International Journal of Future Generation Communication and Networking
Vol. 14, No. 1, (2021), pp. 1496-1506

Area Efficient Carry Select Adder Using Parallel Prefix Adder Structure

A Krishna Kumar
Assistant Professor,
Department of ECE, Chaitanya Bharathi Institute of Technology, Hyderabad, Telangana, INDIA

Abstract

Adders are the basic building blocks of modern digital integrated circuit design and are the necessary part of Digital Signal Processing (DSP) applications. The prerequisite ofthe adder is that, it is primarily fast and secondarily efficient in terms of power consumption and chip area.

A Conventional Carry Select Adder has two Ripple Carry Adder (RCA) which consists of cascaded of "N" single bit full adders. Output carry of previous adder becomes the input carry of next full adder and so on. Therefore, the carry of this adder traverses longest path called worst case delay path through N stages. Now as the value of N increases, delay of adder will also increase in a linear way.

The proposed Carry Select Adder structure which adopts modified parallel prefix adder at the upper part and Binary to Excess-1 convertor at the lower part of a design. This adder is being compared with Conventional CSA and Carry Select Adder using Parallel Prefix Adder in terms of area and performance. The simulations and synthesis of the proposed adder are done using Verilog HDL in XILINX ISE.

1. INTRODUCTION

As the requirement for high performance processor grows, there is a constant need to enhance the performance of data path units. Addition is the most commonly used arithmetic operation and the performance of VLSI processor is enormously impacted by performance of resident adder. A high performance adder with low power consumption designed with minimum area plays an indispensable role in large portion of the hardware circuits. For adding two binary numbers several types of adders have been designed, for example ripple-carry, carry-skip, carry-select adder (CSA) and parallel prefix adders (PPAs). The major speed restriction in the conventional adder circuits, such as ripple carry adder (RCA) and carry save adder arises from the long computation time required for generating the outputs. CSA and carry look-ahead architectures have been suggested to reduce large carry propagation delay of adders.

Ripple Carry Adder consists of cascaded of "N" single bit full adders. Output carry of previous adder becomes the input carry of next full adder and so on. Therefore, the carry of this adder traverses longest path called worst case delay path through N stages. Now as the value of N increases, delay of adder will also increase in a linear way. Therefore, RCA has the lowest speed amongst all the adders because of large propagation delay but it occupies the least area.

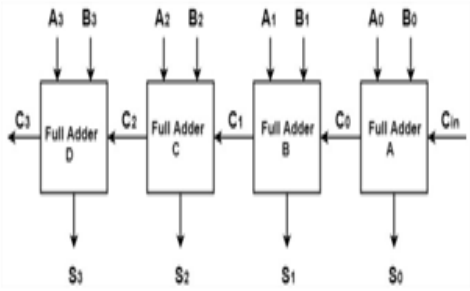


Fig.1.1 Ripple Carry Adder

ISSN: 2233-7857 IJFGCN

2. LITERATURE SURVEY

To justify the objectives of the project, to realize and to make it easier many books and several journals were referred. Adders are the subject of intensive research. From Ripple Carry Adder (RCA) to Parallel Prefix Adders (PPA), methods have been proposed to improve speed, power consumption and area. The basic concepts of different adder architectures and their significance in different applications have been studied. Among the all-digital adders, Carry select Adder using Parallel Prefix Adders is the efficient adder that improves on the delay of a Ripple Carry Adder with little effort compared to other adders.

OBJECTIVES

1. Designing of Conventional CSA.
2. Designing of Binary to Excess-1 Convertor.
3. Designing of Parallel Prefix Adder.
4. Designing of Modified Parallel Prefix Adder.
5. Designing of Multiplexer.
6. Designing of Carry Select Adder using the Parallel Prefix Adder.
7. Designing of Carry Select Adder using the Modified Parallel Prefix Adder.
8. Implementing the Verilog code for all Adders.

3. ADDER ARCHITECTURE:

Types of adder

3.1Half Adder

The Half Adder is an example of a simple, functional digital circuit built from two logic gates. The Half Adder ad stone-bit binary numbers (AB). The output is the sum of the two bits (S) and the carry(C). Note in Fig.2.1 how the same two inputs are directed to two different gates. Table 2.1 gives the truth table of Half Adder.

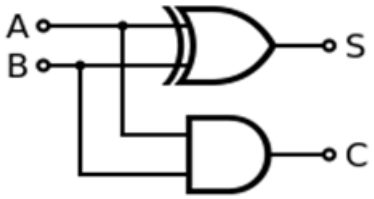


Fig3.1 Schematic diagram of Half Adder

Table 3.1 Truth table of Half Adder (1Bit)

Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

3.2 Full adder

A full adder adds binary numbers and accounts for values carried in as well as out. A one-bit full adder adds three one-bit numbers, often written as A, B, and C_{in} And Bare the operands, and in is a bit carried in from the previous less significant stage as shown in Fig.2.2. The full adder is usually a component in a cascade of adders, which add 8, 16, 32, etc. Bit binary numbers. The circuit produces a two-bit output, output carry and sum. Whereas the equation of the sum and carry is

$$S = A \oplus B$$

$$C_{out} = A.B$$

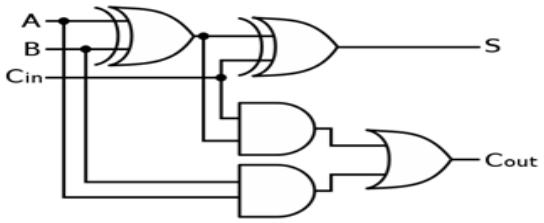


Fig.3.2 Block diagram of full adder

Table 3.2 Truth table Of Full adder (1Bit)

X	Y	C_{in}	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

3.3 Ripple Carry Adder

Arithmetic operation like Addition, Subtraction, Multiplication, Division are basic operation to be implemented digital computer using basic gates among all arithmetic operation if we can implement Addition then it is easy to perform Multiplication Repeated Addition. Half Adders can be used to add two one-bit binary numbers. It is also possible to create a logical circuit using multiple adder to add N bit binary number each full adder inputs carry, which is the output carry of the previous adder. This kind of adder is a Ripple Carry Adder shown in Fig.2.3, since each carry bits "ripples" to the next full adder. The first full adder may be replaced by the Half Adder.

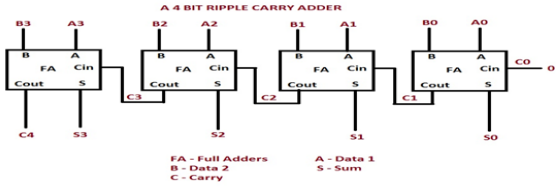


Fig.3.3 Block diagram of 4-bit Ripple Carry Adder (RCA)

3.4 Carry Select adder:

A Carry Select Adder is a way to implement an adder, which is a logic element that computes the (n+1) bit sum of two n-bit numbers. The carry-select adder is simple but rather fast. The carry-select adder generally consists of two ripple carry adders and a multiplexer. Adding two n-bit numbers with a carry-select adder is done with two adders(therefore two ripple carry adders)in order toper form the calculation twice, one time with the assumption of the carry being zero and the other as summing one. After the two results are calculated, the correct sum, as well as the correct carry, is then selected with the multiplexer once the correct carry is known.

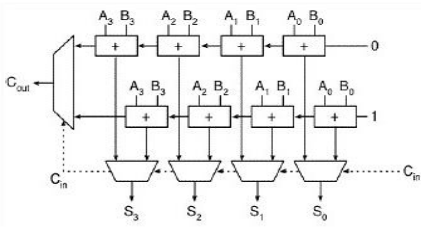


Fig.3.4 Carry Select Adder

3.5 PARALLEL PREFIX ADDER

Parallel prefix adders are used to speed up the binary additions as they are very flexible. The structure of Carry Look Ahead Adder (CLA) is used to obtain parallel prefix adders. Tree structures are used to increase the speed of arithmetic operation. Parallel prefix adders are used for high performance arithmetic circuits in industries as they increase the speed of operation.

The construction of parallel prefix

Adder involve in three stages:
1. Pre- processing stage
2. Carry generation network
3. Post processing stage

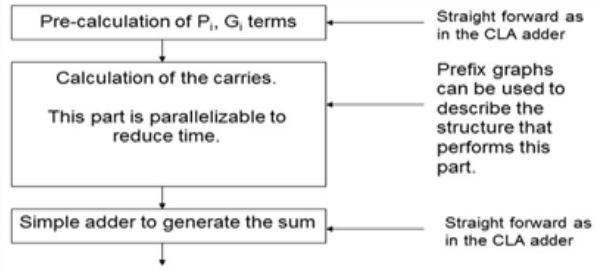


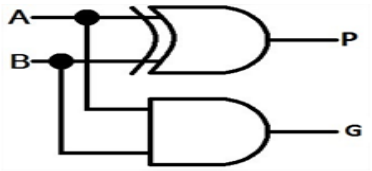
Fig.3.5 Block Diagram for Parallel Prefix Adder

3.6 Pre-possessing stage:

In this section, we have briefly explained the parallel prefix computation formula by using 2 binary operands ofN bits namely 'a' (a1, a2 ……aN) and 'b' (b1, b2 …… bN). In parallel prefix scheme, to efficiently compute the carry signal from 2N number of bits, two signals namely carry generate (Gi) and carry propagate (Pi) are needed .We can generalize these signals to describe that a group of input bits (ai, bi) generate the carry propagate (Pi) andcarry generate (Gi) signals. A carry generate (Gi) signal represent those group of bits which generates a carry, if it scout is true independent of Cin, and carry propagate (P_i) signal refers to those group of bits to propagate a carry if its Cout is true when there is a Cin. These group signals are generated using equations.

$$P_i = A_i \oplus B_i \qquad (3.5)$$

$$G_i = A_i . B_i \qquad (3.6)$$



3.7 Carry generation network:

In this stage, we compute carries equivalent to each bit. Implementation of these operations is carried out in parallel. After the computation of carries in parallel they are segmented into smaller pieces. Carry propagate and generate are used as intermediate signals which are given by the logic.
The logic equations for calculating carry signals for 4 bit input can be expressed is

Post processing Stage:

This is the concluding step to compute the summation of input bits. It is common for all the adders and the sum bits are computed by logic equations

$$C_i = G_i + P_i . C_{i-1}$$

$$S_i = P_i \oplus C_{i-1}$$

$$S0 = P0 \oplus Cin$$

$$S1 = P1 \oplus C0$$

$$S2 = P2 \oplus C1$$

$$S3 = P3 \oplus C2$$

4. METHODOLOGY

1. Designing of Binary to Excess-1 Convertor and multiplexer using the logical gates.
2. Designing of Modified Parallel Prefix Adder involves three steps
    - The pre-processing stage focuses on propagate and generate.
    - carry generation stage focuses on carry generation
    - Post-processing stage focuses on result.
3. Design of Carry select adder by integrating Modified Parallel Prefix Adder, Binary to Excess-1 Convertor and Multiplexers.
4. Writing the Verilog code for entire design in Xilinx ISE.
5. Performing the Simulation and Synthesis.
6. Generating a bit file for the target system.

4.1 SOFTWARE IMPLEMENTATION

A software development process or life cycle is a structure imposed on the development of a software product. There are several models for such operations, each describing approaches to a variety of tasks or activities that take place during the process.

The software development process includes the following steps,
[1] Detailed requirements
[2] Developing an algorithm
[3] Drawing a flowchart
[4] Coding

[5] Software simulation

The software tools used in this project is Xilinx ISE

Xilinx ISE :Xilinx ISE (Integrated Synthesis Environment) is a software tool produced by Xilinx for simulation, synthesis and analysis of HDL designs, and it enables the developer to synthesize ("compile") their designs, perform timing analysis, examine RTL diagrams. And it generates a bit file for the target board

4.2 SOFTWAREDESCRIPTION

Xilinx ISE (**I**ntegrated **S**ynthesis **E**nvironment) is a software tool produced by Xilinx for synthesis and analysis of HDL designs, enabling the developer to synthesize ("compile") their plans, perform timing analysis, examine RTL diagrams, simulate a design's reaction to different stimuli, and configure the target device with the programmer.

Xilinx ISE is a design environment for FPGA products from Xilinx, and is tightly- coupled to the architecture of such chips, and cannot be used with FPGA products from other vendors. The Xilinx ISE is used mainly for circuit synthesis and design, while ISIM or the ModelSim logic simulator is used for system-level testing. Other components shipped with the Xilinx ISE include the Embedded Development Kit (EDK), a Software Development Kit (SDK) and Chip Scope Pro.

4.3 Synthesis

Xilinx patented algorithms for synthesis allow designs to run up to 30% faster than competing programs and allows greater logic density which reduces project time and costs.

Also, due to the increasing complexity of FPGA fabric, including memory blocks and I/O blocks, more complex synthesis algorithms were developed that separate, unrelated modules into slices, reducing post-placement errors.

IP Cores are offered by Xilinx and other third-party vendors, to implement system-level functions such as digital signal processing (DSP), bus interfaces, networking protocols, processing, embedded and peripherals. Xilinx has been instrumental in shifting designs from ASIC-based implementationtoFPGA-basedimplementation.

4.4 SPARTAN 6 DESCRIPTION:

The Spartan-6 family provides leading system integration capabilities with the lowest total cost for high-volume applications. The thirteen-member family delivers expanded densities ranging from 3,840 to 147,443 logic cells, with half the power consumption of previous Spartan families, and faster, more comprehensive connectivity. Built on a mature 45 nm low-power copper process technology that delivers the optimal balance of cost, power, and performance, the Spartan-6 family offers a new, more efficient, dual-register 6-input lookup table (LUT) logic and a rich selection of built-in system-level blocks.

These include 18 Kb (2 x 9 Kb) block RAMs, second generation DSP48A1 slices, SDRAMmemory controllers, enhanced mixed-mode clock management blocks, Select IO technology, power optimized high-speed serial transceiver blocks, PCI Express® compatible Endpoint blocks, advanced system-level power management modes, auto-detect configuration options, and enhanced IP security with AES and Device DNA protection. These features provide a low cost programmable alternative to custom ASIC products with unprecedented ease of use. Spartan-6 FPGAs offer the best solution for high-volume logic designs, consumer-oriented DSP designs, and cost-sensitive embedded applications. Spartan-6 FPGAs are the programmable silicon foundation for Targeted Design Platforms that deliver integrated software and hardware components that enable designers to focus on innovation as soon as their development cycle begins.

ISSN: 2233-7857 IJFGCN

Figure 4.1: Digilent Atlys Spartan-6 FPGA Trainer Board

Spartan-6 FPGAs store the customized configuration data in SRAM-type internal latches. The number of configuration bits is between 3 Mb and 33 Mb depending on device size and user-design implementation options. The configuration storage is volatile and must be reloaded whenever the FPGA is powered up. This storage can also be reloaded at any time by pulling the PROGRAM_B pin Low. Several methods and data formats for loading configuration are available. Bit-serial configurations can be either master serial mode, where the FPGA generates the configuration clock (CCLK) signal, or slave serial mode, where the external configuration data source also clocks the FPGA. For byte-wide configurations, master SelectMAP mode generates the CCLK signal while slave SelectMAP mode receives the CCLK signal for the 8- and 16-bit-wide transfer. In master serial mode, the beginning of the bit stream can optionally switch the clocking source to an external clock, which can be faster or more precise than the internal clock. The available JTAG pins use boundary-scan protocols to load bit-serial configuration data.

The bit stream configuration information is generated by the ISE software using a program called BitGen. The configuration process typically executes the following sequence:

- Detects power-up (power-on reset) or PROGRAM_B when Low.
- Clears the whole configuration memory.
- Samples the mode pins to determine the configuration mode: master or slave, bit-serial or parallel.
- Loads the configuration data starting with the bus-width detection pattern followed by a synchronization word, checks for the proper device code, and ends with a cyclic redundancy check (CRC) of the complete bit stream.
- Starts a user-defined sequence of events: releasing the internal reset (or preset) of flip-flops, optionally waiting for the DCMs and/or PLLs to lock, activating the output drivers, and transitioning the DONE pin toHigh.

The Master Serial Peripheral Interface (SPI) and the Master Byte-wide Peripheral Interface (BPI) are two common methods used for configuring the FPGA. The Spartan-6 FPGA configures itself from a directly attached industry-standard SPI serial flash PROM. The Spartan-6 FPGA can configure itself via BPI when connected to an industry-standard parallel NOR flash. Note that BPI configuration is not supported in the XC6SLX4, XC6SLX25, and XC6SLX25T nor is BPI available when using Spartan-6 FPGAs in TQG144 and CPG196 packages. Spartan-6 FPGAs support MultiBoot configuration, where two or more FPGA configuration bit streams can be stored in a single configuration source. The FPGA application controls which configuration to load next and when to load it. Spartan-6 FPGAs also include a unique, factory-programmed Device DNA identifier that is useful for tracking purposes, ant cloning designs, or IP protection. In the largest devices, bit streams can be copy protected using AESencryption.

5. SIMULATION RESULTS

To design Conventional Carry Select Adder the following modules like Ripple Carry Adder and MUX is required. Ripple Carry Adder is a combination of Full Adders and these are designed using AND and OR gates. The Multiplexers are designed by using the logical gates AND and OR gates only.

To design Proposed Carry Select Adder the following modules like Parallel Prefix Adder , BEC and Multiplexers are required. These modules are designed using AND, EXOR and OR gates.

AND GATE

The RTL schematic diagram of AND gate is shown in Figure 5.1. AND gate is used in Parallel Prefix Adder for Carry Generation Unit, Bit Wise PG logic and it is alos used in Binary to Excess -1 code converter(BEC). The output of AND gate is logic 'high' if both the inputs are logic 'high' otherwise it is logic 'low

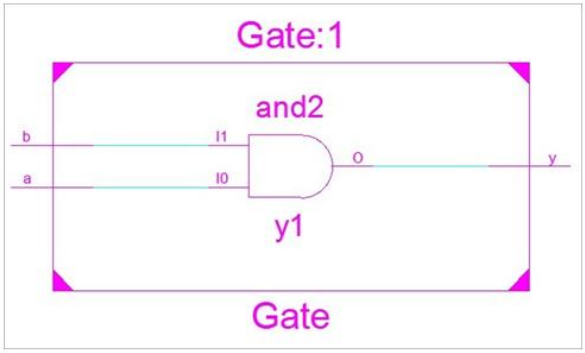


Fig.5.1 RTL Schematic of AND Gate

OR GATE

The RTL schematic diagram of OR gate is shown in Figure 5.2. OR gate is used in Parallel Prefix Adder for carry generation unit. The output of OR gate is logic 'high' if any one of the input is logic 'high'.

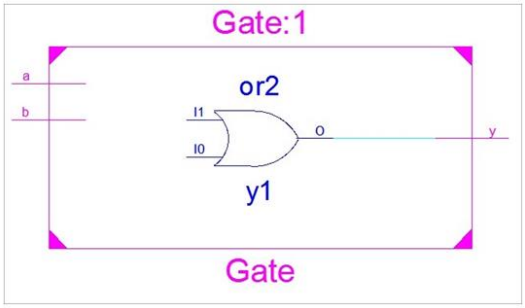


Fig.5.2 RTL Schematic of OR Gate

NOT GATE

The RTL schematic diagram of NOT gate is shown in Figure 5.3. The output of NOT gate is inverted of input.

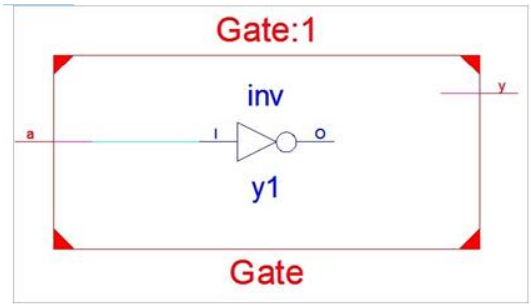


Fig.5.3 RTL Schematic of NOT Gate

EXOR GATE

The RTL schematic diagram of XOR gate is shown in Figure 5.4. The output of EXOR gate is high when both the inputs are different.

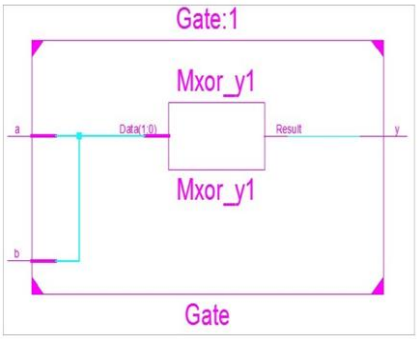


Fig.5.4 RTL Schematic of EXOR Gate

The simulation for 1-Bit EXOR Gate as shown in Figure 5.5.

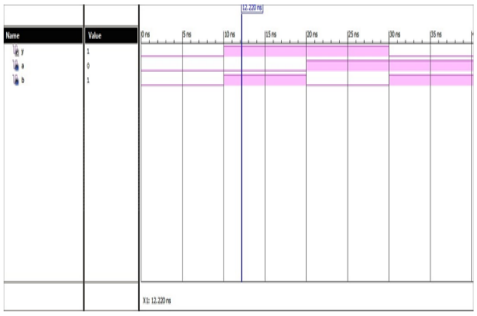


Fig.5.5 RTL Simulation of EXOR Gate

BEC is used to generate the sum output for Cin=0, The RTL Schematic of BEC is

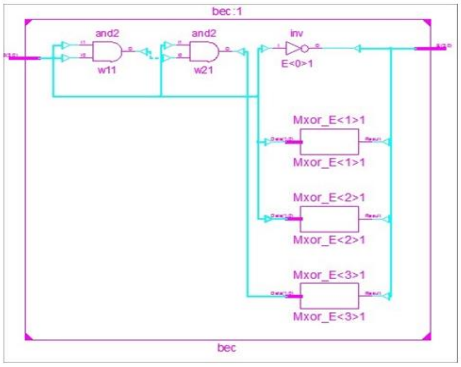


Fig. 5.6 RTL Schematic of BEC for 4-Bit

The simulation results for 4-bit, 8-bit and 16-bit are shown in respectively

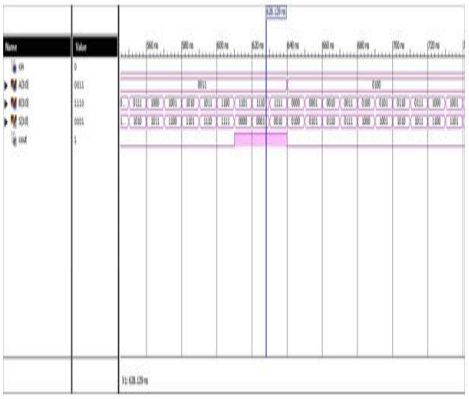


Fig 5.7. Simulation results for 4-bit CSA

Fig.5.8 Simulation results for 8-bit CSA

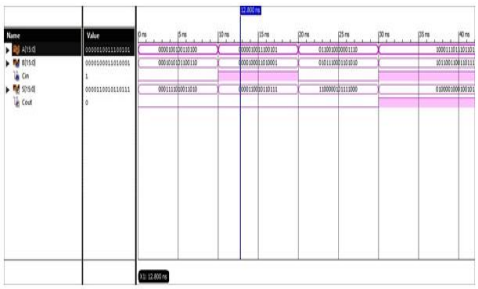


Fig. 5.9 Simulation results for 16-bit CSA

5 RESULTS

Table 6.1 shows the experimental results of the Carry Select Adder by using Modified Parallel Prefix Adder compared to Conventional CSA and CSA by using Parallel Prefix Adder with respect to area and delay. All the simulations and synthesis are carried out using XILINX Seethe proposed CSLA shows significant improvement in percentage area reduction of 20.4%, 6 % and Delay reduction of 32%, 14% as compared to Conventional CSA and CSA using PPA.

Table 5.1 Comparison results with different adders

Word Size	CSA Structure	Delay (ns)	Transistor Count(T)
4-Bit	Conventional CSA	9.595	500
	CSA using PPA	7.579	426
	CSA using modified PPA	6.494	376
8-Bit	Conventional CSA	12.425	1000
	CSA using PPA	9.8148	852
	CSA using modified PPA	8.414	752
16-bit	Conventional CSA	18.078	2000
	CSA using PPA	14.2805	1704
	CSA using modified PPA	12.244	1504

6. CONCLUSION AND FUTURESCOPE

From the results it can be concluded that the modified CSLA structure combines the advantage of area reduction provided by BEC and the high speed achieved through modified Parallel Prefix Adder. Further scope would include in the better parallel prefix structure and implementation of modified CSLA adder structure for 64, 128- bit CSLA.

REFERENCES

[1]. Abhishek R Hebbar, Piyush Srivastava, Vinod Kumar "Design of High Speed Carry Select Adder using Modified Parallel Prefix Adder" published in ELSEVIER ,2018

[2]. Raghava et al. High Speed Power Efficient Carry Select Adder Design, IEEE Computer society annual symposium on VLSI, 2017.

[3]. Er. Aradhana Raju, Richi Patnaik, Ritto Kurian Babu, Purabi Mahato “Parallel Prefix Adders- A Comparative Study ForFastest Response”

[4] Sajesh Kumar, U,MohamedSalih K.K and SajithK."Design and Implementation of Carry SelectAdder without using Multiplexers"978-4673-1627-9112/$31.00 2012 IEEE.

[5] O.J.Bedrij, "Carry-select adder, "IRE Trans Electron, Comput. PP. 340-344, 1962.

[6] Santosh Elangadi and SuhasShirol "Design and Characterization of High Speed Carry Select Adder"2348- 4748, volume 1, Issue 6, July 2014.

[7] B. Ramkumar and Harish M Kittur " Low-Power and Area-Efficient Carry Select Adder"371-375,VOL. 20, NO. 2, FEBRUARY 2012.

[8] k.BalaSindhuri, N.UdayKumar, D.V.N.Bharathi, B.Tapasvi. "I28-Bit Area -Efficient Carry Select Adder" IJRASET.

[9] A Verilog HDL Primer by J.Bhaskar.