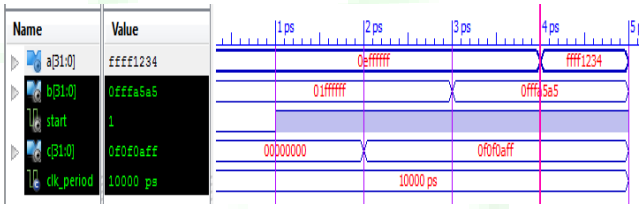

Fig. 21 TB Behave

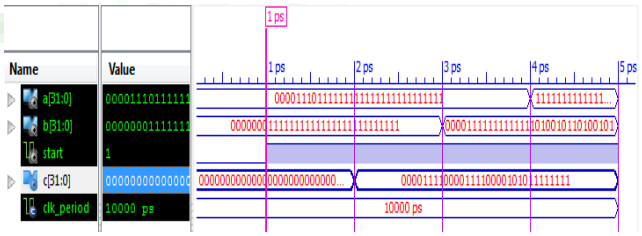

Fig. 22 Floting 32 bit Adder

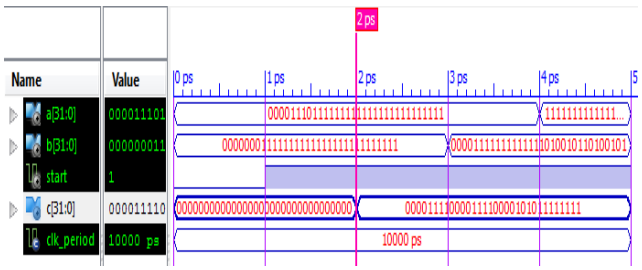

Figure 23 Binary Exponantial Adder.

V. CONCLUSION AND FUTURE WORK

Multiplication is the most important arithmetic operation in many applications . As speed is always constraint in the multiplication operation, hence increasing the speed is very important. The speed of the multiplier determines efficiency of the system. All the signal and data processing operations involve multiplication. In any system design, the three main constraints which determine the performance of the system are speed, area and power requirement. We have done designing of carry look ahead adder through Xillinx EDA tool with a comprehensive simulation testing with all aspects. We have a comparative different reports for time delay and power dissipation analysis with configuration report. As shown in table 1, 2 and 3, multiplier with self-timed CLA shows much better results than with synchronous CLA. The Simulation result shows that multiplier with self-timed CLA takes less time to generate final product than with synchronous CLA because of worst case delay. Similarly, result shows about 20% enhancement in power consumption in case of self-timed CLAbased multiplier. Although our work presents architecture for self-timed 32 bit floating point multiplier in future, it can be extended for 64 bit multiplication.

REFERENCES

[1] I. Koren, Computer Arithmetic Algorithms, New Jersey: Prentice-Hall Inc., Englewood Cliffs, 07632,, 1993.

[2] J.-C. Lo, "A fast binary adder with conditional carry generation," IEEE Trans. Computer, vol. 46, no. 2, p. 248.253, Feb. 1997.

[3] P. Balasubramanian, D. Dhivyaa, J. Jayakirthika, P. Kaviyarasi and K. Prasad, "Low power self-timed carry lookahead adders," in Circuits and Systems (MWSCAS), 2013 IEEE 56th International Midwest Symposium , Columbus, OH, 4-7 Aug. 2013.

[4] F.-C. Cheng, S. H. Unger and M. Theobald, "Self-Timed Carry-Lookahead Adders," IEEE TRANSACTIONS ON COMPUTERS, vol. 49, no. 7, pp. 659-671, JULY 2000.

[5] BehnamAmelifard, FarzanFallah and MassoudPedram, "Closing the Gap between Carry Select Adder and Ripple Carry Adder: A New Class of Low-power High-performance Adders," in Quality of Electronic Design, ISQED 2005. Sixth International Symposium on21-23 March, 2005.

[6] Y. Li and W. Chu, "Implementation of Single Precision Floating Point Square Root on FPGAs," in Proc. Of IEEE Symposium on FPGAs for Custom Computing Machines IEEE Computer Society Press, 1997, pp. 226- 232..

[7] B. Fagin and C. Renard, "Field programmable gate arrays and floating point arithmetic," IEEE Transactions on VLSI, vol. 2, no. 3, pp. 365-367, 1994.

[8] A. Jaenicke and W. Luk, "Parameterized Floating-Point Arithmetic on FPGAs," in Proc. of IEEE ICASSP, 2001 vol.2, pp. 897-900.

[9] Michael L. Overton "Floating Point Representation". [Online]. Available: http://homepage.cs.uiowa.edu/~atkinson/m170.dir/overton.pdf..

[10] K. Ramu and B. S. Rao, "Implementation of Area Efficient 16bit Adder in SPARTAN-3 FPGA," International Journal of Engineering Science and Innovative Technology (IJESIT), vol. 2, no. 2, March 2013.

[11] L. Gerlach, G. Payá-Vayá and H. Blume, "Efficient Emulation of Floating-Point Arithmetic on Fixed-Point SIMD Processors," in 2016 IEEE International Workshop on Signal Processing Systems (SiPS), 26-28 Oct. 2016.

[12] Mnaka, H. Akaike, A. Fujimaki, Y. Yamanashi, N. Yoshikawa, S. Nagasawa, K. Takagi and N. Takagi, "100-GHz single-flux-quantum bit-serial adder based on 10-kA/cm2 niobium process," IEEE Trans. Appl.Supercond., vol. 21, no. 3, p. 792–796, Jun. 2011.

[13] A. F. Kirichenko and O. A. M. T. Mukhanov, " Implementation of novel "pushforward" RSFQ Carry-Save Serial Adders," IEEE Trans. Appl. Supercond., vol. 5, no. 2, p. 3010–3013, Jun. 1995.

[14] A. Y. Kidiyarova-Shevchenk, K. Y. Platov, E. M. Tolkacheva and I. A. Kataeva, "RSFQ asynchronous serial multiplier and spreading codes generator for multiuser detector," IEEE Trans. Appl. Supercond., vol. 13, no. 2, p. 429–432, Jun. 2003.

[15] S. V. Polonsky and A. V. Rylyakov, "RSFQ arithmetic blocks for DSP applications," IEEE Trans. Appl. Supercond., vol. 5, no. 2, p. 2823–2826, Jun. 1995.

[16] H. Park, Y. Yamanashi, N. Yoshikawa, M. Tanaka and A. Fujimaki, "Design of fast digit-serial adders using SFQ logic circuits," IEICE Electronics Express, vol. 6, no. 19, pp. 1408-1413, 2009.

[17] PongyupinpanichSurapong and F. A. Samman, "Floating-Point Division Operator based onCORDIC Algorithm," ECTI Transactions on Computer and Information Technology, vol. 7, no. 1, May 2013.

[18] OmidSarbishei and KatarzynaRadecka, "On the Fixed-Point Accuracy Analysis and Optimization of FFT Units with CORDIC Multipliers",," in 20th IEEE Symposium on Computer Arithmetic., 2011.

[19] T. Vladimirova, D. Eamey, S. Keller and P. S. M. Sweeting, "Floating-Point Mathematical Co-Processor for a Single-Chip On-Board Computer," in Surrey Space Centre School of Electronics and Physical Sciences University of Surrey, Guildford, UK, GU2 7XH, Guildford.

[20] M. Al-Ashrafy, A. Salem and WagdyAnis, "An Efficient Implementation of Floating Point Multiplier," in Electronics, Communications and Photonics Conference (SIECPC), 2011 Saudi International, 24-26 April 2011..

[21] R. P. Singh, P. Kumar and B. Singh, "Performance Analysis of 32-Bit Array Multiplier with a

CarrySave Adder and with a Carry-Look-Ahead Adder," International Journal of Recent Trends in Engineering, vol. 2, no. 6, November 2009.

[22] NiklasLotze, MauritsOrtmanns and YiannosManoli, "A Study on Self-Timed Asynchronous Subthreshold Logic," in Computer Design, ICCD 2007. 25th International Conference on 7-10 Oct. 2007, 2007.

[23] CherriceTraver, R. B. Reese and M. A. Thornton, "Cell Designs for Self-Timed FPGAs," in ASIC/SOC 14th Annual IEEE International , 2001.

[24] N. Shirazi, A. Walters and P. Athanas, "Quantitative Analysis of Floating Point Arithmetic on FPGA Based Custom Computing Machines," in Proc. of IEEE Symposium on FPGAs for Custom Computing Machines IEEE Computer Society Press, 1995, pp. 155-162.

[25] L. Lourca, T. A. Cook and W. H. Johnson, "Implementation of IEEE Single Precision Floating Point Addition and Multiplication on FPGAs," in Proc. of IEEE Symposium on FPGAs for Custom Computing Machines, IEEE Computer Society Press,, 1996, pp. 107-116..

[26] M. E. Louie and M. D. Ercegovac, "Mapping Division Algorithms to Field Programmable Gate Arrays," in Proc.26th Asilomar Conference on Signals, Systems, and Computers, 1992, pp. 371-375.

[27] S. K. V. K. V. A. M. P. Nagaraj Y and Dr. Chirag Sharma, "FPGA implementation of different adder architectures," International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 8, pp. 362-364, August 2012.

[28] Z. Navabi., VHDL Analysis and Modeling of Digital Systems, Mc Graw-Hill, Inc, 1993.

