IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

VLSI IMPLEMENTATION OF PARALLEL PREFIX ADDER FOR HIGH SPEED ARITHMETIC LOGIC CIRCUITS

Sivasakthi .R[1], Sudharshan.V[2], Sunil Kumar.B.R[3], Suriya Prakash.J[4], Dr.N .Usha Bhanu[5]
[1,2,3,4]Student [5]Professor
Electronics and Communication Engineering,
[1]SRM Valliammai Engineering College, Potheri, India.

Abstract: The basic component of arithmetic and logic circuits involves the design of high speed adders. Adder is one of the essential building blocks for signal processing applications. This paper includes analysis of various adders like Ripple-carry (RC), Carry skip (CS), Carry-lookahead (CL) and Parallel prefix (KS) adders. It is analyzed that the Parallel prefix adder is having the minimum delay of 7.820ns and 10.995ns for the 4-bit and 8-bit respectively. Power consumption is 2.918W and 5.311W for 4-bit and 8-bit respectively and it is less compared to other adders. The efficient high speed parallel prefix adder is implemented in ALU (Arithmetic and Logic Unit). The adders performance and power is analyzed using Xilinx ISE 14.7 Design suite and Xilinx vivado.

I.INTRODUCTION

ALU and Register File are the two fundamental components of a processor. Serial adders (RCA, CSA) and Parallel adders (CLA, Parallel Prefix adder) are the two types of adders. High-speed adders occupying less area and consumes low power support high-speed Arithmetic Logic Circuits. Adders are the most extensively used components in digital integrated circuits (ICs), and they are also used in essential parts of digital processors. The most commonly used operation in any application-specific processor is addition. Adders are used to accomplish addition, therefore researchers tried to improve adder performance through the advancements. It is utilized to improve arithmetic logic units because it is a vital component. However, in some situations, it is also utilized to improve other processing elements. With advancements in signal processing technology, there has been an increase in the use of portable devices, which necessitates a tradeoff between speed, area, and power. There are two techniques through which binary addition is performed namely serial and parallel. In serial addition, a block of circuits, such as a ripple carry adder (RCA) or a carry skip adder, must wait for the carry from the previous block (CSA). The RCA architecture is one of the most basic, and therefore requires more delay with less sophisticated circuitry. In order to enhance this long carry delay route, substantial study was carried out. The hardware approach includes establishing a VLSI architecture for performing multiple changes in terms of improving performance parameters including speed, hardware usage (area), and power consumption. These are the key parameters that characterize what modern digital systems require. The goal of this paper is to examine a high-performance adder and implement it in an ALU. An adder is implemented in this paper for high-speed calculation applications where speed is the most important factor. In this paper, performance analysis of different adders for various bit lengths is analyzed. (Area-delay-power).

II. EXISTING ADDERS

RIPPLE CARRY ADDER:

The addition of two numbers with bits that can be extended indefinitely is the basic operation of a ripple carry adder, as demonstrated in the Fig.1 below. The full adders are the most fundamental component of a ripple carry adder. The full adder generates a result and a carry by adding two bits and the carry bit. The created carry is passed on to the next full adder in the sequence, and so on. As a result, we can say that the delay in a ripple carry adder rises according to the number of bits.

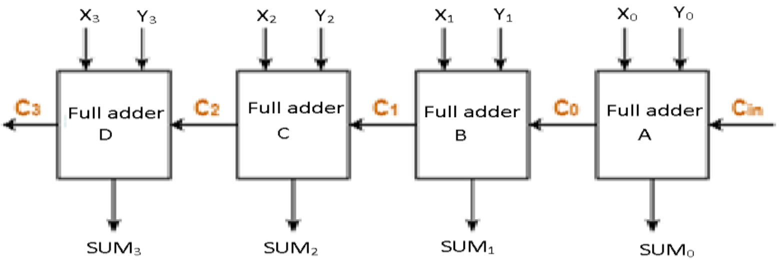


Fig.2 Block diagram of Ripple Carry adder

CARRY SKIP ADDER:

Carry skip adder reduces carry propagation time by skipping over groups of successive adder stages. The input to be added is separated into blocks of varying sizes. Within each block, ripple carry generates the sum bit and the carry. Carry computation, or skipping carry over groups of subsequent adder stages as shown in Fig.2 below is the primary premise of the carry Skip Adder. Because the carry bit for each block can now "skip" over blocks with a group propagate signal set to logic 1, the adder's critical route latency is considerably reduced.

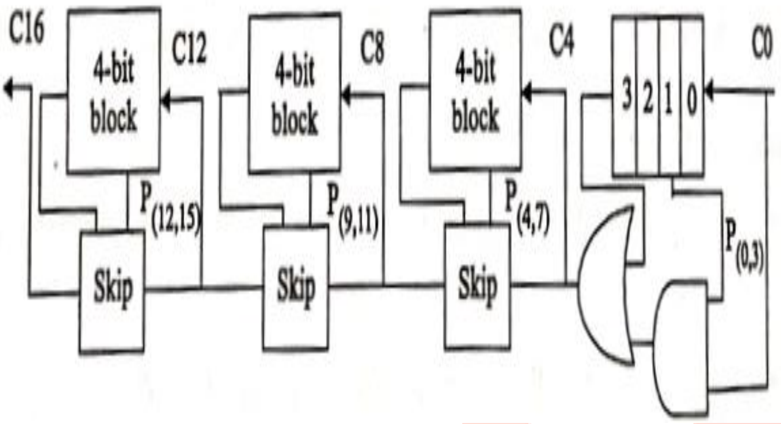


Fig.2 Block diagram of Carry skip adder

CARRY LOOK AHEAD ADDER:

Carry Look Ahead adder design is to look at lower adder bits of argument and add if higher order carry is created. By producing the carries of each stage in parallel, the Carry Look Ahead Adder reduces the propagation delay. The delay time of the CLA architecture is logarithmic in proportion to the size of the adder, allowing the carry signal propagation delay to be minimised. The following are the expressions for Gi and Pi with two binary input operands Ai and Bi:

$$G_i=A_i.B_i,$$

$$P_i=A_i\square B_i$$

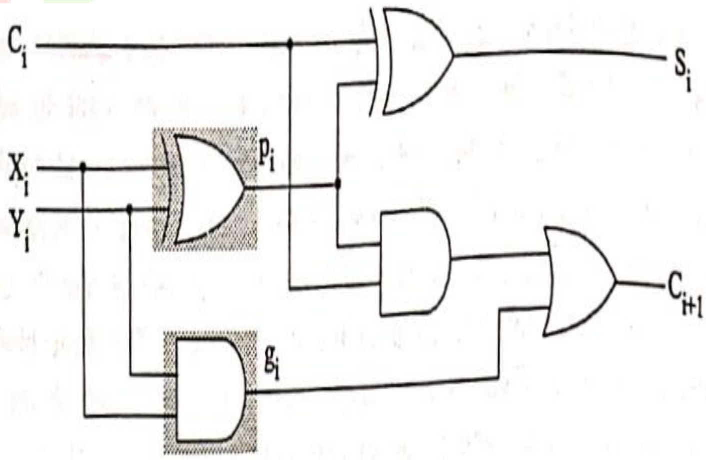


Fig.3 Block diagram of Carry look ahead adder

III. Proposed Adder

There are three stages of operation in the Parallel prefix adder as shown in Fig.4 below. Carry is generated in O(logn) time and computed quickly in parallel in parallel Prefix (KSA), but at the cost of increased area.

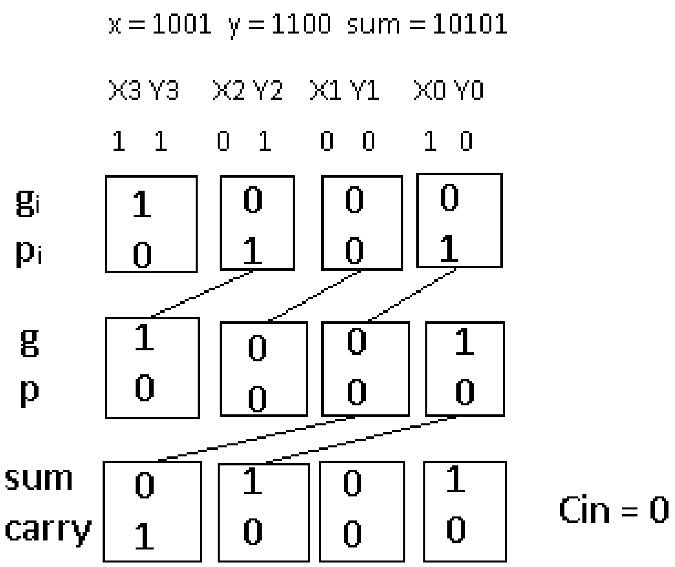


Fig.4 Block diagram of Parallel prefix adder

1) Multi-bit inputs are added during the pre-processing stage. In stage 1, generate and propagate are the two inputs.

Generate, $g_i = a_i \,\&\, b_i$

Propagate, $p_i = a_i \wedge b_i$

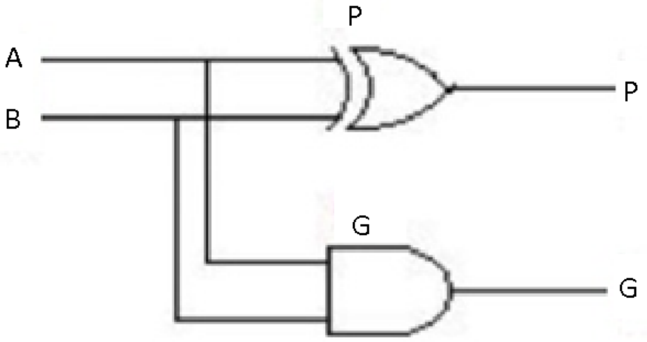


Fig.5 Block diagram of Pre-processing stage

2) Prefix network Stage calculates final carry. The final carry is calculated using the calculations below.

Generate, $g = (p_i \,\&\, g_{iprev})$ OR g_i

Propagate, $p = p_i \,\&\, p_{iprev}$

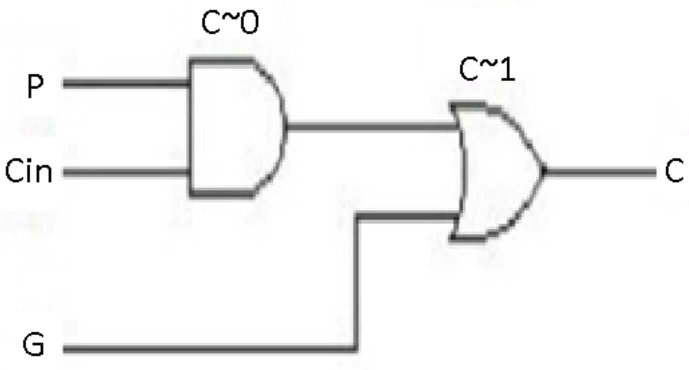


Fig.6 Block diagram of Prefix network stage

3) Post-Processing Stage calculates the total Sum. It helps to compute the total sum from carry created in prefix network stage.

Sum, $s_i = p_i$ XOR c_{i-1}

Carry, $c_i = g_i$

IV. Methodology:

The adders are simulated in Xilinx ISE Design suite 14.7 for various bitlengths using Verilog code and area, delay parameters are analysed. For power, Xilinx Vivado Design suite is used. The analysed parameters are analysed to determine the efficient high-speed adder.

V. Results and Discussion

The Xilinx Vivado and Xilinx ISE Design suite 14.7 are the software's used. Xilinx ISE is a software tool from Xilinx for synthesis and analysis of HDL designs, which primarily targets development of embedded firmware for Xilinx FPGA and CPLD integrated circuit product families. Vivado Design Suite is a software suite produced by Xilinx for synthesis and analysis of hardware description language designs, superseding Xilinx ISE with additional features for system on a chip development and high-level synthesis. Vivado represents a ground-up rewrite and re-thinking of the entire design flow. Vivado also introduces high-level synthesis, with a toolchain. The device utilization and power analysis for 4-bit is shown in Table.1 and Table.2 respectively. The device utilization and power analysis for 8-bit is illustrated in Table.3 and Table.4 respectively.

Table 1. Device utilization summary for 4-bit adders

PARAMETERS	RCA	CSA	CLA	Parallel prefix adder (KSA)
NUMBER OF SLICES	4 out of 960	3 out of 960	5 out of 960	4 out of 960
NUMBER OF 4 INPUT LUTs	8 out of 1920	6 out of 1920	10 out of 1920	7 out of 1920
NUMBER OF BONDED IO'S	14 out of 66	13 out of 66	14 out of 66	14 out of 66
TOTAL DELAY	8.959ns	10.029ns	7.930ns	7.820ns

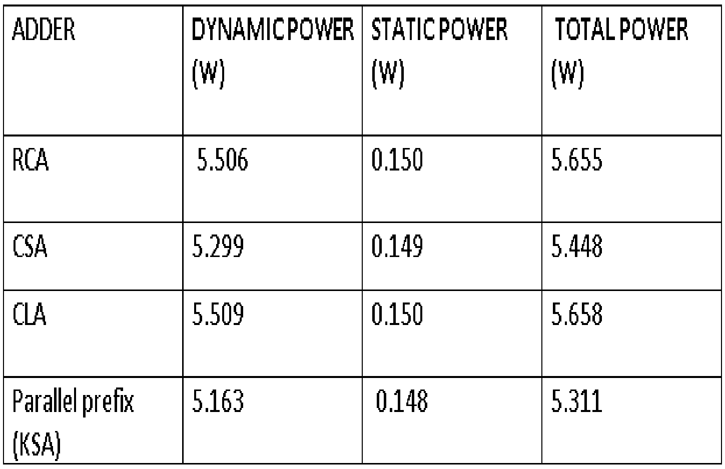
Table 2. Power of 4-bit adders

ADDERS	DYNAMIC POWER (W)	STATIC POWER (W)	TOTAL POWER (W)
RCA	2.868	0.139	3.007
CSA	2.865	0.139	3.005
CLA	2.703	0.139	2.843
Parallel prefix (KSA)	2.779	0.139	2.918

Table 3. Device utilization summary of 8-bit adders

PARAMETER	RCA	CSA	CLA	Parallel Prefix (KSA)
NUMBER OF SLICES	9 out of 960	10 out of 960	8 out of 960	9 out of 960
NUMBER OF 4 INPUT LUTs	16 out of 1920	18 out of 1920	15 out of 1920	16 out of 1920
NUMBER OF BONDED IO'S	26 out of 66	25 out of 66	25 out of 66	25 out of 66
TOTAL DELAY	13.203ns	12.057ns	11.986ns	10.995ns

Table 4. Power of 8-bit adders



ADDER	DYNAMIC POWER (W)	STATIC POWER (W)	TOTAL POWER (W)
RCA	5.506	0.150	5.655
CSA	5.299	0.149	5.448
CLA	5.509	0.150	5.658
Parallel prefix (KSA)	5.163	0.148	5.311

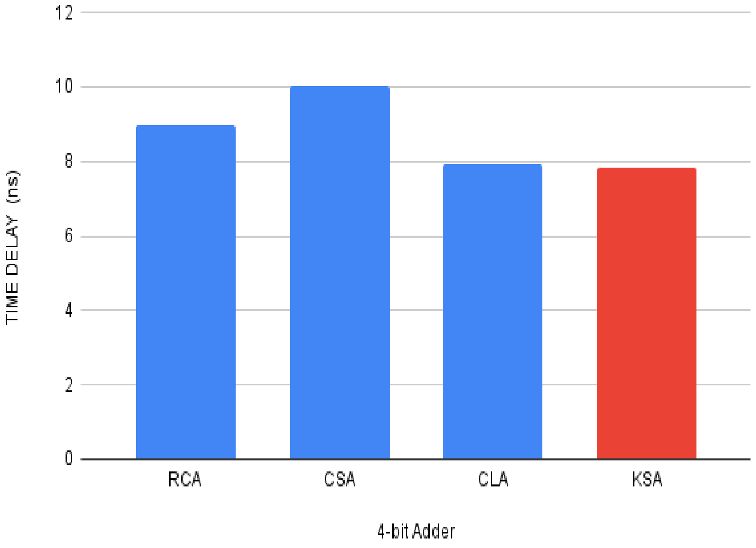


Fig.7 Time delay of 4-bit adders

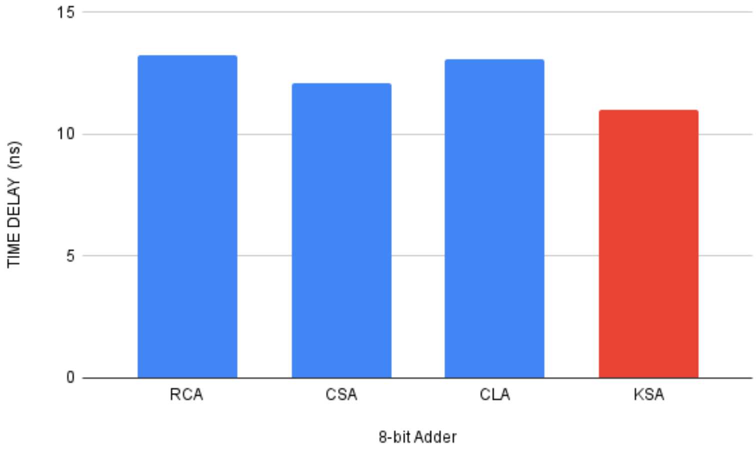


Fig.8 Time delay of 8-bit adders

From Fig.8 it is observed that the parallel prefix adder(KSA) 8-bit shows good result in terms of Speed.

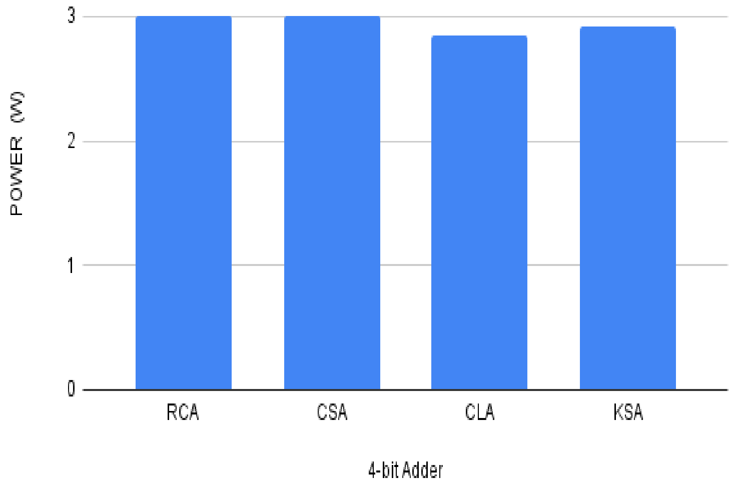


Fig.9 Power consumption of 4-bit adders

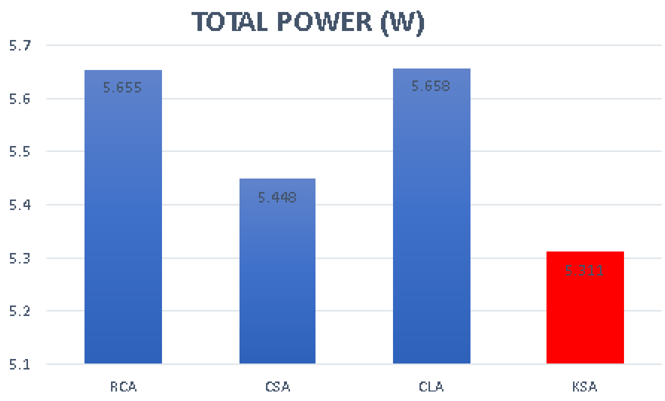


Fig.10 Power consumption of 8-bit adders

Fig.10 shows that the power consumption of parallel prefix adder (KSA) 8-bit is less(5.311W) compared to other adders.

VI. Implementation in ALU

The arithmetic logic unit (ALU) is a digital combinational digital circuit used to perform arithmetic and bitwise operations on binary integers. The arithmetic operation addition in ALU is replaced by the adder proposed in this paper.

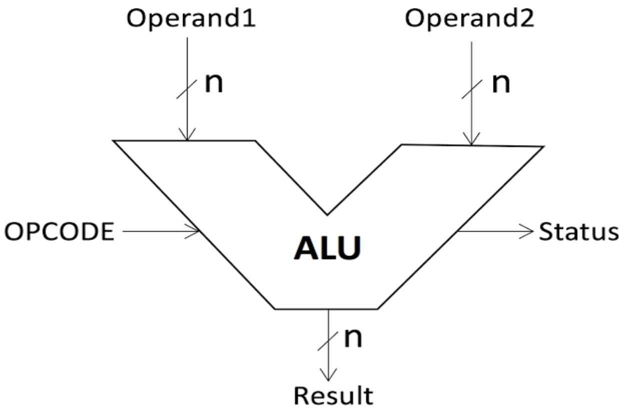
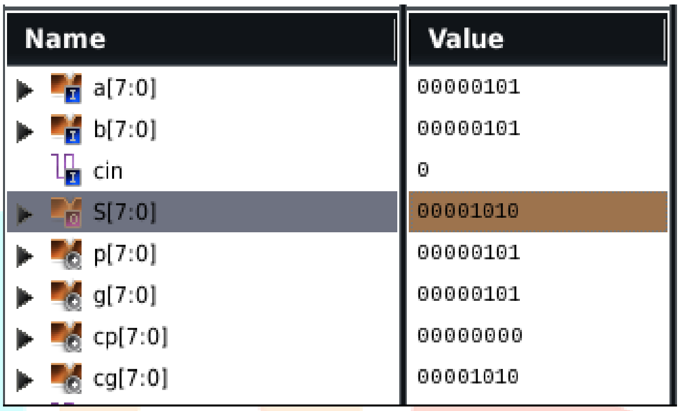


Fig.11 Block diagram of ALU

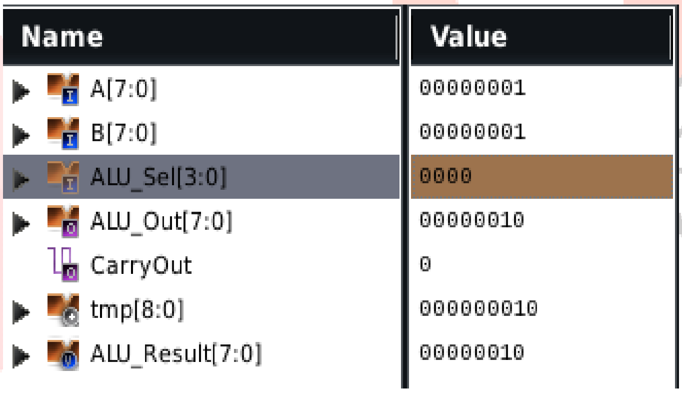
Table 5. Arithmetic and Logic operations of ALU

ALU_SELECT (OPCODE)	ALU OPERATION
0000	ALU_OUT = A + B
0001	ALU_OUT = A - B
0010	ALU_OUT = A * B
0011	ALU_OUT = A / B
1000	ALU_OUT = A and B
1001	ALU_OUT = A or B
1010	ALU_OUT = A xor B
1011	ALU_OUT = A nor B
1100	ALU_OUT = A nand B



Name	Value
a[7:0]	00000101
b[7:0]	00000101
cin	0
S[7:0]	00001010
p[7:0]	00000101
g[7:0]	00000101
cp[7:0]	00000000
cg[7:0]	00001010

Fig.12 8-bit Parallel prefix adder simulation output



Name	Value
A[7:0]	00000001
B[7:0]	00000001
ALU_Sel[3:0]	0000
ALU_Out[7:0]	00000010
CarryOut	0
tmp[8:0]	000000010
ALU_Result[7:0]	00000010

Fig.13 ALU(8-bit) simulation output

The parallel prefix adder is implemented in ALU and the output is shown in Fig.13. The total time delay for the ALU to perform addition operation using the parallel prefix adder (KSA) is 11.075ns which includes Logic-7.976ns and Route-3.099ns.

VII. Conclusion

The primary purpose is to improve the speed of the addition process in the ALU. In this paper, RCA, CSA, CLA and Parallel prefix adder (KSA) have been analysed for different parameters. Parallel Prefix adder (KSA) results in 7.820ns and 10.995 ns for 4-bit and 8-bit respectively. This leads to conclusion that 8-bit Parallel prefix adder uses the minimum delay with reasonable area. The power consumption is 2.918 W and 5.311 W for 4-bit and 8-bit respectively. Due to this reason, 8-bit Parallel prefix adder is implemented in ALU and performance is analysed.

VIII. Acknowledgement

We thank the Department of ECE, SRM VEC for their support and laboratory facilities for completion of the project.

REFERENCES

[1] Garima Thakur, Harsh Sohal, Shruti Jain, “Design and analysis of High Speed Parallel Prefix Adder for Digital Circuit Design Applications”, in IEEE Access, ph:978-1-7281-6644-5/20/$31.00, July 2020.

[2] Akash Kumar R, Ashish Ashok Shetty, Mohammed Saud, Preemal Sharanya Serrao, Rohan Pinto, “Design and implementation of 64-bit Parallel Prefix Adder”, in IEEE Access, ph:978-1-7281-9885-9/20/$31.00, 2020.

[3] U. Dinesh Kumar J R, Ganesh Babu C, Karthi S P, “Performance Analysis of 16-bit Adders in High Speed Computing Applications”, in IEEE Access,ph:978-1-7281-3250-1/19/$31.00, 2019.

[4] Chandrasekaran, S., Nageswaran, U.B., A mutated addition–subtraction unit to reduce the complexity of FFT, Applied Nanoscience, 2022.

[5] Niharika Mittal, Dr.Shipli Birla, Design and implementation of high-speed adders for various digital applications, International Journal of Advance Engineering and Research Development, Volume 5, Special Issuse 01, Jan-2018.

[6] R, Pooja and N, Padmapriya and N, Minesh Kanna Ram and S, Gokul and Chandrasekaran, Saravanakumar, PRBS Generator using FinFET, International Journal of Grid and Distributed Computing, 2020, Vol.13 No.01.