

COMPARATIVE PERFORMANCE ANALYSIS OF PARALLEL PREFIX ADDER

Submitted in partial fulfillment of the requirements for the award of Bachelor of
Engineering Degree in Electronics and Communication Engineering

by

RAGHUMANDA NANI (40130234)

THAMMINIENI VIVEK (40130234)



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SCHOOL OF ELECTRICAL AND ELECTRONICS

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

CATEGORY -1 UNIVERSITY BY UGC

Accredited with Grade "A++" by NAAC | 12B status by UGC | Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 019

APRIL – 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **R. NANI (40130186)**, **T. VIVEK (40130234)** who carried out the Project entitled "**COMPARATIVE PERFORMANCE ANALYSIS OF PARALLEL PREFIX ADDERS**" under our supervision from November 2023 to April 2024.

Internal Guide

Ms. I. MARY SAJIN SANJU, M.Tech (Ph.D).,

Head of the Department

Dr. T. RAVI, M.E, Ph.D.,

Submitted for Viva voce Examination held on 17.04.2024

Internal Examiner

External Examiner

DECLARATION

We **R. NANI** and **T. VIVEK** hereby declare that the Project Report entitled "**COMPARATIVE PERFORMANCE ANALYSIS OF PARALLEL PREFIX ADDERS**" done by us under the guidance of **Ms. I. MARY SAJIN SANJU, M. Tech (Ph.D)**, Dept of Electronics and Communication Engineering, **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY, CHENNAI**, is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in **ELECTRONICS AND COMMUNICATION ENGINEERING**.

DATE: 17-04-2024

PLACE: Chennai

SIGNATURE OF THE CANDIDATES:

1. R. Nani
2. T. vivek

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to **The Board of Management of Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N.M. NANDHITHA M.E., Ph.D., Dean, School of Electrical and Electronics Engineering**, and **Dr. T. RAVI M.E., Ph.D., Head of the Department of Electronics and Communication Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Ms. I. MARY SAJIN SANJU, M.Tech(Ph.D.)**, dept. of Electronics and Communication Engineering for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

Every CPU is built around the Arithmetic Logic Unit (ALU), which is centered around the vital Adder circuit. With the use of 2's complement arithmetic, the adder is in charge of both addition and subtraction and is a crucial component of multiplier circuits. Since addition is the most basic arithmetic operation, VLSI arithmetic experts have been fascinated with the design and optimization of the adder. A number of conventional and novel techniques have been developed over time with the goal of improving addition efficiency with respect to area, power consumption, latency, and gate output loading. This paper explores the classic addition techniques, analyses their performance in detail, and then compares and assesses the effectiveness of multiple parallel prefix adders. Parallel prefix adders are a topic of discussion since they are acknowledged as the most effective DSP chip solution to date. Finding the best adder for a given application is the main goal, with low on-chip power consumption, resource efficiency, and quick computation times being given priority. The Vivado Design Suite's Verilog Hardware Description Language is used to implement the adders that are being shown, and the designs are realized on the Xilinx ZYNQ XC7Z020 (7000 series) SoC. The study then looks into the adders' power usage, latency, and hardware utilization. Most notably, this work presents a new addition to the conventional methods the Hybrid BK-KS Adder. The Hybrid BK-KS Adder, which combines components of the Brent-Kung and Kogge-Stone architectures, seeks to achieve a compromise between performance metrics and may represent a breakthrough in terms of space usage, power efficiency, and latency.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	v
	LIST OF ABBREVIATIONS	ix
	LIST OF FIGURES	x
	LIST OF TABLES	xi
1	INTRODUCTION	1
	1.1 FULL ADDER	2
	1.2 CONVENTIONAL MULTI-BIT ADDERS	2
	1.3 GENERAL PROPAGATE LOGIC	3
	1.4 FEATURES OF VLSI DESIGN	
	1.4.1 Vlsi Design Overview	5
	1.4.2 Vlsi Design Tools	5
	1.4.3 Vlsi Design Styles	6
	1.4.4 Challenges In Vlsi Design	6
	1.4.5 Low Power Consumption	6
	1.5 ADVANTAGES OF VLSI USING CMOS	
	1.5.1 Low Area Occupancy	7
	1.5.2 Low Power Dissipation	7
	1.5.3 Less Area Efficient Architecture In Vlsi	8
	1.5.4 Less Delay In Vlsi	9
	1.5.5 Simple Architecture	9

1.6	DISADVANTAGES OF VLSI DESIGN	
1.6.1	Complexity Design	10
1.6.2	High Initial Costs	10
1.6.3	Power Consumption	11
1.6.4	Design Complexity	12
2	LITERATURE SURVEY	
2.1	GENERAL SURVEY	13
2.2	OVERVIEW OF EXISTING RESEARCH	16
3	AIM AND SCOPE OF THE EXISTING DESIGN	
3.1	AIM OF THE PROJECT	17
3.2	CONVENTIONAL ADDERS	
3.2.1	Carry Select Adder	17
3.2.2	Carry Skip Adder	19
3.2.3	Carry Look Ahead Adder	20
3.2.4	Carry Save Adder	22
3.3	PARALLEL PREFIX ADDERS	
3.3.1	Sklansky Adder	25
3.3.2	Koggle Stone Adder	26
3.3.3	Brent Kung Adder	27
3.3.4	Ladner Fischer Adder	28
3.4	IMPLEMENTATION OF EXISTING WORK	29

4	METHODS AND ALGORITHMS USED	
4.1	PROPOSED WORK	30
4.2	XILINX IMPLEMENTATION	30
4.3	VERILOG DESIGN	31
4.4	VIVADO SIMULATOR	32
4.5	FPGA DESIGN SYNTHESIS	33
4.6	BITSTREAM GENERATION	35
4.7	STANDARDS	35
4.8	CONSTRAINTS	36
4.9	TRADE OFF	38
5	RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS	
5.1	HYBRID BRENT KUNG-KOGGLE STONE ADDER	39
5.2	PERFORMANCE ANALYSIS	41
6	SUMMARY AND CONCLUSION	
6.1	SUMMARY	43
6.2	CONCLUSION	43
	REFERENCES	
	APPENDIX	

LIST OF ABBREVIATIONS

CMOS	Complementary Metal Oxide Semiconductor
PMOS	P-channel Metal Oxide Semiconductor
NMOS	N-channel Metal Oxide Semiconductor
VLSI	Very Large Scale Integrated Circuits
CCD	Charge Coupled Device
SSD	Solid State Drive
EDA	Electronic Design Automation
MCA	Memristor Crossbar Array
LTSPICE	Linear Technology Simulation Program with Integrated Circuit Emphasis
MRL	Memristor Ratioed Logic
CVD	Chemical Vapour Deposition
ALD	Atomic Layer Deposition
RRAM	Resistive Random Access Memory
PCB	Printed Circuit Board
NVRAM	Non-Volatile Random Access Memory
SCM	Storage Class Memory

LIST OF FIGURES

FIG No.	TITLE	PAGE No.
1.1	Full Adder Logic Circuit	2
1.2	4-Bit Ripple Carry Adder	3
1.3	Rca With Generate Propagate Bits	4
3.1	Carry Select Adder	18
3.2	Carry Skip Adder	19
3.3	Carry Look Ahead Adder	21
3.4	4-Bit Carry Save Adder	23
3.5	Three Operand Binary Carry Save Adder	24
3.6	Sklansky Adder	25
3.7	Koggle Stone Adder	26
3.8	Brent Kung Adder	27
3.9	Ladner Fischer Adder	28
5.1	Simulation Results for Hybrid Koggle Stone-Brent Kung Adder	35
5.2	Circuit Diagram for Hybrid Brent Kung-Koggle Stone Adder	40
5.3	Rtl Schematic of Hybrid Brent Kung-Koggle Stone Adder	40

LIST OF TABLES

TABLE No.	TITLE	PAGE No.
1.1	Truth Table of Full Adder	3
3.1	Truth Table of Carry Look Ahead Adder	22
5.1	Performance Analysis of All Prefix Adders	41

CHAPTER-1

INTRODUCTION

The ability of VLSI chips to do computations depends largely on their adder circuits, which must be fast and power-efficient. Conventional adders usually work best within a specific input bit width range. However, some of these adders slow down calculations while using less power and taking up less space when the width of the input bits grows. Conversely, some other kinds of adders are more powerful but operate at a faster speed. On the other hand, parallel prefix adders effectively balance power consumption and speed. Our goal in this post is to examine and evaluate a number of traditional adders in addition to parallel prefix adders. We will also contrast how well they work in terms of latency and power consumption, emphasizing the drawbacks of conventional methods. Finally, we will outline and contrast the areas, delays, and power consumption of each parallel prefix adder covered in this paper. In summary, compared to traditional adders, parallel prefix adders provide a favourable trade-off between speed and power usage. We have outlined the benefits and shortcomings of the various adder circuits by carefully evaluating and analyzing them. Additionally, we have included a thorough analysis of the size, latency, and power consumption of each of the parallel prefix adders that were examined. This section will address the full adder, which is the fundamental building block of traditional adders, but not just yet. Later on! Let's discuss a few select adders from the past before moving on to Section II. You know, we have the Carry-lookahead adder, the Carry-bypass adder, and the Ripple-carry adder. There are several problems with these adders; therefore, we should reconsider. The parallel prefix adders then join the fun there! I assure you that we will get into the specifics in Section III! We'll cover the fundamentals, prefix trees, ideas, and gist of parallel prefix adders, including the Kogge-Stone, Brent-Kung, Han-Carlson, and Ladner-Fischer adders. Things really start to heat up in Section IV, don't they? Here, we're running late! The comparison of all the previously discussed parallel prefix adders' performances is the main focus of this section. After delving further into the theoretical assessment, we'll immediately use the Vivado design suite to evaluate them. Of course, the Xilinx Zynq-7000SoC will see some Verilog code. I can hardly contain my excitement.

As previously said, let's start by talking about the basic 1-bit full-adder, which serves as the foundation for many excellent conventional adder circuit designs. Thus, it shouldn't truly cost anything, if you understand what I mean. Thus, this remarkable 1-bit full-adder is, to put it mildly, crucial to all of the conventional methods of designing adder circuits. It basically adds two 1-bit values, calculates their sum, and then executes the result.

1.1 FULL ADDER

A full adder is an addition to the half adder, which can compute the addition of two inputs and the previous carry. The delay in the circuit is 2 times units. A full adder cannot fulfill the requirement of an adder in an ALU.

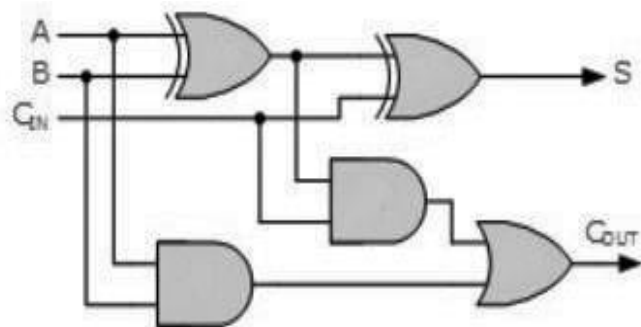


Fig:1.1 : Full Adder Logic Circuit

While one-bit inputs have been included thus far, real-time inputs consist of multiple bits. Let's now examine the circuits that can add values over several bits.

1.2 CONVENTIONAL MULTI-BIT ADDERS

The delay is the most crucial element to consider when evaluating the performance of any arithmetic circuit. Arithmetic circuits are incorporated into the circuit's data flow. The speed at which the data path may be achieved is paramount, with power consumption ranking second in importance. The longest potential delay between any two input and output registers where the inputs and outputs are stored is known as the critical path. This is significant because the critical path makes it simple to analyse and assess adders other than parallel prefix adders. A ripple carry adder is a type of logic circuit where the next most significant full adder is carried in after each full adder is carried out. The reason it's named a ripple carry adder is that every carry bit ripples into the subsequent step.

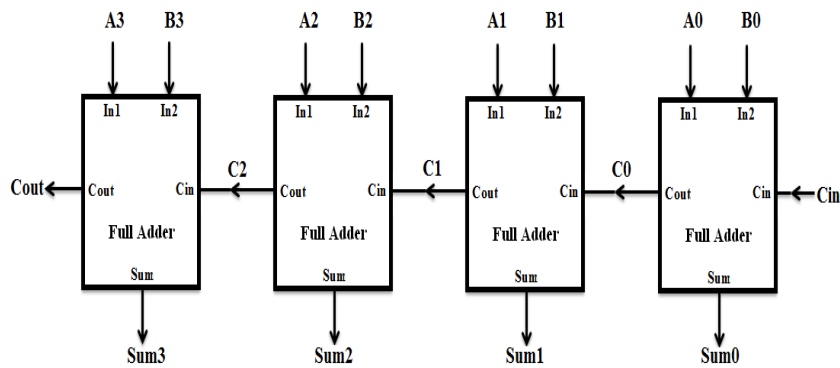


Fig:1.2 : 4-Bit Ripple Carry Adder

An N-bit integer can be added by cascading multiple complete adder circuits in parallel. There must be N complete adder circuits for every N-bit parallel adder. A ripple carry adder is a type of logic circuit where the next most significant full adder is carried in after each full adder is carried out. The reason it's named a ripple carry adder is that every carry bit ripples into the subsequent step.

The sum and carry out bits of each half adder stage in a ripple carry adder are invalid until the carry in of that stage happens. This is caused by propagation delays within the logic circuitry. The amount of time that passes between applying an input and the associated output occurring is known as the propagation delay. Think about a NOT gate. Its output will be "1" when the input is "0," and vice versa.

1.3 GENERAL PROPAGATE LOGIC

Table 1.1: Truth table for a full adder with a different perspective including the generate and Propagate bits.

A	B	C_{in}	C_{out}	S	G	D	P
0	0	0	0	0	0	1	0
0	0	1	0	1	0	1	0
0	1	0	0	1	0	0	1
0	1	1	1	0	0	0	1
1	0	0	0	1	0	0	1
1	0	1	1	0	0	0	1
1	1	0	1	0	1	0	0
1	1	1	1	1	1	0	0

The truth table for a complete adder, viewed from a different angle, is shown above. Designing quicker adders starts with this new method. Here, we're using G (Generate), P (Propagate), and D (Delete) to represent sum and cout. Because C is intimately linked to these three bits, they define the execution and are introduced. The following is the definition of the mutually exclusive variables G, P, and D.

Generate: Regardless of Cin, Cout is 1. $G = A \text{ and } B$ Propagate: Cout receives a copy of Cin. $P = A \oplus B$ Delete: Regardless of Cin, Cout is 0. $D = (\sim A) \& (\sim B)$ $G/(P \& C_{in}) = C$: Cout (D is absent from the expression since Cout is 0 when G and P are also 0. At this point, $D = 1$) and $\text{Sum} = A \oplus B \oplus C_{in}$ Furthermore take note of the G, P, and D's independence from Cin. This indicates that these bits may be computed directly using the operand bits A and B that are provided, meaning that our adder is no longer reliant on the Cout from the previous stage, i.e. Cin for the current stage.

Actually, within Tgp (Time to calculate G and P) of the operation's beginning, all Gs and Ps are computed concurrently. It should be noted that because the generate-propagate bits are computed simultaneously for every bit location, the coefficient of Tgp (Delay due to generate-propagate logic) is 1. After then, the process is comparable to a ripple carry adder. The propagation delay is therefore computed as follows: $T_{gp} + (N-1) \cdot T_{pd} = T_{pd} + T_{sum}$, where T_{carry} is the time it takes to calculate each bit's Cout $T_{sum} = \text{Time needed to calculate the sum of the final bit}$. Given that the T_{carry} term would therefore be dominant, the calculation demonstrates that this has no appreciable impact on longer word lengths.

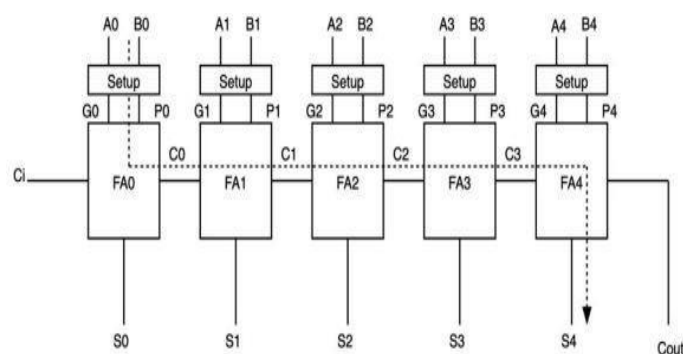


Fig:1.3 : RCA with generate-propagate bits

1.4 FEATURES OF VLSI

- A. VLSI Design Overview.
- B. VLSI Design Tools.
- C. VLSI Design Styles.
- D. Challenges in VLSI Design
- E. Low Power Consumption

1.4.1 VLSI Design Overview

Very Large Scale Integration (VLSI) design involves the creation of integrated circuits that incorporate thousands or millions of transistors on a single chip. The design process encompasses several stages, starting with specification and architecture definition, followed by Register-Transfer Level (RTL) design using Hardware Description Languages (HDL) like VHDL or Verilog. Functional verification ensures that the design meets requirements, and synthesis transforms the RTL description into a gate-level netlist. Subsequent steps involve floor planning, place and route, layout design, and physical verification to create an actual chip layout. The process is facilitated by Electronic Design Automation (EDA) tools such as Cadence and Synopsys, with various design styles like full-custom, standard cell, gate array, and Field-Programmable Gate Array (FPGA) offering different trade-offs in customization and design effort.

1.4.2 VLSI Design Tools

Electronic Design Automation (EDA) tools are indispensable in VLSI design, providing a suite of software for different design stages. Leading EDA tools, including Cadence, Synopsys, and Mentor Graphics, offer functionalities for synthesis, simulation, place and route, and physical verification. These tools streamline the complex design process, allowing engineers to design and analyze integrated circuits efficiently.

1.4.3 VLSI Design Styles

VLSI design embraces various styles to cater to different design requirements. Full-custom design involves meticulous manual transistor and interconnect design, offering maximum control but demanding substantial effort. Standard cell design strikes a balance by utilizing pre-designed and characterized standard cells. Gate array design uses predefined logic blocks with customizable interconnect, providing a compromise between customization and design effort. Field-Programmable Gate Arrays (FPGAs) offer configurability after manufacturing, facilitating prototyping and rapid design iteration.

1.4.4 Challenges in VLSI Design

VLSI designers face several challenges in creating high-performance and reliable integrated circuits. Managing power consumption is critical, especially in mobile devices. Achieving accurate timing closure to meet specified clock frequencies poses a significant challenge. Design for Manufacturability (DFM) ensures that the design is manufacturable within fabrication process constraints. Technology scaling introduces challenges related to leakage power, variability, and reliability. Ensuring defect-free chips through rigorous testing remains a paramount challenge in VLSI design.

1.4.5 Low Power Consumption

Reducing power consumption in Very Large Scale Integration (VLSI) design is imperative for the development of energy-efficient electronic devices. To achieve this, designers employ various techniques such as power gating, selectively turning off power to idle sections, and clock gating, dynamically controlling the clock signal to specific blocks. Voltage scaling minimizes dynamic power by lowering the operating voltage, though careful consideration of circuit reliability is necessary. Implementing multiple power domains enables fine-grained control over power states, allowing sections of the chip to operate at different voltages or be powered down when inactive. Energy-efficient logic styles, like adiabatic or reversible logic, contribute to lower power consumption, as do data compression techniques that reduce the volume of data to be processed. Dynamic Voltage and Frequency Scaling (DVFS) adjusts the operating frequency and voltage based on workload,

while low-power design libraries and sleep modes further enhance overall power efficiency. These strategies collectively address both static and dynamic power consumption, aligning VLSI designs with the growing demand for energy-conscious electronic systems.

1.5 ADVANTAGES OF VLSI USING CMOS

- A. Less Area Occupation
- B. Low Power Dissipation
- C. Efficient Architectures
- D. Minimum Delay
- E. Simple Structure

1.5.1 Low Area Occupation

Achieving low area in Very Large Scale Integration (VLSI) design is pivotal for creating compact and cost-effective integrated circuits. This goal is pursued through various strategies, starting with compact circuit design that prioritizes efficient architectures and logic styles to inherently occupy less space. Leveraging technology scaling, advancements in semiconductor manufacturing contribute to smaller feature sizes, enabling the creation of denser and more space-efficient chip layouts. Embracing a design hierarchy facilitates modularization, allowing the reuse of smaller, compact blocks throughout the chip design. Customized cell libraries tailored to specific design requirements play a crucial role in optimizing area consumption, providing a more tailored and efficient set of components compared to generic libraries. Additionally, employing logic synthesis optimization techniques further refines the design, ensuring that logic gates and elements are arranged and interconnected in an optimal manner, ultimately contributing to a minimized chip area. These collective strategies address the imperative need for compactness in VLSI design, accommodating the constraints of modern electronics where size, efficiency, and cost are paramount considerations.

1.5.2 Low Power Dissipation

Minimizing power dissipation is a paramount goal in Very Large Scale Integration (VLSI) design, crucial for enhancing the energy efficiency of integrated circuits. To

achieve lower power dissipation, designers deploy various advanced techniques. Clock gating, for instance, dynamically controls the clock signal to specific sections of the chip, shutting off the clock to inactive blocks and thereby reducing dynamic power consumption. Power gating, on the other hand, involves selectively turning off power to specific sections of the chip during periods of inactivity, effectively minimizing static power consumption. Voltage scaling is employed to lower the operating voltage, directly reducing dynamic power consumption; however, this necessitates careful consideration of circuit reliability. Furthermore, dividing the chip into multiple power domains allows different sections to operate at distinct voltages or be completely powered down when not in use, providing fine-grained control over power dissipation. These combined strategies address both dynamic and static power concerns, contributing to the creation of energy-efficient VLSI designs that align with contemporary demands for reduced power consumption in electronic devices.

1.5.3 Less Area Efficient Architectures in VLSI

Achieving high-speed performance in Very Large Scale Integration (VLSI) design is a fundamental pursuit, essential for meeting the increasing demands of modern electronic systems. Designers employ various strategies to optimize speed, starting with the exploitation of parallelism by breaking down tasks into smaller parallel operations, enhancing processing speed through techniques like pipelining and parallel processing architectures. Increasing the clock frequency is another approach to accelerate data processing, though it necessitates a careful balance with considerations for power consumption and heat dissipation. Strategies such as reducing interconnect length and complexity, advancing semiconductor manufacturing through technology scaling, designing low-latency memory structures, and employing optimized logic synthesis contribute to minimizing delays and improving overall system speed. Additionally, efficient clock distribution networks, specialized high-speed libraries, and advanced pipelining techniques are integral for ensuring synchronized and high-speed operation. Critical path analysis identifies and optimizes the longest paths in the design, determining the maximum achievable speed. By integrating these methodologies, VLSI designers can create integrated circuits that operate at high frequencies, meeting the performance

requirements of contemporary applications in areas such as data processing and high-performance computing.

1.5.4 Less Delay in VLSI

Minimizing delay is a pivotal goal in Very Large Scale Integration (VLSI) design, aimed at improving the responsiveness and overall performance of integrated circuits. Designers implement various strategies to reduce delays throughout the design process. At the gate level, advanced optimization techniques, such as Boolean logic minimization and gate merging, are employed to simplify logic circuits and decrease gate delays. Technological advancements in semiconductor manufacturing, achieved through technology scaling, contribute to shorter gate delays and interconnect lengths, thereby reducing signal propagation delays. Efficient clock distribution with minimized clock skew ensures synchronous signals, effectively reducing delays associated with synchronization. The use of pipelining breaks down complex operations into smaller stages, enabling concurrent processing and reducing effective delays. Parallel processing, exploiting simultaneous execution of multiple operations, further contributes to delay reduction. Optimization of interconnects minimizes the length and complexity of connections, reducing signal propagation delays. Designing low-latency memory structures ensures quick data retrieval, minimizing delays in accessing critical information. Overall, these strategies, implemented throughout the VLSI design process, collectively contribute to achieving lower delays, thereby enhancing the speed and efficiency of integrated circuits.

1.5.5 Simple Architecture

Simplicity in Very Large Scale Integration (VLSI) architecture is a guiding principle that fosters efficiency and reliability in the design of integrated circuits. A simple VLSI architecture is characterized by reduced logic complexity, achieved through the application of optimization techniques and streamlined Boolean expressions to minimize the number of gates and logic elements. Hierarchical design is instrumental, breaking down the overall system into smaller, modular blocks that can be independently designed, thus simplifying the overall architecture. Regularity and symmetry in structures contribute to simplicity, promoting efficient resource utilization. Additionally, minimizing interconnect complexity is crucial, as a straightforward and

direct interconnection scheme simplifies the routing and reduces signal propagation delays. In essence, a simple VLSI architecture balances functionality with uncomplicated design, facilitating ease of implementation and enhancing the overall robustness of integrated circuits.

1.6 DISADVANTAGES OF USING VLSI DESIGN

- A. Complexity and Design Challenges
- B. High Initial Costs
- C. Power Consumption
- D. Design Complexity

1.6.1 Complexity and Design Challenges

The complexity and design challenges in Very Large Scale Integration (VLSI) stem from the integration of millions or billions of transistors onto a single chip. This intricacy poses significant hurdles for designers and engineers, necessitating advanced tools and methodologies for successful implementation. Managing the intricate relationships between components, ensuring correct functionality, and optimizing for performance while minimizing power consumption are formidable tasks. The complexity extends to the verification process, where confirming the correct interaction of numerous components and identifying potential defects becomes a time-consuming and resource-intensive endeavor. The rapid evolution of VLSI technology compounds these challenges, requiring continuous adaptation to new fabrication processes and design methodologies. Overall, the complexity inherent in VLSI design demands a high level of expertise, sophisticated tools, and a meticulous approach to ensure the successful creation of integrated circuits with the desired functionality and performance.

1.6.2 High Initial Costs

The high initial costs associated with Very Large Scale Integration (VLSI) design and fabrication constitute a significant barrier and challenge within the semiconductor industry. The development of VLSI circuits involves substantial upfront expenditures, primarily attributed to the complex and sophisticated nature of the design process. Initial costs encompass various aspects, including the creation

of photomasks essential for semiconductor fabrication, prototyping, and the establishment of fabrication facilities. These costs are particularly burdensome for smaller companies, startups, or research initiatives aiming to enter the VLSI domain. The need for state-of-the-art equipment, cleanroom facilities, and skilled personnel further contributes to the financial burden. Moreover, given the rapidly evolving nature of VLSI technology, continuous investment is essential to keep pace with advancements, adding to the overall expense. While economies of scale can be achieved through large-scale production, the initial financial outlay remains a notable challenge and consideration in the decision-making processes of organizations venturing into VLSI design.

1.6.3 Power Consumption

Power consumption is a critical consideration in Very Large Scale Integration (VLSI) design, reflecting the amount of electrical energy consumed by integrated circuits during operation. The power dissipated in a VLSI chip can be broadly categorized into two types: dynamic power and static power. Dynamic power is associated with the switching of transistors and is directly proportional to the operating frequency and the square of the operating voltage. Therefore, increasing the clock frequency or operating voltage results in higher dynamic power consumption. Static power, on the other hand, is attributed to leakage currents that persist even when the transistors are in a static state. Reducing static power involves minimizing leakage through advanced fabrication technologies and power gating techniques. Balancing these two aspects is crucial, as lowering the operating voltage or frequency to reduce dynamic power may lead to an increase in static power. Techniques such as clock gating, which involves turning off the clock signal to inactive sections of the chip, and voltage scaling, where the operating voltage is reduced to lower power consumption, are employed to optimize power efficiency. Additionally, advancements in low-power design libraries, the use of energy-efficient logic styles, and the exploration of emerging technologies contribute to addressing power consumption challenges in VLSI design. As energy efficiency becomes increasingly vital in portable and battery-operated devices, minimizing power consumption remains a central focus in the ongoing development of VLSI technology.

1.6.4 Design Complexity

Heat dissipation challenges represent a significant concern in Very Large Scale Integration (VLSI) design, stemming from the densely packed nature of modern integrated circuits. As the number of transistors on a chip increases, the power density also rises, leading to elevated temperatures. Efficient heat dissipation is essential to prevent overheating, which can compromise the performance, reliability, and lifespan of the integrated circuit. The challenge lies in managing this heat within the confined space of the chip while ensuring that no component exceeds its specified operating temperature range. VLSI designers employ various techniques to address heat dissipation, including the use of heat sinks, fans, and thermal management materials. Additionally, optimizing the placement of components, improving the thermal conductivity of materials, and exploring advanced cooling solutions, such as liquid cooling, are strategies to mitigate heat-related challenges. The constant push for higher performance and integration in VLSI design amplifies the heat dissipation challenge, requiring a delicate balance between performance optimization and thermal management to ensure the reliable and sustained operation of integrated circuits.

CHAPTER-2

LITERATURE SURVEY

2.1 GENERAL SURVEY

In the realm of digital VLSI design, the exploration of adder architectures has been a topic of significant interest due to their pivotal role in arithmetic operations. Traditional adder designs, such as the Brent-Kung, Kogge-Stone, Han-Carlson, and Ladner-Fischer adders, have been extensively studied, revealing distinct trade-offs in terms of power consumption, delay, and hardware utilization. However, with the increasing demand for optimized adder designs, the introduction of innovative architectures like the Hybrid BK-KS Adder has garnered attention. This survey delves into the performance evaluation and comparative analysis of traditional adders alongside the Hybrid BK-KS Adder. Through methodologies encompassing design, simulation, synthesis, and evaluation using tools like the Vivado Design Suite, this survey aims to identify the strengths and weaknesses of each adder architecture, highlighting the potential of the Hybrid BK-KS Adder in addressing key challenges and advancing adder design paradigms.

2.1.1 Morgana M. A. Rosa, Eduardo Costa, Rafael Soares, Sergio Bampi .” Accuracy, Delay and Area-Driven Evaluation of Lower-Part Approximate Parallel Prefix Adder” 2023 30th IEEE International Conference on Electronics, Circuits and Systems (ICECS).

This work proposes an accuracy-driven evaluation model of an approximate adder (AxA) called an approximate parallel prefix adder (AxPPA). It explores the significance of approximate computing (AxC) in optimizing performance and energy efficiency in computer systems by accepting minor accuracy trade-offs. We examine the AxPPA through a meticulous error analysis model and an automated analysis framework that comprehensively evaluates area and delay based on the fundamental equations of parallel prefix adders (PPAs). The choice of approximation method directly impacts computational accuracy, resulting in variations in the number of approximated bits and carry chain length for a given tolerance level. Conducting a thorough comparison of accuracy trade-offs requires a unique design and simulation for each unit and approximation level, which can be time-consuming, particularly in larger circuits. Therefore, the proposed approach facilitates the exploration of larger circuit design space encompassing multiple AxAs, significantly

minimizing time investment. In our experiments, the accuracy-driven evaluation of the lower part approximation method in AxA took 71.268 milliseconds for 1 million random inputs, with a K-bits approximation ranging from 1 to 16. The results indicate that AxPPA exhibits a minor error magnitude among the explored literature AxAs, being approximately 50.97% more accurate on average.

2.1.2 Kiran V, E M Savi Teja,” A Novel Approach to Braun Multiplier Design Utilizing High-Speed Parallel Prefix Adder for Low Power Applications” 2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS).

This project develops an innovative Braun Multiplier design to address power consumption and chip area challenges. Integrating a high-speed parallel prefix adder enhances computational speed by leveraging parallel processing capabilities. The synergy between the Braun Multiplier and the adder yields a solution optimizing performance, power efficiency, and chip area usage. Meticulous design and engineering yield notable improvements in power efficiency and chip area utilization. The project drives advancements in digital circuit design and optimization methodologies, achieving a 16% area reduction and 18.2% lower power consumption via the FS-GDI technique, while achieving full-swing output voltage, distinguishing it from GDI and M-GDI techniques.

2.1.3 Dr B N Shashikala, Pooja R, Pavan Prahallad, Ramya S K,Ramyashree M V, “Design and implementation of Kogge-stone, Sparse Kogge-stone and Spanning tree adder” 2023 International Conference on Smart Systems for applications in Electrical Sciences (ICSSES).

Many Very Large Scale Integration (VLSI) circuits typically employ Parallel Prefix Adders (PPA). Power, size, and speed are the key constraints for designing an adder circuit because there is a trade-off between them. In-depth research is still being done on minimising the power delay and thus improving adder performance because the binary adder is an essential part of the majority of digital circuit designs, as well as Digital Signal Processors (DSP) and microchip knowledge path units. The Kogge-stone (KSA), Sparse Kogge-stone (SKS), and Spanning tree adders are examined in this research along with current adder designs. Their performances are compared in light of various design constraints to discover which adder performs the best among them. The survey results conclude that Spanning Tree adder

achieves a good compromise between power consumption and area (Uses 128 LUTs out of 66560) and Kogge Stone Adder achieves a low worst-case delay (29.638 ns) and good scalability.

2.1.4 Pixia Guo, Guangjun Xie, Xiaoyue Chen, Jie Han, Yongqiang Zhang, " A Variable Latency Ling Adder Based on Brent-Kung ParallelPrefix Topology" 2023 IEEE 23rd International Conference on Nanotechnology (IEEE-NANO 2023).

For emerging applications, such as deep learning, design complexity can be reduced to lower hardware costs by releasing the strict requirement of computing accuracy. In this paper, a variable latency Ling adder (VLLA) with reduced propagation delay is proposed to produce exact results in most cases through a speculative operation. When an error is detected, the error detection and correction modules are activated to generate exact results. All circuits are synthesized in the Synopsys Design Compiler with a TSMC 65 nm standard cell library and evaluated for the error probability in MATLAB. Experimental results show that the proposed VLLA achieves reductions in power and smallest average delay by up to 17% and 23%, respectively, compared to the original exact Ling adder (LA). In addition, the proposed VLLA achieves reductions of 8%, 18%, 11.4%, and 7.6% in power, smallest average delay, area average-delay product, and power average-delay product, compared to an existing variable latency adder (VLA).

2.1.5 Apostolos Stefanidis, Ioanna Zoumpoulidou, Dionysios Filippas, Giorgos Dimitrakopoulos and Georgios Ch. Sirakoulis, "Synthesis of Approximate Parallel-Prefix Adders" IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 31, NO. 11, NOVEMBER 2023.

— Approximate computation has evolved recently as a viable alternative for maximizing energy efficiency. One aspect of approximate computing involves the design of hardware units that return a sufficiently accurate result for the examined occasion, rather than computing an accurate result. As long as the hardware units are allowed to compute approximately, they can be designed with multiple new ways. In this work, we focus on the synthesis of approximate parallel-prefix adders. Instead of exploring specific architectures, as done by state-of-the-art approaches, the introduced synthesizer can produce every solution that meets the designer's criteria, resulting in adders with various delay, area, and error tradeoffs. This

automatic design space exploration allows approaching, in several cases, optimal solutions that could have not been designed with any other known parallel-prefix architecture. The synthesized adders, when compared with state-of-the-art adders, achieve 27%–36% better error frequency (EF) on average for random inputs and improve image quality metrics by 8%-42% for image filtering. These results are achieved with the proposed adders requiring the same or marginally more hardware area or energy. On the contrary, in split-accuracy configurations, more than 30% of hardware area/energy can be saved for the same classification accuracy for a neural network application.

2.2 OVERVIEW OF EXISTING RESEARCH

Adder architectures play a crucial role in digital VLSI design, serving as fundamental components in arithmetic units. Over the years, extensive research has been conducted to optimize adder designs in terms of power consumption, delay, area utilization, and overall performance. Traditional adder architectures, including the Brent-Kung, Kogge-Stone, Han-Carlson, and Ladner-Fischer adders, have been extensively studied and analyzed in various research works.

The Brent-Kung Adder, known for its regular structure and efficient carry propagation, has been a subject of interest in low-power VLSI designs where delay is not the primary concern. Similarly, the Kogge-Stone Adder, with its parallel prefix structure and efficient carry lookahead, has been favored for high-speed applications, particularly in DSPs (Digital Signal Processors).

Recent research has also explored hybrid approaches in adder design, aiming to leverage the strengths of multiple architectures while mitigating their limitations. The Hybrid BK-KS Adder, introduced as a novel architecture, combines features from both Brent-Kung and Kogge-Stone designs. This innovative approach seeks to strike a balance between power efficiency and performance, offering a versatile solution for various computational requirements. These studies have provided valuable insights into the trade-offs and design considerations essential for advancing adder design paradigms in modern digital systems.

CHAPTER-3

AIM AND IMPLEMENTATION OF EXISTING SYSTEM

3.1 AIM OF THE PROJECT

The aim of this project is to comprehensively evaluate and compare traditional adder architectures, including the Brent-Kung, Kogge-Stone, Han-Carlson, and Ladner-Fischer adders, with a focus on power consumption, delay, area utilization, and overall performance. Additionally, the project aims to introduce and assess the Hybrid BK-KS Adder, a novel architecture that combines features from both Brent-Kung and Kogge-Stone designs. Through design, simulation, synthesis, and evaluation using tools like the Vivado Design Suite, the project seeks to identify the strengths and weaknesses of each adder architecture and highlight the potential benefits of adopting the Hybrid BK-KS Adder in modern digital systems. The ultimate goal is to contribute to the advancement of adder design paradigms by providing insights into innovative approaches that offer improved performance and efficiency in various computational applications.

3.2 CONVENTIONAL ADDERS

The delay is the most crucial element to consider when evaluating the performance of any arithmetic circuit. Arithmetic circuits are incorporated into the circuit's data flow. The speed at which the data path may be achieved is paramount, with power consumption ranking second in importance. The longest potential delay between any two input and output registers where the inputs and outputs are stored is known as the critical path. This is significant because the critical path makes it simple to analyze and assess adders other than parallel prefix adders.

3.2.1 Carry Select Adder

The carry-select adder generally consists of 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 to perform the calculation twice, one time with the assumption of the carry-in being zero and the other assuming it will be one. After the two results are calculated, the correct sum, as well as the correct carry-out, is then selected with the multiplexer once the correct carry-in is known.

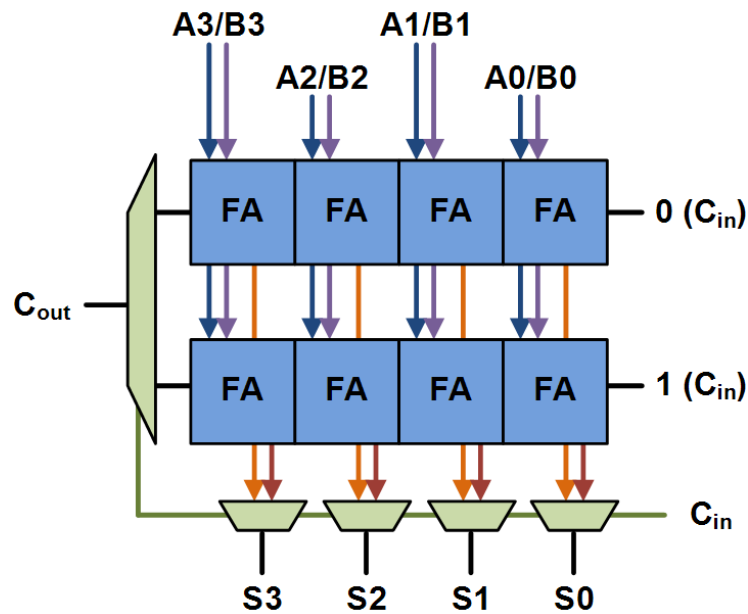


Fig:3.1 : Carry Select adder

A Carry Select Adder (CSA) is a parallel adder designed to enhance the speed of binary addition by leveraging Carry Look-Ahead (CLA) logic. Unlike conventional ripple-carry adders, the CSA operates in parallel on multiple pairs of bits, precomputing carry signals to minimize the critical path delay associated with carry propagation. The key feature lies in the efficient use of multiplexers to select the appropriate carry result based on the actual carry-in, hence the name "Carry Select." This parallelized approach significantly improves performance and reduces propagation delay, making CSAs advantageous in applications demanding high-speed addition, such as arithmetic units in microprocessors and digital signal processors. However, it's important to note that CSAs may incur higher hardware overhead and area compared to simpler adders. The key feature of a Carry Select Adder is its use of Carry Look-Ahead (CLA) logic.

Instead of waiting for the carry bit to propagate through the adder, the CSA precomputes the carry signals for different pairs of bits. This precomputation significantly reduces the critical path delay associated with carry propagation. In a Carry Select Adder, there are multiple carry chains, and a multiplexer is used to select the correct carry result based on the actual carry-in. This is where the "select" in Carry Select Adder comes from. The primary advantage of a Carry Select Adder is its improved speed compared to ripple-carry adders. By parallelizing the addition

process and efficiently handling carries, it achieves a lower propagation delay and higher performance. While Carry Select Adders offer faster addition, they come with some trade-offs. They may require more hardware resources and exhibit higher area overhead compared to simpler adders like ripple-carry adders. In summary, a Carry Select Adder is a parallel adder that leverages Carry Look-Ahead logic to reduce the critical path delay associated with carry propagation. This makes it particularly useful in applications where high-speed addition is crucial, such as in arithmetic units of microprocessors and digital signal processors.

3.2.2 Carry Skip Adder

A Carry Skip Adder (also known as a Carry Bypass Adder or a Carry-Bypass Adder) is a type of parallel adder designed to overcome the delays associated with carry propagation in binary addition. It falls under the category of adders that aim to improve the speed of addition processes. The fundamental idea behind a Carry Skip Adder is to divide the adder into smaller blocks, each capable of processing a subset of the bits, and then introducing additional logic to "skip" the carry propagation across blocks when it is not necessary.

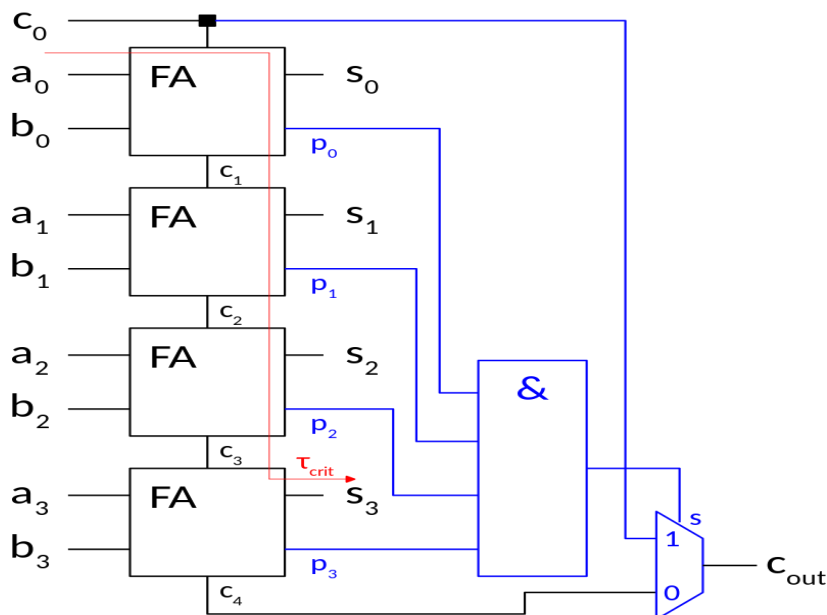


Fig:3.2 : Carry Skip Adder

The architecture typically consists of a series of interconnected, smaller full adders grouped into blocks. These blocks operate on distinct sections of the binary numbers being added. If the carry generated within a block is determined to have no impact on subsequent blocks (i.e., no bits to the left are set to 1), a carry signal can be skipped, bypassing the delay associated with the carry chain. This mechanism allows for parallel processing within each block while minimizing the overall critical path delay.

Carry Skip Adders are particularly useful in applications where high-speed addition is crucial, such as in arithmetic units of microprocessors and digital signal processors. While they offer advantages in terms of speed, they may introduce additional hardware complexity and area overhead compared to simpler adders like ripple-carry adders. The trade-offs involve finding the right balance between speed improvement and increased hardware complexity. Carry Skip Adders are commonly employed in critical paths of arithmetic circuits in microprocessors, digital signal processors (DSPs), and other applications where fast addition is essential for overall system performance.

The critical innovation in a Carry Skip Adder is the introduction of carry skip logic. This logic allows the carry generated within a block to be "skipped" or bypassed if it is determined that the carry will not affect subsequent blocks. This is determined by examining the bits to the left of the current block and checking if any of them are set to 1. While Carry Skip Adders provide a notable improvement in speed, they come with trade-offs. The additional logic required for carry skipping introduces complexity, and there's an associated increase in hardware overhead. The optimal design balances the advantages in speed against the added complexity. In summary, a Carry Skip Adder is a parallel adder designed to expedite binary addition by selectively skipping carry signals, reducing the overall propagation delay. Its application is prominent in scenarios where high-speed addition is critical, despite the associated increase in hardware complexity.

3.2.3 Carry Look Ahead Adder

The adder produce carry propagation delay while performing other arithmetic operations like multiplication and divisions as it uses several additions or subtraction steps. This is a major problem for the adder and hence improving the speed of

addition will improve the speed of all other arithmetic operations. Hence reducing the carry propagation delay of adders is of great importance. There are different logic design approaches that have been employed to overcome the carry propagation problem. One widely used approach is to employ a carry look-ahead which solves this problem by calculating the carry signals in advance, based on the input signals. This type of adder circuit is called a carry look-ahead adder. In ripple carry adders, for each adder block, the two bits that are to be added are available instantly. However, each adder block waits for the carry to arrive from its previous block. So, it is not possible to generate the sum and carry of any block until the input carry is known. The i^{th} block waits for the $i-1^{\text{th}}$ block to produce its carry. So there will be a considerable time delay which is carry propagation delay.

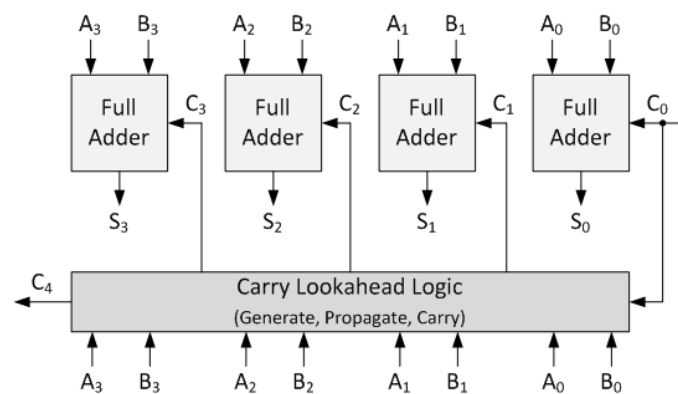


Fig:3.3 : Carry Look Ahead Adder

The primary advantage of Carry Look-Ahead Adders is their ability to process multiple bit positions in parallel. This parallelism significantly reduces the critical path delay associated with carry propagation, leading to faster addition. The primary advantage of Carry Look-Ahead Adders is their ability to process multiple bit positions in parallel. This parallelism significantly reduces the critical path delay associated with carry propagation, leading to faster addition. The primary advantage of Carry Look-Ahead Adders is their ability to process multiple bit positions in parallel. This parallelism significantly reduces the critical path delay associated with carry propagation, leading to faster addition.

Table 3.1: Truth Table of Carry Look Ahead Adder

A	B	C	C +1	Condition
0	0	0	0	No Carry Generate
0	0	1	0	
0	1	0	0	
0	1	1	1	No Carry Propagate
1	0	0	0	
1	0	1	1	
1	1	0	1	Carry Generate
1	1	1	1	

The primary advantage of Carry Look-Ahead Adders is their ability to process multiple bit positions in parallel. This parallelism significantly reduces the critical path delay associated with carry propagation, leading to faster addition. The CLA is often organized in a binary tree structure, where each node represents a stage of addition. The carry signals are generated and propagated upwards through the tree, facilitating efficient computation. While Carry Look-Ahead Adders provide improved speed compared to ripple-carry adders, they may require more hardware resources due to the additional logic for carry look-ahead. The trade-offs involve finding the right balance between speed and hardware complexity. Carry Look-Ahead Adders find applications in scenarios where high-speed addition is crucial, such as in arithmetic units of microprocessors and digital signal processors. They are effective in reducing the critical path delay associated with carry propagation, contributing to enhanced overall system performance.

3.2.4 Carry Save Adder

In arithmetic & computer circuits, adders play an important role that can be utilized within the signaling methodologies. Adders are normally found in many building blocks of digital signal processing & microprocessor chips. There are different types of adders available like carry skip adder, CLA adder, RCA or ripple carry adder, and many more but, the carry save adder is a high-speed and low-cost adder with less

latency. In array processing, division & multiplication, multi-operand addition is frequently encountered. To add many numbers in place of two together, several powerful adders are necessary. As a result, it avoids speed-up computation & time-consuming carry propagation. This article discusses an overview of a carry save adder or CSA. To add many numbers in place of two together, several powerful adders are necessary. A carry-save adder or CSA is a high-speed multi-operand type adder and this adder is perfect for adding numerous operands together. As a result, it avoids speed-up computation & time-consuming carry propagation. This article discusses an overview of a carry save adder or CSA.

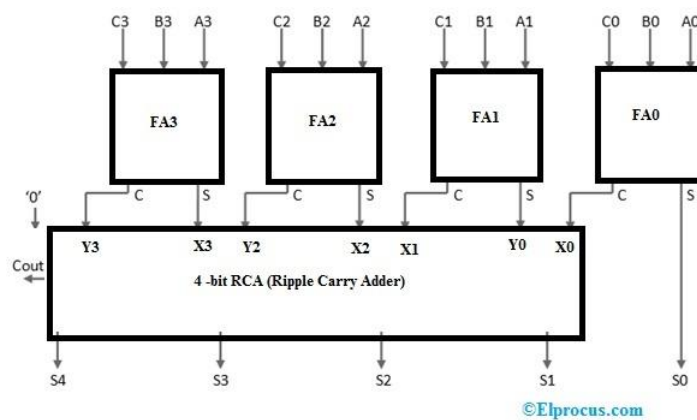


Fig:3.4 : 4-Bit Carry Save Adder

The three-operand binary addition can be carried out either by using two two-operand adders or one three-operand adder. Carry-save adder (CS3A) is the area-efficient and widely adopted technique to perform the three-operand binary addition in the modular arithmetic used in cryptography algorithms and PRBG methods. However, the longer carry propagation delay in the ripple-carry stage of CS3A seriously influences the performance of the MDCLCG and other cryptography architectures on IoT based hardware devices. In order to shorten the critical path delay, a parallel prefixed two-operand adder such as Han-Carlson (HCA) can also be used for three-operand binary addition. It reduces the critical path delay in the order of $O(\log_2 n)$ but increases the area in the order of $O(n \log_2 n)$ [15]. Therefore, it is necessary to develop an efficient VLSI architecture to carry out the fast three-operand binary addition with minimum hardware resources. Hence, a new high-speed area-efficient adder technique is proposed using pre-compute bitwise addition followed by carry-prefix computation logic to perform the three-operand

addition in this paper that consumes considerably less gate area while minimizing the propagation delay in comparison to the HCA-based three-operand adder (HC3A). Furthermore, the proposed adder architecture is implemented with the Verilog HDL, and then synthesized with commercial available 32nm CMOS technology library. Also, the area-delay and power-delay products of the proposed adder technique are measured and compared with respect to the existing CS3A and HC3A three-operand adder techniques.

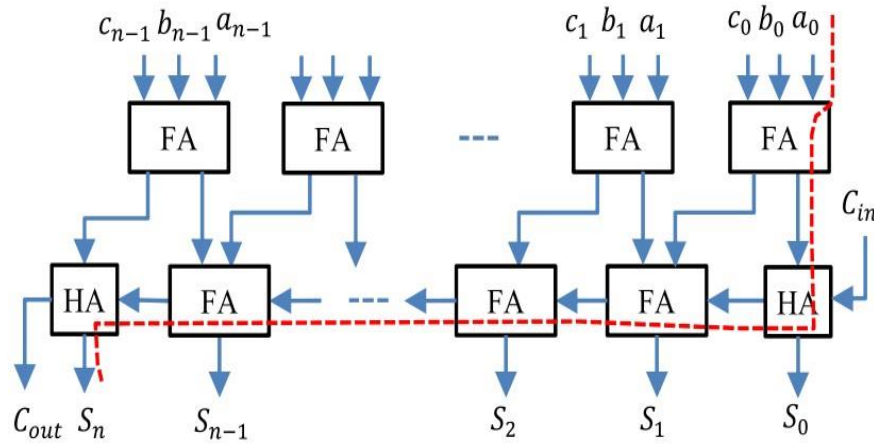


Fig:3.5 : Three Operand Binary Carry save Adder

3.3 PARALLEL PREFIX ADDER

Parallel prefix adders, also known as carry-select adders, are a class of digital circuits designed to efficiently perform binary addition by computing carry values in parallel. Unlike traditional ripple-carry adders where the carry ripples through each bit sequentially, parallel prefix adders process multiple bits simultaneously. They are favored in applications requiring fast arithmetic operations. One popular type of parallel prefix adder is the Kogge-Stone adder, known for its regular and symmetric structure. The design employs a binary tree structure, allowing for the parallel computation of carry values. This architecture reduces the critical path delay, enhancing overall speed. However, it's important to note that parallel prefix adders often come with increased hardware complexity and power consumption compared to simpler adders. These adders leverage concepts like prefix sum and prefix product networks, where each bit's output depends on the sum or product of certain inputs. This parallel processing of carry values enables faster addition of binary

numbers. The Brent-Kung adder is another example of a parallel prefix adder that uses a balanced binary tree structure to optimize speed and regularity. While parallel prefix adders offer advantages in terms of speed, they may be more resource-intensive in terms of hardware complexity and power consumption. The choice of a specific parallel prefix adder depends on the application's requirements, considering factors such as speed, area constraints, and available resources.

3.3.1 Sklansky Adder

The Sklansky adder, also known as the divide-and-conquer tree, was proposed by Sklansky in 1960. It is a type of parallel prefix adder that employs a divide-and-conquer strategy to achieve minimum depth in the prefix network. This strategy involves breaking down the addition problem into smaller sub-problems, solving them independently, and then combining the results. The Sklansky adder, in its conditional sum addition logic for prefix addition, achieves this with a minimal depth prefix structure. However, there are trade-offs associated with the Sklansky adder. One notable drawback is the increased fan-out for certain computation nodes, especially along the critical path. This substantial increase in fan-out, particularly along the critical path, results in a significant amount of latency. The consequence of this elevated fan-out is a degradation in performance, especially when dealing with a large number of bits in the adder. As the number of bits increases, the latency introduced by the fan-out becomes more pronounced, impacting the overall speed and efficiency of the Sklansky adder.

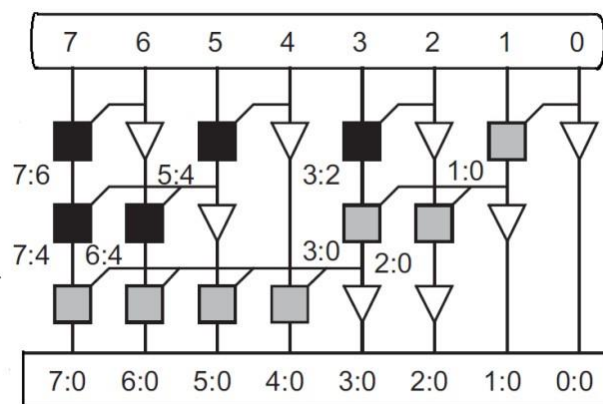


Fig:3.6 : Sklansky Adder

3.3.2 Kogge Stone Adder

The Kogge-Stone adder algorithm, introduced by Kogge and Stone in 1973, represents a significant advancement in the design of parallel prefix adders. This algorithm is known for achieving both optimal depth (logarithmic in terms of the number of bits, denoted as N) and low fan-out. The concept of a prefix adder with controlled fan-out of two at each stage minimizes the latency associated with carry propagation, resulting in improved performance. However, the Kogge-Stone adder does have some trade-offs. While it achieves optimal depth and low fan-out, the circuit realizations can be massively complex, involving a large number of interconnects. The adders implemented using this technique exhibit a regular layout with a controlled fan-out of two, ensuring a balanced and efficient structure. One notable characteristic of the Kogge-Stone tree is that it achieves $\log_2 N$ stages, where N is the number of bits. At each stage, the fan-out is limited to two, maintaining a controlled and consistent structure. Despite these advantages, the Kogge-Stone tree introduces challenges related to interconnects. Many long wires are required to be routed between stages, adding to the overall complexity of the circuit.

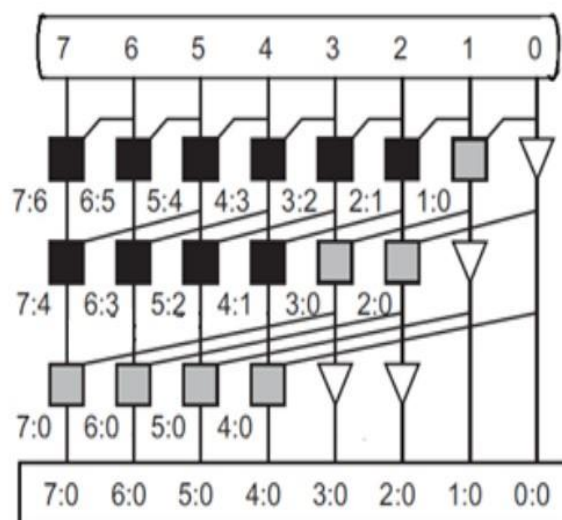


Fig:3.7 : Kogge Stone Adder

3.3.3 Brent Kung Adder

The algorithm presented by Brent and Kung in 1982 offers an alternative approach to parallel prefix addition with a focus on reducing the number of computation nodes. While their method results in a simpler tree structure, it comes with a trade-off in terms of maximal depth, leading to increased latency. Brent and Kung's algorithm simplifies the tree structure by computing the carry values only at positions corresponding to powers of two. This approach significantly reduces the number of computation nodes needed in the adder. Instead of computing carries for every bit position, the algorithm strategically selects positions that enable an efficient computation of intermediate carries. To further enhance the efficiency of the algorithm, an inverse carry tree is introduced to compute intermediate carries. This inverse carry tree is designed to complement the computation of carries at power-of-two positions, contributing to the overall simplicity of the adder structure. Compared to the Kogge-Stone adder, the Brent-Kung adder exhibits lower wire complexity. The reduction in wire complexity is particularly advantageous as it simplifies the routing of signals between different stages of the adder. While the Brent-Kung adder sacrifices some speed due to its maximal depth, its simpler structure and reduced wire complexity make it a practical choice in scenarios where these factors are critical considerations.

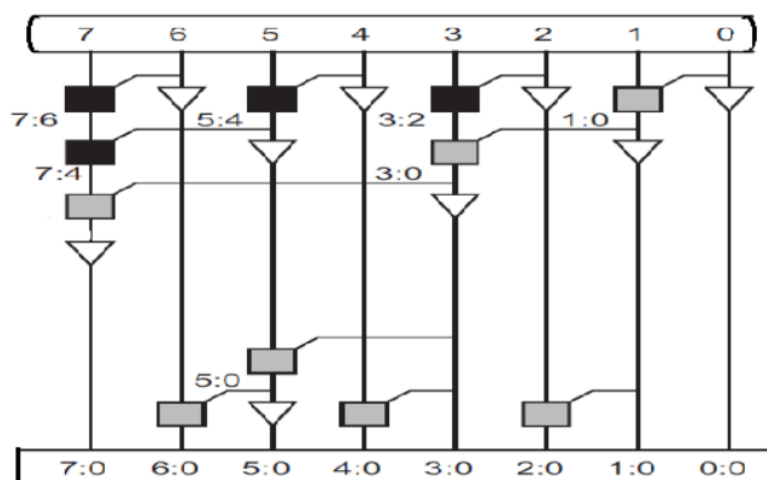


Fig:3.8 : Brent Kung Adder

3.3.4 Ladner Fischer Adder

The Ladner-Fischer trees represent a family of networks that fall between the Sklansky and Brent-Kung adder structures. In Ladner-Fischer trees, prefixes for the odd-numbered bits are computed separately, and an additional stage is utilized to ripple these results into the even positions. The Ladner-Fischer topology was introduced by Ladner and Fischer in 1980 as a general method for constructing a prefix network with a slightly higher depth compared to the Sklansky topology. The Ladner-Fischer method aims to reduce the maximum fan-out for computation nodes along the critical path. Fan-out refers to the number of gates or nodes that a signal must drive, and reducing it is essential for managing the latency in parallel prefix adders. The Ladner-Fischer tree has a computational node count given by $\lceil n/2 * \log_2(n) \rceil$, indicating the number of computational nodes required for a Ladner-Fischer adder. In summary, Ladner-Fischer trees offer a compromise between Sklansky and Brent-Kung adders, providing a structured approach that computes prefixes for odd-numbered bits separately and introduces an additional stage to propagate the results to even positions. This methodology, proposed by Ladner and Fischer, results in a prefix network with slightly higher depth than Sklansky but with reduced fan-out along the critical path, contributing to more efficient parallel prefix addition.

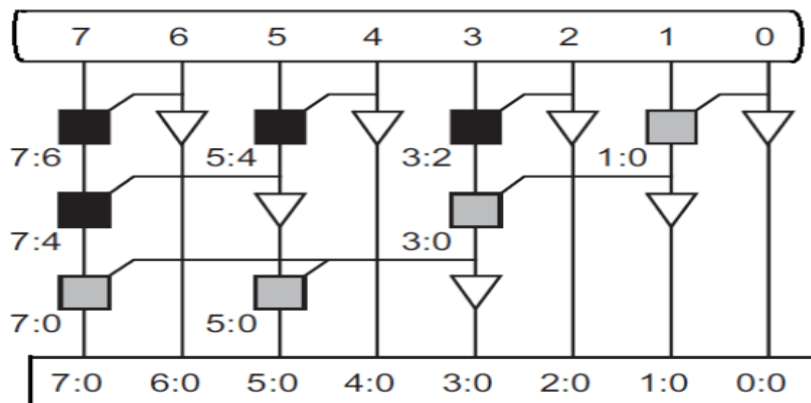


Fig:3.9 : Ladner Fischer Adder

3.4 IMPLEMENTATION OF EXISTING WORK

The implementation of existing work involves designing, simulating, and synthesizing traditional adder architectures such as Brent-Kung, Kogge-Stone, Han-Carlson, and Ladner-Fischer adders using Verilog or VHDL. Simulation tools like ModelSim or Vivado Simulator are utilized to verify functionality, followed by synthesis to generate netlists. Place and Route tools map these netlists onto FPGA fabric for hardware implementation. Static Timing Analysis ensures timing constraints are met. Bitstreams are generated for FPGA programming, allowing performance evaluation encompassing dynamic and static power consumption, delay, area utilization, and resource usage. Comparison and analysis of these metrics inform about each adder's strengths, weaknesses, and suitability for diverse applications, documented comprehensively for future optimization and research directions. Recent research has also explored hybrid approaches in adder design, aiming to leverage the strengths of multiple architectures while mitigating their limitations. The Hybrid BK-KS Adder, introduced as a novel architecture, combines features from both Brent-Kung and Kogge-Stone designs. This innovative approach seeks to strike a balance between power efficiency and performance, offering a versatile solution for various computational requirements. Adder architectures play a crucial role in digital VLSI design, serving as fundamental components in arithmetic units. Over the years, extensive research has been conducted to optimize adder designs in terms of power consumption, delay, area utilization, and overall performance. Traditional adder architectures, including the Brent-Kung, Kogge-Stone, Han-Carlson, and Ladner-Fischer adders, have been extensively studied and analyzed in various research works. Existing research works have focused on performance evaluation, optimization techniques, comparison studies, and application-specific implementations of adder architectures. These studies have provided valuable insights into the trade-offs and design considerations essential for advancing adder design paradigms in modern digital systems.

CHAPTER-4

METHODS AND ALGORITHMS USED

4.1 PROPOSED WORK

The proposed work for the Hybrid Brent-Kung-Kogge Stone (BK-KS) Adder project includes optimizing the adder's architecture and algorithms to improve performance, minimize critical path delay, and reduce power consumption. This involves refining algorithms, implementing advanced optimization techniques, and exploring low-power design methodologies such as clock gating and power gating. Thorough timing analysis, performance analysis, and functional verification will be conducted to ensure timing closure, assess speed improvements, and validate design correctness. Comprehensive documentation, including design specifications and test plans, will be maintained, and integration into larger FPGA-based systems will be tested to demonstrate the adder's versatility and suitability for high-speed signal processing and arithmetic operations. The Hybrid BK-KS Adder introduced in this study represents a novel and innovative approach that amalgamates the strengths of the Brent-Kung and Kogge-Stone architectures. This hybrid design is strategically crafted to address the inherent trade-offs found in conventional adders, offering a unique blend of optimized power efficiency, area utilization, and delay characteristics. With a remarkably low footprint of only 10 Look-up Tables (LUTs), the Hybrid BK-KS Adder stands out for its resource efficiency while simultaneously achieving a delicate balance between high-speed performance and low-power consumption. Leveraging the best features of both Brent-Kung and Kogge-Stone, this adder presents a versatile solution that fills a critical gap in the landscape of parallel prefix adders. Its potential applications span various domains, promising enhanced performance and adaptability in diverse computational scenarios.

4.2 XILINX IMPLEMENTATION

Xilinx is a prominent American technology company specializing in the development and manufacturing of programmable logic devices and associated technologies. Founded in 1984 and headquartered in San Jose, California, Xilinx has been a major player in the field of Field-Programmable Gate Arrays (FPGAs) and Programmable Logic Devices (PLDs). Here's a comprehensive overview of Xilinx. Xilinx is

renowned for its FPGA products, which are semiconductor devices that can be programmed after manufacturing to implement digital logic circuits.

FPGAs offer flexibility and reconfigurability, making them suitable for a wide range of applications, including telecommunications, automotive, aerospace, data centers, and more. Xilinx FPGAs are used in diverse applications, from embedded systems to high-performance computing. They can be reprogrammed to adapt to changing requirements, allowing for rapid prototyping and development. Xilinx FPGAs are classified into families such as Artix, Kintex, Virtex, and UltraScale, each catering to different performance, power, and capability requirements. These families evolve over time with newer generations introducing advanced features.

Xilinx provides a comprehensive suite of development tools called Vivado, which includes tools for synthesis, implementation, and verification of FPGA designs. Vivado enables FPGA developers to design and implement complex digital systems efficiently. Xilinx FPGAs are widely used in telecommunications infrastructure for functions such as signal processing, protocol handling, and baseband processing. Xilinx FPGAs find applications in automotive systems.

4.3 VERILOG IMPLEMENTATION

Verilog is a hardware description language (HDL) used for the modeling and simulation of digital circuits and systems. It is widely employed in the design and verification of electronic systems, particularly in the field of digital integrated circuits and FPGAs. Here's a full description of Verilog implementation. Verilog serves as a language for describing the behavior and structure of digital systems. It allows hardware designers to model complex digital circuits and systems using a textual representation. Verilog designs are typically organized into modules, which represent individual blocks of hardware. Modules can be hierarchical, allowing for a top-down design approach. Verilog code is written following a specific syntax. A module is declared using the module keyword, including the module name and a list of ports. Data types such as wire, reg, integer, and real are available for variable declaration. Procedural constructs like always blocks and conditional statements (if, else, case) are used to describe the logic and control flow of the system. Verilog is employed to create testbenches for simulating and verifying the functionality of digital designs. Testbenches are modules designed to apply inputs to the design and observe its outputs.

Simulation tools like ModelSim or VCS execute these testbenches, allowing designers to verify the correctness of their designs before implementation. After simulation and verification, Verilog code is synthesized into a netlist, representing the hardware implementation. Synthesis tools map high-level constructs to specific hardware components available in a target technology. This netlist is then used for subsequent steps in the design flow, such as place and route. In the context of FPGAs, Verilog is instrumental in mapping designs to specific FPGA resources.

This involves the assignment of logical elements, like Look-Up Tables (LUTs) and flip-flops, to physical locations on the FPGA. Place-and-route tools determine the optimal physical layout of the design in the FPGA, considering factors like signal timing and resource utilization. Verilog supports debugging through waveform viewers during simulation. Designers can analyze the behavior of signals over time, aiding in identifying and fixing potential issues. Assertion-based verification is also supported in Verilog, providing a formal method for specifying and verifying design properties.

Verilog designs are typically organized into modules, which represent individual blocks of hardware. Modules can be hierarchical, allowing for a top-down design approach. Verilog code is written following a specific syntax. A module is declared using the module keyword, including the module name and a list of ports. Data types such as wire, reg, integer, and real are available for variable declaration. Procedural constructs like always blocks and conditional statements (if, else, case) are used to describe the logic and control flow of the system. Verilog is employed to create testbenches for simulating and verifying the functionality of digital designs.

4.4 VIVADO SIMULATOR

ModelSim or Vivado Simulator is employed for functional verification and behavioural simulation of the designed adders. Testbenches are written to generate stimulus inputs and monitor the outputs of the adder designs under different test scenarios. Simulation results validate the correctness and accuracy of the adder implementations, ensuring they operate as intended. Vivado Synthesis translates the high-level Verilog/VHDL code into a gate-level netlist representing the logic gates and interconnections of the adder designs. Optimization techniques such as logic restructuring, technology mapping, and timing optimization are applied during

synthesis to achieve efficient implementation on the target FPGA device. Vivado Place and Route tools map the synthesized netlist onto the physical resources of the FPGA chip, including lookup tables (LUTs), flip-flops, routing resources, and I/O pins. Placement algorithms optimize the physical placement of logic elements to minimize signal delays and meet timing constraints. Routing algorithms establish connections between logic elements while considering resource availability and signal integrity. Static Timing Analysis in Vivado evaluates the timing characteristics of the adder designs to ensure that signal paths meet setup and hold time constraints. Timing constraints, such as clock frequency, input/output delays, and maximum propagation delays, are defined and analyzed to guarantee reliable and synchronous operation of the adders within the specified timing requirements. Performance metrics such as dynamic power consumption, static power consumption, delay (propagation delay, critical path delay), area utilization (number of LUTs, flip-flops), and resource usage (routing congestion, clock regions) are measured and analyzed for each implemented adder architecture. Vivado provides built-in tools for collecting and analyzing these metrics during simulation and hardware implementation. The performance results obtained from simulations and hardware implementation are compared and analyzed to identify the strengths and weaknesses of each adder architecture. Trade-offs between power consumption, delay, area utilization, and resource usage are evaluated to determine the most suitable adder design for specific application scenarios. Detailed documentation is prepared, including design specifications, simulation waveforms, synthesis reports, static timing analysis reports, performance evaluation data (tables, graphs), comparison charts, analysis findings, and conclusions. These reports serve as a comprehensive reference for understanding the design process, results interpretation, and future design optimizations.

4.5 FPGA DESIGN SYNTHESIS

FPGA design involves a comprehensive process starting with specification and requirements analysis, where the functional aspects of the design, performance targets, and constraints are defined. Following this, the architecture design phase selects the appropriate FPGA device and partitions the design into functional modules. RTL design using Verilog or VHDL captures the behavior and

interconnections of logic elements, while simulation and verification validate the design's functionality and timing characteristics. Synthesis optimizes the RTL code into a gate-level netlist, and physical implementation utilizes Place and Route tools to map the design onto the FPGA hardware. Static Timing Analysis ensures timing constraints are met, leading to bitstream generation and FPGA programming. Testing and validation verify the design's correctness and performance, with comprehensive documentation capturing the design process, optimizations, test results, and user manuals for future reference and knowledge sharing. Use simulation tools (e.g., ModelSim, Vivado Simulator) to perform functional verification of the RTL design, creating testbenches to simulate different input scenarios and verify correct behavior. Synthesize the RTL code using synthesis tools (e.g., Vivado Synthesis) to generate a gate-level netlist optimized for the target FPGA device. Apply optimization techniques such as logic restructuring, technology mapping, and resource sharing to minimize area utilization, improve timing performance, and reduce power consumption. Utilize Place and Route tools (e.g., Vivado Place and Route) to map the synthesized netlist onto the physical resources of the FPGA chip. Perform functional testing and validation on the programmed FPGA to verify correct operation, functionality, and performance under real-world conditions. Generate the FPGA configuration bitstream, which contains the programming instructions for configuring the FPGA with the synthesized and placed-and-routed design. Program the FPGA device using programming tools (e.g., Xilinx Vivado, Intel Quartus Prime) to load the bitstream and instantiate the designed digital circuit on the FPGA hardware. Prepare comprehensive documentation, including design specifications, simulation results, synthesis reports, place-and-route reports, timing analysis reports, validation test results, and user manuals. Document design constraints, optimizations, trade-offs, and lessons learned during the FPGA design process for future reference and knowledge sharing. FPGAs are widely used for hardware acceleration, offloading computationally intensive tasks from general-purpose processors to dedicated hardware blocks implemented on the FPGA fabric. This acceleration improves system performance, energy efficiency, and scalability for applications like machine learning, signal processing, cryptography, and scientific computing. FPGA designers can create and integrate custom Intellectual Property (IP) cores, such as digital signal processing (DSP) blocks, memory controllers, communication interfaces, and specialized accelerators. These IP cores enhance

the functionality and performance of FPGA-based systems, offering tailored solutions for specific application domains. FPGAs support high-speed interfaces like PCIe (Peripheral Component Interconnect Express), DDR (Double Data Rate) memory interfaces, Ethernet, USB, HDMI, and others. These interfaces facilitate seamless integration with external devices, data exchange, and communication in diverse embedded and computing systems.

4.6 BITSTREAM GENERATION

A bitstream is a binary file that contains the configuration data necessary for programming FPGA (Field-Programmable Gate Array) devices. It encapsulates information about logic element placements, routing resources, I/O pin assignments, clock settings, and other parameters essential for configuring the FPGA with a specific digital design. Generated during the synthesis and implementation stages of FPGA design, bitstreams are vendor-specific and may include security features like encryption to protect configuration data. They play a crucial role in programming the FPGA hardware, enabling engineers to instantiate desired digital circuits or systems on the FPGA platform. Additionally, bitstreams can support partial reconfiguration in some FPGA architectures, allowing specific regions of the FPGA to be reprogrammed dynamically.

4.7 STANDARDS

The standards for the Hybrid Brent-Kung-Kogge Stone (BK-KS) Adder project encompass several aspects to ensure quality, reliability, and adherence to best practices in FPGA design and implementation. Here are some key standards that the project may adhere to:

- **IEEE 1076 (VHDL Standard):** IEEE 1076 defines the VHDL (VHSIC Hardware Description Language) standard for specifying digital systems and circuits. It provides guidelines for syntax, semantics, and behaviour modelling in VHDL, which is commonly used for FPGA design.
- **IEEE 1364(Verilog Standard):** IEEE 1364 specifies the Verilog Hardware Description Language standard, another widely used language for describing digital systems. It covers syntax, semantics, and simulation of Verilog code, which is often used in FPGA design alongside VHDL.

- **IEEE 1149.1 (JTAG Standard):** IEEE 1149.1, also known as the Joint Test Action Group (JTAG) standard, defines a serial interface for testing and debugging digital circuits. It is commonly used for boundary-scan testing in FPGAs and other integrated circuits.
- **IEEE 1164 (VHDL Standard for Synthesis):** IEEE 1164 extends the VHDL standard with additional features specifically for synthesis. It includes guidelines for synthesizable constructs, timing annotations, and design considerations for FPGA synthesis tools.
- **IEEE 1800 (System Verilog Standard):** IEEE 1800 specifies the System Verilog standard, an extension of Verilog with additional features for verification, testbench development, and system-level modelling. It is widely used in FPGA design for verification and testbench automation.
- **FPGA Design Best Practices:** Follow industry-standard guidelines and recommendations for FPGA design, including coding style conventions, naming conventions, documentation standards, and design hierarchy structures. Utilize version control systems (e.g., Git) for managing design files, revisions, and collaboration among team members.
- **Verification and Validation:** Conduct rigorous functional verification using simulation tools to validate the correct behaviour of the adder design under various input conditions and corner cases. Perform timing analysis and static timing verification to ensure that timing constraints are met and there are no timing violations in the design.

4.8 CONSTRAINTS

The constraints of the Hybrid Brent-Kung-Kogge Stone (BK-KS) Adder project encompass various limitations and requirements that must be considered during design, implementation, and testing phases. Here are some common constraints that may apply to this project:

- **Timing Constraints:** Ensure that the adder design meets timing requirements specified by the target FPGA device, such as maximum clock frequency, setup and hold times, and propagation delays. Address timing closure challenges by optimizing critical paths, minimizing clock skew, and resolving timing violations identified during static timing analysis (STA).
- **Resource Constraints:** Manage resource utilization within the constraints of the target FPGA device, including lookup tables (LUTs), flip-flops, memory blocks, DSP slices, and routing resources. Optimize resource allocation and placement to minimize area utilization and maximize performance without exceeding resource availability limits.
- **Power Constraints:** Design the adder circuit with consideration for power consumption constraints, aiming to minimize dynamic power dissipation, static power leakage, and overall energy consumption. Apply power optimization techniques such as clock gating, power gating, voltage scaling, and low-power design methodologies to meet power budget constraints.
- **Technology Constraints:** Adhere to technology-specific constraints and guidelines provided by the FPGA vendor (e.g., Xilinx, Intel) regarding supported features, architecture limitations, tool compatibility, and design flow requirements. Ensure compatibility with the chosen FPGA device family, device package, I/O standards, and interface protocols required for interfacing with external components or systems.
- **Performance Constraints:** Validate the adder's performance against specified performance metrics, such as throughput, latency, operating frequency, and signal integrity, to ensure that it meets application requirements and performance targets. Address performance bottlenecks, timing violations, and resource contention issues through iterative design optimization and performance tuning.
- **Functional Constraints:** Verify the correct functionality and operation of the adder design under various input conditions, edge cases, and corner scenarios.

to ensure functional correctness and reliability. Validate compliance with design specifications, functional requirements, and industry standards (e.g., IEEE standards for digital circuits) through thorough testing and verification methodologies.

4.9 TRADE OFF

The Hybrid Brent-Kung-Kogge Stone (BK-KS) Adder project involves several trade Offs that designers must consider to optimize the adder's performance, area utilization, power consumption, and functionality. Here are some common trade Offs associated with this project:

- **Performance Vs. Area Utilization:** Increasing performance often requires more hardware resources, leading to higher area utilization. Designers must optimize the adder's architecture and algorithms to achieve desired performance levels while minimizing area overhead.
- **Speed Vs. Power Consumption:** Higher operating speeds typically result in increased power consumption due to faster switching activity. Designers need to strike a balance between speed and power efficiency by implementing clock domain optimizations, power gating techniques, and low-power design strategies.
- **Resource Utilization Vs. Functionality:** Adding more functionality to the adder design may increase resource utilization and complexity. Designers must prioritize essential features and functionalities while managing resource constraints to avoid excessive area usage and maintain design simplicity.
- **Timing Closure Vs. Design Complexity:** Achieving timing closure may require complex design optimizations and constraints, increasing design complexity. Designers should employ efficient timing closure techniques, such as pipelining, retiming, and slack management, while maintaining manageable design complexity.
- **Flexibility Vs. Performance Optimization:** Design flexibility allows for easy modifications and adaptations but may impact performance optimization efforts. Designers need to balance design flexibility with performance optimization by choosing scalable architectures, modular design approaches, and configurable parameters.

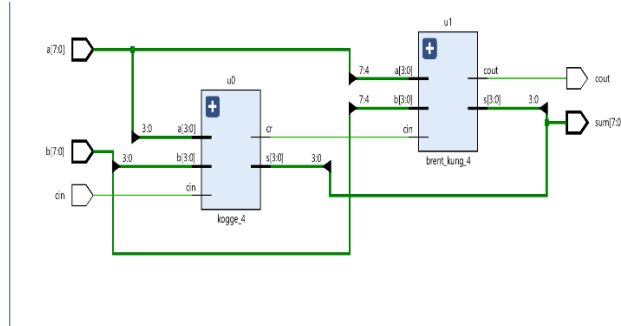


Fig:5.2 : Simulation results for Hybrid Kogge Stone-Brent Kung Adder

The Hybrid BK-KS Adder may use a multi-stage design where initial stages focus on fast parallel carry computation using Kogge-Stone-like structures, while subsequent stages employ recursive carry computation for efficient carry propagation and fan-out reduction. By strategically combining these two approaches, the Hybrid BK-KS Adder aims to minimize the overall delay, reduce area overhead, and enhance performance in terms of speed and power efficiency compared to standalone Brent-Kung or Kogge-Stone adders. The exact implementation details of the Hybrid BK-KS Adder can vary based on design goals, target FPGA device, technology constraints, and optimization strategies employed during synthesis and place-and-route phases.

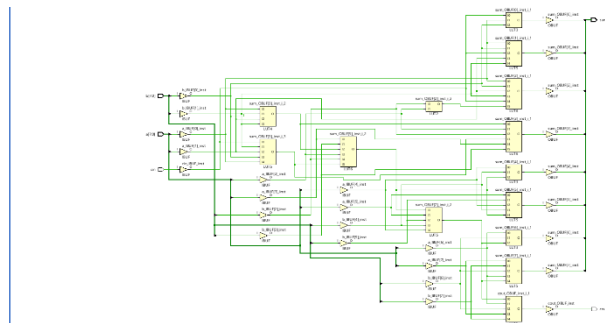


Fig:5.3 : RTL Schematic of Hybrid Kogge Stone-Brent Kung Adder

Overall, the Hybrid Brent-Kung-Kogge Stone Adder represents an innovative approach that leverages the strengths of two established parallel prefix adder architectures to achieve a well-balanced trade-off between speed, area, and power in digital circuit designs.

5.2 PERFORMANCE ANALYSIS

Now that we have understood several parallel prefix adders and the corresponding prefix tree diagrams, it's time for valuating them for the best pick for a particular application. The below table gives a glance of performance attributes of all the PPAs discussed in this paper in theory as a function of input bit width. Note that the table also includes a class of Knowles parallel adders.

Table 5.1: Summarizing the Performance Comparison of Prefix Adders discussed in the paper.

Prefix Adder	Static Power	Dynamic Power	Delay(ns)	No.of Logic LUTs
Koggle-Stone	0.165	0.911	26.895	37
Han-Carlson	0.158	0.843	27.091	32
Brent-Kung	0.150	0.747	27.317	20
Ladner-Fischer	0.156	0.838	26.902	28
Hybrid Brent-Kung and Koggle Stone	0.104	0.538	23.093	10

In comparing existing adders, the Brent-Kung Adder stands out for its lower dynamic power consumption, while the Kogge-Stone Adder exhibits higher power consumption and hardware utilization. The Han-Carlson and Ladner-Fischer adders show subtle differences in power consumption, with Han-Carlson positioning itself as a hybrid approach. Introducing the Hybrid BK-KS Adder, a novel architecture. It leverages the strengths of both Brent-Kung and Kogge-Stone, offering a unique balance in power efficiency, delay, and hardware utilization. The Hybrid BK-KS Adder introduces a compelling alternative, providing designers with a versatile solution that aims to optimize performance characteristics in parallel prefix adder. We evaluate the area efficiency of each adder circuit by considering the number of Logic LUTs (Look-up tables) used during the implementation process, as indicated

by the tool. Figure 23 underscores the remarkable efficiency of the Brent-Kung adder, evident in its impressive hardware utilization, utilizing almost half the number of logic LUTs compared to the Kogge-Stone adder.

The Han-Carlson adder, featuring 32 LUTs, strategically positions itself between Kogge-Stone and Brent-Kung in terms of hardware utilization. Notably, our introduced Hybrid BK-KS Adder, boasting a mere 10 LUTs, presents an innovative approach by seamlessly integrating features from both Brent-Kung and Kogge-Stone architectures. This hybrid design strives to strike a balance in terms of power efficiency, area utilization, and delay, offering a compelling alternative for diverse applications.

In comparison, while Brent-Kung exhibits superior performance in power and area, Han-Carlson, slightly more area-efficient than Ladner-Fischer, falls behind in dynamic power consumption. Despite Kogge-Stone's suboptimal suitability for low power and resource-critical applications, it emerges as the preferred choice for high-speed applications, particularly DSPs, owing to its minimal delay, as highlighted in Figure 24. The summarized table encapsulates all performance aspects discussed, including delay, static power consumption, and the innovative characteristics introduced by the Hybrid BK-KS Adder. The table also provides a good intuition that Hybrid Kogge Stone-Brent Kung adder has much lower delays compared to those of other prefix adders when the bit-width of the inputs increases. The performance analysis of the Hybrid Brent-Kung-Kogge Stone (BK-KS) Adder project involves a comprehensive evaluation of key metrics to gauge its efficiency and effectiveness. Firstly, the speed analysis focuses on determining the critical path delay of the adder design through timing simulations and Static Timing Analysis (STA), ensuring that timing constraints are met for reliable operation. Secondly, area utilization is assessed by calculating the number of logic elements utilized in the FPGA implementation, analyzing resource distribution and routing efficiency to optimize FPGA resource usage. Thirdly, power consumption is evaluated by estimating both dynamic and static power components during operation, aiming to enhance energy efficiency and minimize power overhead. Additionally, functional verification tests validate the adder's correct operation under various input conditions, ensuring its reliability and robustness. Comparative analysis with traditional adder architectures aids in identifying strengths and weaknesses, guiding optimization efforts for improved performance.

CHAPTER-6

SUMMARY AND CONCLUSION

6.1 SUMMARY

The project focuses on the development and analysis of a Hybrid Brent-Kung-Kogge Stone (BK-KS) Adder, which integrates features from both traditional adder architectures. The aim is to achieve a balance between speed, area utilization, and power consumption in FPGA designs. The project involves phases such as specification analysis, architecture design, RTL coding, simulation, synthesis, and physical implementation using Xilinx Vivado Design Suite. Performance analysis metrics include speed, area utilization, power consumption, resource usage, and functional verification. The project aims to demonstrate the efficiency and effectiveness of the Hybrid BK-KS Adder compared to conventional adder designs, with potential applications in high-speed signal processing and low-power VLSI designs.

6.2 CONCLUSION

In this project, we explored the limitations of conventional adders and investigated Parallel Prefix adders. Implementing these designs in Vivado Design Suite revealed trade-offs among them. The Kogge-Stone Adder proved fastest but with higher power and area usage, ideal for high-speed signal processing. The Brent-Kung Adder excelled for low-power VLSI designs. Our innovative Hybrid BK-KS Adder, merging Brent-Kung and Kogge-Stone features, offers a versatile solution. The Han-Carlson Adder presented a hybrid balance, and Ladner-Fischer outperformed it in resource consumption. Future work includes extending bit-width, optimizing power consumption, fan-out comparisons, and designing FIR filters. Leveraging reconfigurable FPGAs in Zynq-7000 SoC for dynamic adder reconfiguration is a promising avenue.

REFERENCES

- [1] Balasubramanian P, Dang C, Maskell D. L, and Prasad K, "Approximate ripple carry and carry lookahead adders-A comparative analysis," in Proc. IEEE 30th Int. Conf. Microelectron. (MIEL), Oct. 2017, pp. 299–304.
- [2] Ban T, Wang B, and Naviner L, "Design, synthesis and application of a novel approximate adder," in Proc. IEEE 61st Int. Midwest Symp. Circuits Syst. (MWSCAS), Aug. 2018, pp. 488–491.
- [3] Chandrika B, Poorna Krishna G, "Implementation and Estimation of Delay, Power and Area for Parallel Prefix Adders", International Journal for Modern Trends in Science and Technology Volume: 02, Issue No: 11, November 2016.
- [4] Dimitrakopoulos G, Papachatzopoulos G, and Paliouras V, "Sum propagate adders," IEEE Trans. Emerg. Topics Comput., vol. 9, no. 3, pp. 1479–1488, Jul. 2021.
- [5] Ebrahimi Azandaryani F, Akbari O, Kamal M, Afzali Kusha A, and Pedram M, "Block-based carry speculative approximate adder for energy-efficient applications," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 67, no. 1, pp. 137–141, Jan. 2020.
- [6] Esposito D, De Caro D, and Strollo A.G.M, "Variable latency speculative parallel prefix adders for unsigned and signed operands," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 63, no. 8, pp. 1200–1209, Aug. 2016.
- [7] Mahdiani H, Ahmadi A, Fakhraie S, and Lucas C, "Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 4, pp. 850–862, April 2010.
- [8] Schlachter J, Camus V, Palem K.V, and Enz C, "Design and applications of approximate circuits by gate-level pruning," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 5, pp. 1694–1702, May 2017.
- [9] Seo H, Yang Y, and Kim Y, "Design and analysis of an approximate adder with hybrid error reduction," Electronics, vol. 9, p. 471, Mar. 2020.
- [10] Shafique M, Ahmad W, Hafiz R, and Henkel J, "A low latency generic accuracy configurable adder," in Proc. 52nd ACM/EDAC/IEEE Design Autom. Conf. (DAC), Jun. 2015, pp. 1–6.

- [11] Shilpa C.N, Kunjan Shinde G, Nithin H.V, "Design Implementation and Comparative Analysis of Kogge Stone Adder using CMOS and GDI design: A VLSI Based Approach", IEEE, pp. 570-574, Dec. 2016.
- [12] Simsek C, and Turk K, "Hardware optimization for belief propagation polar code decoder with early stopping criteria using high-speed parallel-prefix ling adder." pp. 182-185, Jan.2021.
- [13] Slipkoand V.A, Pershin Y.V, "Importance of the window function choice for the predictive modelling of memristors", IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 68, no. 6, pp. 2167 2171,2019.
- [14] Soares L.B, Da Rosa M.M. A, Diniz C.M, and Bampi S, "Design Methodology to Explore Hybrid Approximate Adders for Energy-Efficient Image and Video Processing Accelerators," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 66, no. 6, pp. 2137– 2150, 2019.
- [15] Strollo A.G.M, Napoli E, De Caro D, Petra D, Saggese G, and Meo G.M, "Approximate multipliers using static segmentation: Error analysis and improvements," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 69, no. 6, pp. 2449–2462, Jun. 2022.
- [16] Thakur G, Sohal H, and Jain S, "FPGA-based parallel prefix speculative adder for fast computation application," in Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), Wagnaghat, India, 2020, pp. 206-210.
- [17] Veena M, Akkamanchi P, and Uttarkar V, "Low power high speed brent kung adder using spst," in IEEE 3rd Global Conference for Advancement in Technology (GCAT), 2022, pp. 1-6.
- [18] Xiaoping Wang, Bowen Xu, and Lin Chen, "Efficient Memristor Model Implementation for Simulation and Application", IEEE Transactions on computer-aided design of integrated circuits and systems, vol. 36, No. 7, July 2017.
- [19] Yang Z, Jain A, Liang J, Han J, and Lombardi F, "Approximate XOR/XNOR-based adders for inexact computing," in Proc. 13th IEEE Int. Conf. Nanotechnol., Aug. 2013, pp. 690–693.
- [20] Zhu H, Goh W.L, Zhang W, Yeo K.S, and Kong Z.H, "Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing," IEEE Trans. Very Large Scale Integrated (VLSI) Syst., vol. 18, no. 8, pp. 1225–1229, Aug. 2010.

APPENDIX

A. PLAGARISM REPORT OF CONFERENCE PAPER

nani 1

ORIGINALITY REPORT

10%

SIMILARITY INDEX

6%

INTERNET SOURCES

10%

PUBLICATIONS

1%

STUDENT PAPERS

PRIMARY SOURCES

1

ebin.pub

Internet Source

4%

2

"Advances in Cognitive Science and Communications", Springer Science and Business Media LLC, 2023

Publication

1%

3

Nikhil Advait Gudala, Trond Ytterdal, John J. Lee, Maher Rizkalla. "Implementation of High Speed and Low Power Carry Select Adder with BEC", 2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), 2021

Publication

1%

4

de.slideshare.net

Internet Source

1%

5

K. Rajarajan, Ginson P. Joseph, S. M. Ravi Kumar, I. Vetha Potheher, A. Joseph Arul Pragasam, K. Ambujam, P. Sagayaraj. "Growth and Optical Studies of a Novel Organometallic Complex NLO Crystal: Tetrathiourea

<1%