Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Optimized Carry Select Adder with Speculative Logic Using Kogge-Stone Prefixing for Low Latency Applications

Kalaivani Ramanathan

kalaivani.r@esec.ac.in

Erode Sengunthar Engineering College

Srijaa S

Erode Sengunthar Engineering College

Sampoornam KP

Article

Keywords: Carry Select Adder, Kogge- Stone, Speculative Logic, Parallel, Prefix, Low Latency, VLSI

Posted Date: July 9th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10077849/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Optimized Carry Select Adder with Speculative Logic Using Kogge-Stone Prefixing for Low Latency Applications

Kalaivani R
Professor / Department of Electronics and Communication Engineering
Erode Sengunthar Engineering College
Erode, India.
kalaivani.r@esec.ac.in
Corresponding Author

Srijaa. S
UG Student / Department of Electronics and Communication Engineering
Erode Sengunthar Engineering College
Erode, India.
srijaasekar1912@gmail.com

Sampoornam K P
Professor / Department of Electronics and Communicartion Engineering
Bannari Amman Institute of Technology
Sathyamangalam, Erode, India
sampoornam@bitsathy.ac.in

Abstract - The increased performance demand and reduced power dissipation addressed in modern processors, digital signal processing (DSP) systems and embedded systems set off the needs of different efficient adder architectures. Typical adders such as the Ripple Carry Adder (RCA) are limited in terms of high-speed applications to relying on carry propagation delay from the previous bit additions. Carry select adder (CSLA) easily speeds up calculations by computing the sum with either carry (0 or 1) simultaneously, but equals needing duplicated adder blocks. In order to address these issues, the project introduces an Optimized Carry Select Adder (OCSLA) which combines the improvements offered by Speculative Logic with those of a Kogge stone Adder (KSA). Speculative Logic is based on the scheme of pre-calculating the carry signals, so computing the partial sums in parallel and overcoming the drawback of the long carry propagation. The Kogge stone parallel prefix structure allows the carries at all the bit positions minimizing the delay of the critical path. Verification of this proposed design was built and simulated using Vivado Design Suite with Verilog HDL. Experimental results shows that optimized architecture can improve performance with power consumption, LUT utilization and Input/Output utilization, compared with the present design which make use of power and Input. From the simulation results presented we can conclude that overall the suggested hybrid architecture reduces the computation time along with efficient utilization of hardware. So, the designer architecture designed is more convenient for high-speed processors, ALU & low power VLSI systems.

Keywords : Carry Select Adder, Kogge- Stone, Speculative Logic, Parallel, Prefix, Low Latency, VLSI

1. Introduction

The project “Optimized Carry Select Adder with Speculative Logic Using Kogge-Stone Adder Techniques for Low Latency Applications” proposes structures for devices performing addition, to speed up and optimize them. Addition is one of the most basic digital operations and one of the most used as many processing elements perform addition operations, such as ALUs, multi-purpose processors and an innumerable number of systems performing DSP operations. The structure described below is one of the best-known structures and the one performing addition operations at the greatest speed. It is called Carry Select Adder (CSLA) and is faster than the Ripple Carry Adder as it reduces the speed limitation of the latter by splitting the adder into blocks that perform two possible values for an adder assuming that the carry-in is 0 or 1 and then using a multiplexer to select the correct one by applying the real carry-in., In order to solve this problem, the design is using Speculative Logic that predicts the carries of the adder firstly. Then the adder predicts the sum according to this carry. The prediction is entered so that the sum could just start run before the carry propagates. Furthermore, overall structure is against the Kogge-stone Adder (KSA) which is also high-performance parallel prefix adder, KSA has a fast carry computation to enable fast calculation, it computes all the carries at the same time by using parallel tree instead of serial one so that rapidly reduce the delay time. Hybrid adder implemented by with combination CSLA, the speculative the KSA logic can both achieve rapid and powerful logic. By comparing the different architecture, we see that this is the fastest for speed and hardware reduction and power saving (Propagation delay, hardware complexity and power consumption are minimized), thus this can be used in the modern high speed digital system. Building bigger bit width adder also scales would be more efficient as we had already minimized the delay. The final output obtained by using this design is a high-speed high throughput low latency adder which is ideally suited for general high-speed computing (should take as much the same power as one bit adder).

2. Literature Review

Another important aspect in VLSI system design optimization research for VLSI system is the optimization of adder design and optimization. Designing fast and power efficient adders for applications in digital signal processing, cryptography and real-time computing systems is an important aspect of VLSI system design. Speaking of fast adders, extensive research efforts have been

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

made to study of parallel-prefix adders such as the Kogge-Stone Adder (KSA).The authors once investigating the problem of improving power efficiency of the Kogge-Stone adder using the optimized OR gate circuit was in the 45 nm CMOS technology in the work John et al,. 2020. This introduced approach can reduce the power dissipation and PDP value. However, it is merely targeted at the optimization at the gate level and doesn‘t talk about the architectures-level delay minimization or the hybrid adder integration techniques. Balasubramanian and Maskell (2024) suggested a new design of Carry Look Ahead Adder (CLA) architecture, consisting of modules of various sizes to minimize the latency and energy. With a design of such a structure, the delay and the power are evident compare to the ordinary CLA and other high-speed adder. It is merely has an optimization on CLA, without the other factor of the speculative carry nature or the hybrid of the parallel-prefix, which may discover the further minimum latency of the system. Sathish and Jagsdeesh(2023) studied the comparison between the Sklansky and the Kogge-Stone based modification to Wallace multipliers. The study proves that delay through Sklansky is lower than KSA in the case of multipliers. Although the result points to the feature of adding delay as entire system delay contributor, the study didn‘t found out the optimized design in the standalone adder architecture or the hybrid adder architecture. Speculative computation methods have been studied to improve speed. Pragadeswaran et al. designed an Inexact Speculative Adder (ISA) combined with a Carry Look-Ahead Adder (CLA) where the use of pipelining and clock gating improved the operating frequency and lowered the power consumption. Speculative logic enhances the performance, but the drawback is the aspect of approximation, and such design is mainly focused on CLA architectures rather than combining speculative techniques with state-of-art prefix adders such as KSA.

Hybrid adder architectures have been investigated to take advantage of the benefits from various implementations. Simson and D (2021) also designed a hybrid Carry select adder (CSLA) using Kogge–Stone and CLA style for increased speed and energy efficiency. But the study uses a hybrid design that does not fully utilizes the speculative logic or the further optimized ways of carry selection which could further reduce latency. The comparison of various types of parallel prefix adders have also been studies. For instance, Abhilash and Hussain (2024) are able to study different implementations of prefix adder like Brent–Kung, Kogge–Stone, Sklansky and Ladner-Fischer adders and come to the conclusion that none of the implementation is able to provide optimal compromise between power and delay, thus highlighting the effectiveness of hybrid architectures.

Degradation of area overhead in KSA design has been reported by Daphni and Vijula Grace (2021) who have obtained reduction in transistor number and area through pass transistor logic (PTL). Nonetheless the hardware complexity is decreased but it does not solve the high latency issues and interaction with other high speed techniques is overlooked.

Yang et al. (2010) developed a modified parallel prefix adder, in order to alleviate both the delay and the areas are aspect of the conventional KSA. Improvements were achieved, however, in terms of delay, the reduction is still minimal and the design does not make use of speculative and hybrid carry selection techniques to speed up the calculation.

Last but not the least, to explore the advantage of various adder architectures, Mohakud et al. (2024) used and compared several adder architectures on FPGA platforms. They concluded Kogge–Stone adder had greater power and latency efficiency than ripple carry and carry select adder architecture. But the research was only limited to the comparison of mentioned architectures, without any new architectural design proposed for optimization.

It can be seen from the review above that most of the researches present either an optimization focusing on the gate level to reduce power and area, or an architecture level optimization for speedup, and a unified view of performance and low power/area characteristics has not yet been achieved. While the use of Kogge–Stone adder gives advantages in high speed, it introduces area overhead and complex structural arrangements, and there seems to be very little work on employing its utility for latency optimization with carry select adder organization. Speculative logic has been predominantly incorporated to carry look-ahead based adders, with not much work on merging it with the hybrid CSLA–KSA architecture. The potential of using speculative carry generation and selection to further optimize the hybrid adder architecture has not been utilized in existing designs.\

To overcome the limitations of the approaches discussed above, this work is aimed to be an Optimized Carry Select Adder with Speculative Logic using Kogge-stone Adder Techniques, aimed at reducing the critical path delay by use of Speculative Carry prediction, increasing the efficiency of the algorithm by increasing the speed of computation, by use of the Parallel Prefix Kogge-stone Structures, and improvement in the area and power by enhancement of the efficiency of design, by use of hybrid approaches. The optimization is aimed at reaching an optimal compromise among power, latency, and hardware efficiency, for applications demanding low latency.

3. Existing System

The conventional Carry Select Adder (CSA) is most commonly used architecture in high-speed arithmetic units. In the existing system, each CSA block consists of two Ripple Carry Adders operating in parallel - one assuming carry-in as ‘0’ and the other

assuming carry-in as '1'. A multiplexer is used to select the correct sum and carry output once the actual carry-in is known. Although this approach reduces propagation delay compared to RCA, it results in increased hardware redundancy, leading to higher area and power consumption.

In many existing designs, Ripple Carry Adders are used within CSA blocks due to their simplicity. However, the internal carry propagation within these RCAs still contributes to delay. Even optimized CSAs that use modified block sizes or logic sharing techniques struggle to meet the stringent timing requirements of modern high-performance processors.

Parallel prefix adders such as the Kogge–Stone adder are sometimes that are used independently in existing systems to achieve high speed. While they provide fast carry generation, their high interconnect complexity and power dissipation limit scalability when used alone. Furthermore, existing systems typically do not exploit speculative carry computation, resulting in underutilization of parallelism in carry generation.

Therefore, the existing adder architectures face limitations in achieving an optimal balance between speed, power, and area, motivating the need for a hybrid approach that combines speculative carry logic, Carry Select Adder structure, and fast parallel prefix networks.

4. Architecture and Block Diagram

The Architecture and the block diagram of the presented system show the architecture of optimized Carry select adder architecture for low latency arithmetic operation. The architecture combines the speculative operation with fast parallel prefix carry generation to fasten the addition operation. The system is then integrated into three important functional modules such as the Kogge Stone adder block, the speculative c logic block and the adaptive control logic block.

4.1 Kogge Stone Adder Block:

The Kogge - Stone Adder Block handles high speed carry propagation when calculated all way through the adder that can be shown in Figure 4.1. It is built based on a parallel prefix network which takes the propagate and generate signals calculated from the input operands. In this structure, the carry signals are generated through a number of prefix stages arranged in a tree structure, each stage combines the propagate and generate signals from other previous stages to generate the carry signal at next higher bit position.

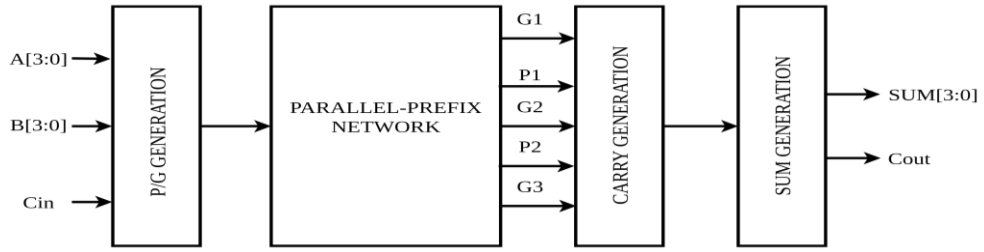


Fig 4.1 Kogge - Stone Adder Block

To quantify, the Kogge-Stone network produces all carry signals within six logic levels, which is an enormous decrease in delay compared to standard design based on preceding carries. This block guarantees the efficient and scalable carry computation for wide bit width operation. In the design of this paper, the Kogge–Stone network is the semantic core of the carry computation block and facilitates the high-speed inter-block communication of carry signals.

4.2 Speculative carry logic block:

Speculative Carry Logic Block: An additional block which will reduce delay by predicting several carry values before even computing the exact carry. This block contains two local adder paths working in parallel. The first adder path assumes that carry-in is 0 and the second adder path assumes that c in is 1. Both the adders calculate the sum outputs in parallel**.**

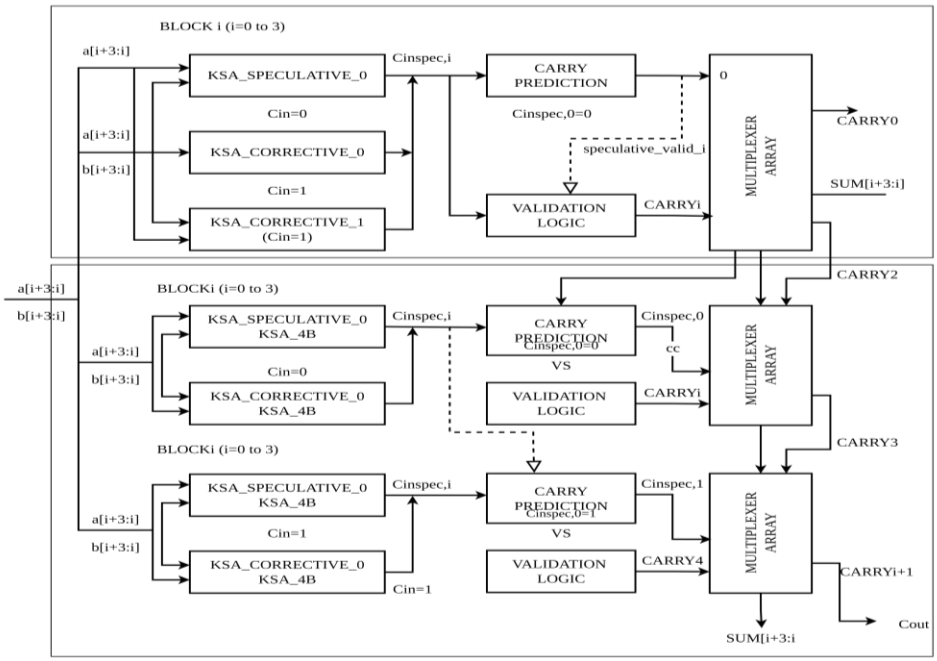


Fig. 4.2 block diagram of Speculative Carry Logic

The Fig. 4.2 explains both the scenarios are known beforehand, this speculations presumes to them and start calculation before the carry is generated by the previous stage. In this way, the time of the carry propagation can be hidden. When the prefix network generates the real carry, the correct output can be chosen immediately.

4.3 Adaptive Control Logic Block:

Finally, the actual carry signals from the Kogge–Stone prefix network are used as control signals of the Multiplexer to determine which sum is the right one out of the two speculative sums from the previous stage. The Fig. 4.3 shows the adaptive logic control is employed to guarantee that the final output is consistent with the correct carry condition, and to keep the speculative computation along with the final carry resolution in synchronization. In the adaptive control logic, sum and carry value are always correct without losing the high-speed benefit of the speculative calculation and the prefix carry generation.

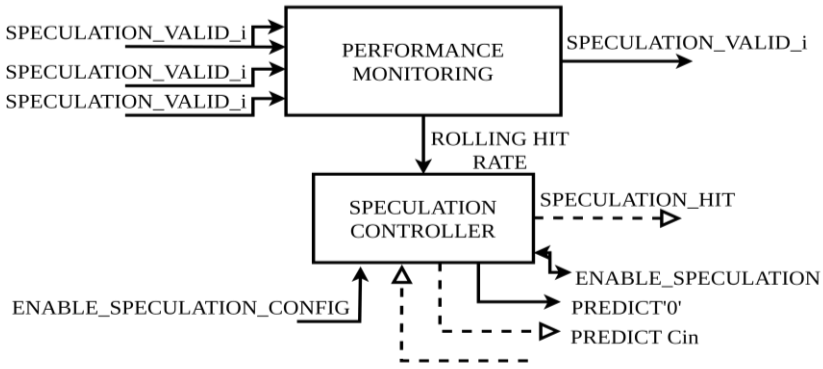


FIG. 4.3 BLOCK DIAGRAM OF ADAPTIVE CONTROL BLOCK

5.1 Component Description

Propagate and Generate Signals

The first stage in the architecture of the proposed Carry Select Adder is the Propagate and Generate (PG) Logic Units. And the purpose of this section is to provide the basic signal generating outputs required to find carry values in the binary

addition operation. This is also the signal providing stage to allow for fast carry calculation in subsequent stages of the architecture in the parallel prefix carry network.

For each bit position i, the PG logic unit has two input bits from A and B. These two bits are used to compute two signals: generate signal (g) and propagate signal (p). Generate signal indicates if a position will create a carry immediately regardless of the fact if the carry is coming into it from the previous position or not. Generate signal is determined by an AND operation with the two bits as inputs. When both bits are equal to logic '1', generate logic is equal to 1 which transmits a carry.

The propagate signal is used to indicate if the carry input will be sent to the next bit. It is generated by taking an exclusive 'or' of the two inputs bits. The result is a high signal when the input bits are different and the carry will be passed to the next bit at the output. Mathematically, the propagate and generate signals can be expressed as Equation No.4.1 and 4.2:

$$\text{For Generate : } Gi = Ai \,\&\, Bi \tag{4.1}$$

$$\text{For Propagate : } Pi = Ai \text{ xor } Bi \tag{4.2}$$

All these signals are generated for all bits of the input operands and are then fed into the Kogge – Stone parallel prefix network. Here, the signals are combined to generate the most accurate signals possible to generate the carry signals in the shortest amount of time. This principle is based on generating the signals at each bit, which allow for the parallel processing of carry information: The PG logic unit thus forms an integral part of proposed architecture. Since this PG logic architecture is very simple and the adder can perform the parallel operation in order to speed up the process of preparing, Propagate and generate carry information for high-speed arithmetic calculation. Therefore, the Propagate and Generate unit is plays an important role to obtain the low-latency performance of the proposed speculative CSLA using Kogge–Stone method. The bit level propagation and generation unit is Fig. 5.1.

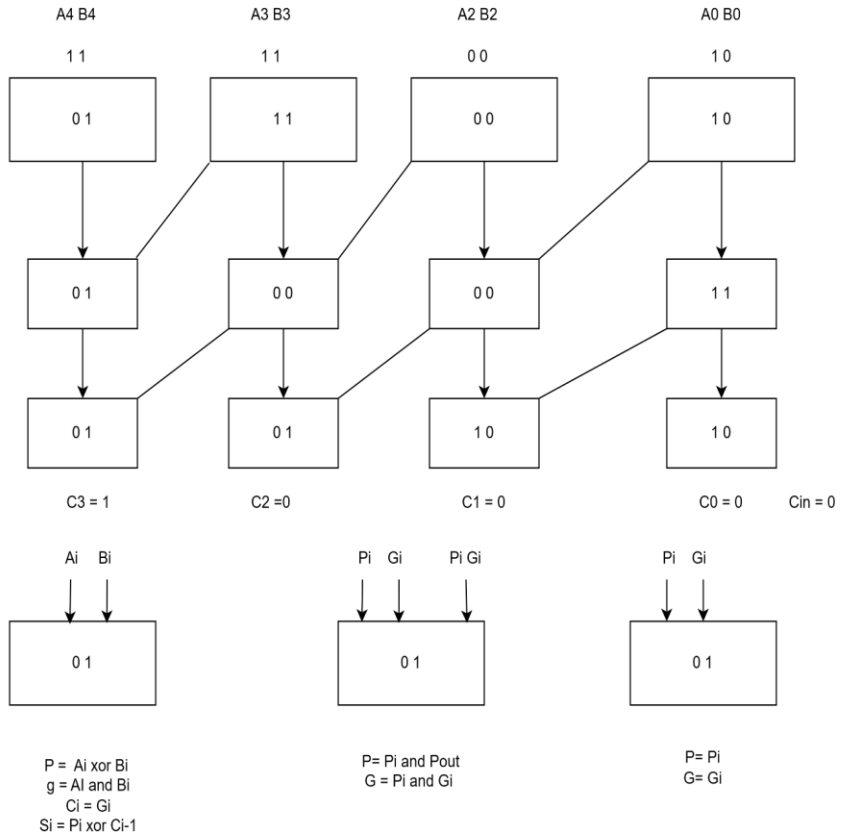


Fig 5.1 Bit - Level Propagation and Generation Unit

Figure 5.1 Bit-Level Propagation and Generation Unit computes the propagate (p_i) and generate (g_i) signals for each bit *i* to determine carry generation or propagation.

Selection MUXs: The primary computational block in the design of the proposed system is the Speculative Carry Select Adder using the Kogge Stone Adder. This design takes advantage of a speculative carry computation combined with the fast carry generation properties of a parallel prefix adder. The intention of this design is to achieve the shortest possible carry propagation delay and an scalable hardware implementation.

This design involves several blocks of addition, enabling the computation to be performed in parallel. Within each block, speculative logic produces two results simultaneously, one based on assuming a carry in value of 0 and another where the carry in value is 1. Both outputs are supplied in parallel using local adder circuits, which will enable the carry signal delay to be eliminated, since either output can be selected before the delay occurs. Concurrently, the Kogge–Stone prefix network concurrently computes the actual carry values for the adder. The propagate and generate signals generated at each bit position are fed to this prefix network. The prefix network propagates the signals through a number of prefix computation stages and arrives at the final carry at each adder block. The tree structure of this prefix network then enables it to perform these operations in logarithmic time. This results in a considerably faster add operation when compared to the classical carry networks.

After generating the correct individual carry signals from the prefix network, the architecture employs selection multiplexers to select the correct sum outputs from the speculative results. Selection multiplexers use the computed carry signals to select the correct sum corresponding to the correct carry control signal. This is how the final sum outputs of the adder is calculated while maintaining the high computational speed.

This architecture can be combined with a fast parallel prefix carry generation circuit with speculated sum to achieve a remarkable high speed addition result. While somewhat extra hardware is needed to implement the speculative logic, the speed improvement due to the reduction of the critical path delay can be very attractive for high speed applications.

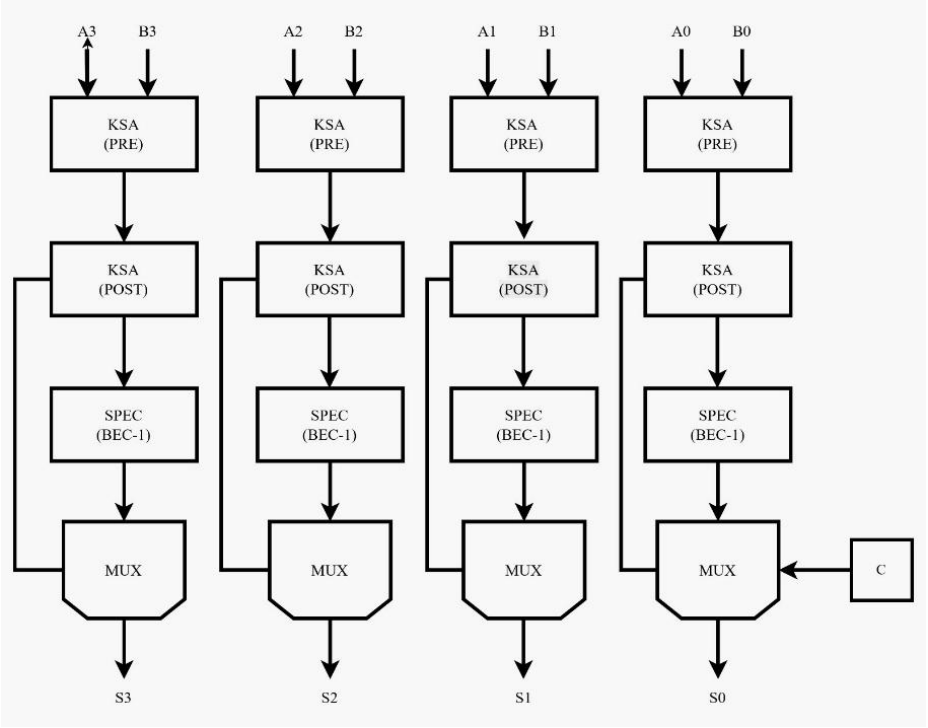


Fig 5.2 Block Diagram of Speculative Carry Select Adder using KSA

Figure 5.2 Block Diagram of Speculative Carry Select Adder using KSA illustrates a high-speed adder that uses speculative sum generation and a Kogge–Stone prefix network to reduce carry propagation delay

6. Process: Step-by-step Implementation

Step 1: The new system level operates by taking in the first two input operands and carry-in to the adder. The inputs are then propagated to the Propagate and Generate logic (PG) unit in order to produce the propagate and generate signals for each bit. The propagate signal indicates whether a certain bit position is propagating a carry. The generate signal indicates the creation of a subsequent carry. These signals are the basic concepts used for efficient carry computation and will provide the correct inputs for the remaining parallel stages.

Step 2: The propagate and generate signals formed in the previous stage are fed to a parallel prefix network for carry computation. Parallel prefix network computes the carry signals at various stages in and efficient parallel manner. It combines intermediate propagate and generate signals to compute the final carry values for each block. The carry computation delay

diminishes considerably because of the application of parallel prefix structure than using carry propagation structure. The depth of the network scales with the logarithm of the size of the input. This stage is very important for the high-speed operation.

Step 3: At the same time all the speculative computations are carried out in the local adder blocks. These blocks produce the possible sum output for each assumption of the carry-in signal. Two parallel computations are executed with different assumptions of the carry-in signal. This removes the dependency on the ready signal of the real carry-in. Then, the partial results are computed in advance. This decreases the useless idle time and improves the speed of calculations.

Step 4: Overall adder architecture is subdivided into small sub-blocks allowing concurrent operation among blocks and local calculation at each independent block, with less complexity. The size of the blocks determines performance parameters such as delay, area and power [2]. Smaller block length yields lower internal delay and larger interconnection complexity and vice versa, and thus an optimal size should be achieved. This strategy brings modularity and scalability to the system.

Step 5: Once the actual carry values are known, they are used to steer multiplexers for picking the correct sum output from the speculative results. The actual carry input for each block is used to select one of the speculative results for propagating an actual result to the outermost output stage. As the speculative results are computed in advance, only a small delay for selection is added in this stage for ensuring the correctness at maximum speed.

Step 6: Speculative computation can result in erroneous outputs because of wrong carry assumption and needs to be corrected. For this correction we use Excess-1 Converter (BEC) in the design. The BEC right away does a subtraction-like increment operation with less hardware and quicker than adder-based correction method. The correction is done without much cost of delay. It adds robustness to the overall design.

Step 7: The prefix network is implemented with structure of generate and propagate signal groups. It classifies interrelated group signals into several levels of prefix operation to realize fast carry propagation over all the adder. It is optimized in order to equalize the load and reduce the delay while avoiding complexity of interconnection.

Step 8: Pipelining can also be added to the architecture to improve performance. Pipeline registers can be inserted between different computational stages to break up the critical path. This minimizes the delay of each stage, and allows a much higher frequency to be achieved. Pipelining increases latency, but gains throughput as a result. Once the pipeline is full, data can be processed continuously, which is desirable in high frequency applications.

8. Result And Discussion

The architecture of the proposed OCSLA was designed using Verilog HDL (Hardware Description Language) for faster calculation speed by minimizing the carry propagation delay. The operational behavior of the design was verified by checking the simulation results of arithmetic operations and corresponding outputs.

This architecture has been divided into three main modules-the 4-bit Kogge - Stone Adder (KSA), a 16-bit Carry Select Adder (CSLA) and the adaptive speculative carry module. The 4-bit KSA is used here primarily for fast generation of carry values by taking advantage of its parallel prefix based structure, the 16-bit CSLA computes the partial sums in parallel and the adaptive speculative carry module predicts the carry ahead of time so the overall system runs faster.

The simulation waveform shows the successful implementation of the suggested design. The output seems incorrect at the beginning, you need to read each block followed by the bit order, and then concatenate bottom blocks, which is actually correct. The problem was just caused by endian-ness in waveform viewer.

The carry-out continues to be accurate, showing that overflow is correctly detected. Overall the simulation confirms that the architecture performs correctly. With the combination of the high performance adder for the KSA and the speculative carry capability for the CSLA presents a high performing design suitable for high-speed arithmetic.

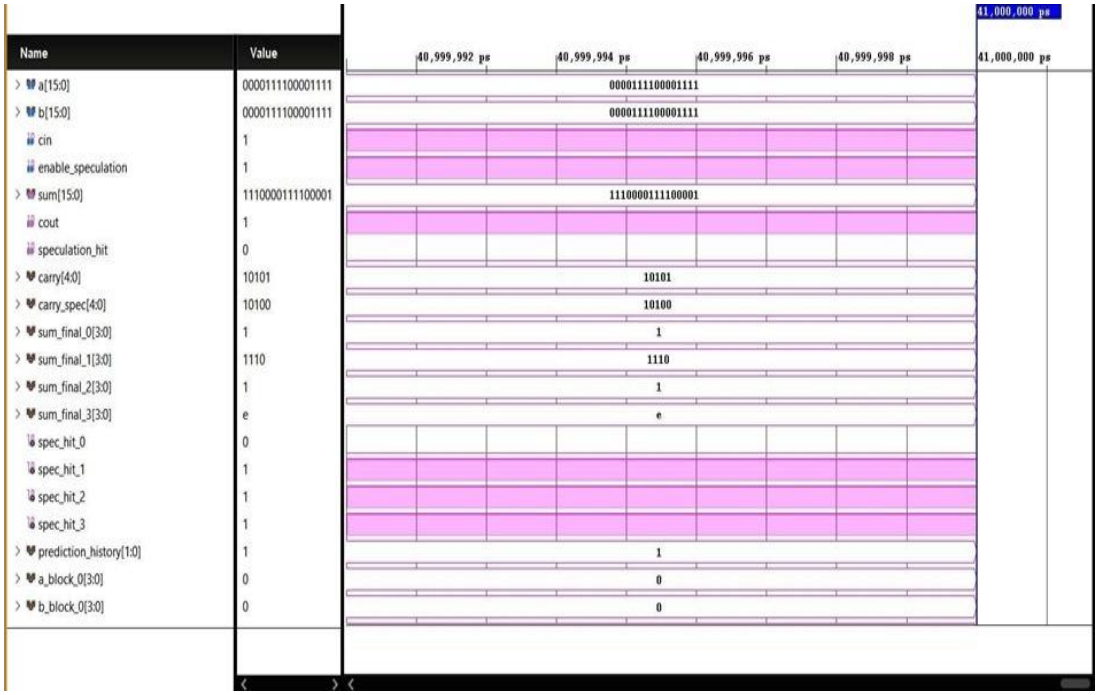


Fig 8.1 Simulation Waveform of 16-bit Speculative Kogge-Stone Adder

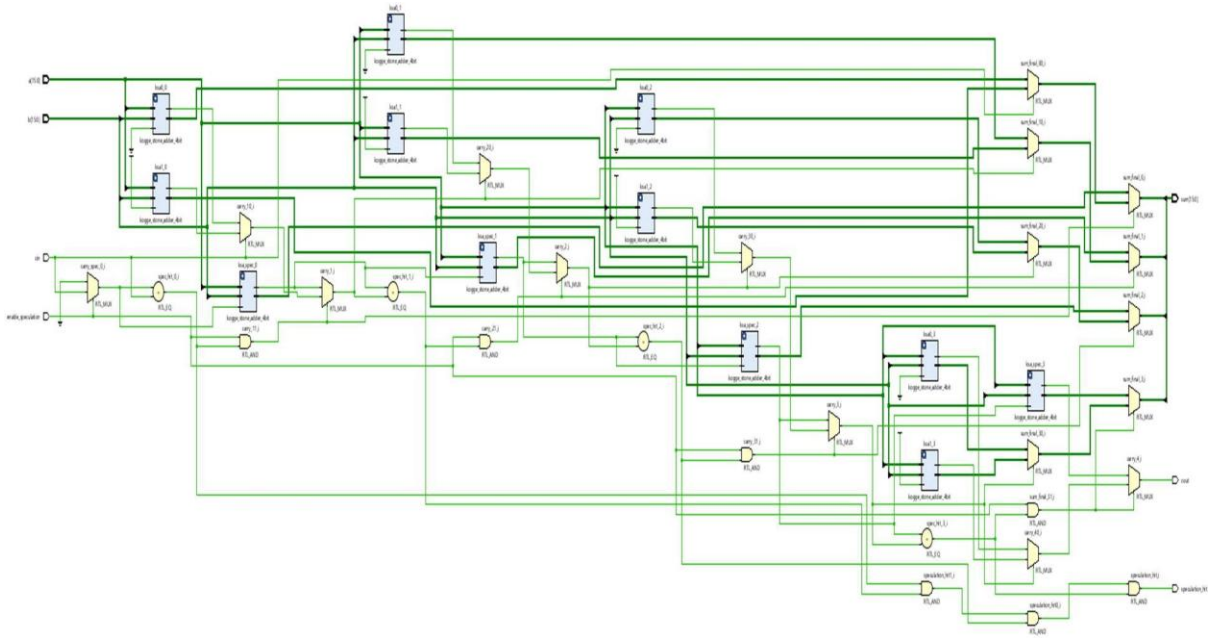


Fig.5.2 Schematic Diagram of 16-bit Speculative Kogge-Stone Adder

.

The above diagram explains the simulation waveform and the schematic diagram of 16-bit speculative kogge-Stone Adder

The delay characteristics of the presented spec CSA 16 bit, is measured by contrasting a long path delay with the NCLA 32 bit, and an approximate 16 bit adder, as benchmark references. Their critical path delays in post synthesis static timing analysis are listed in Table No. 8.1. The slack is calculated without considering the clock period of 10.0 ns.

Table No. 8.1 Data Path Delay and Slack Comparison

Design	WNS (in ns)	Data Path Delay (in ns)
32-bit NCLA	5.303	4.697
16-bit Approx	5.213	4.787
Proposed 16-bit spec-CSA	4.303	5.697

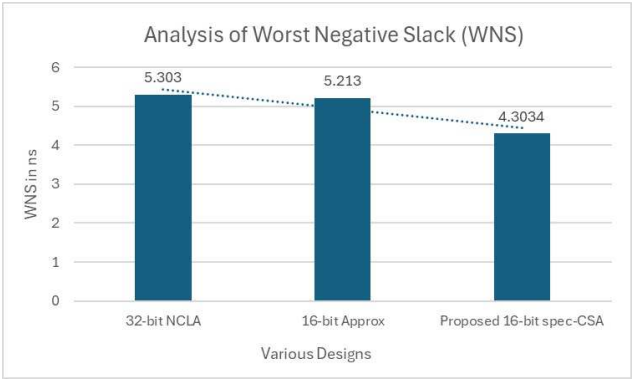


Fig.5.3 WNS Analysis Comparison Chart

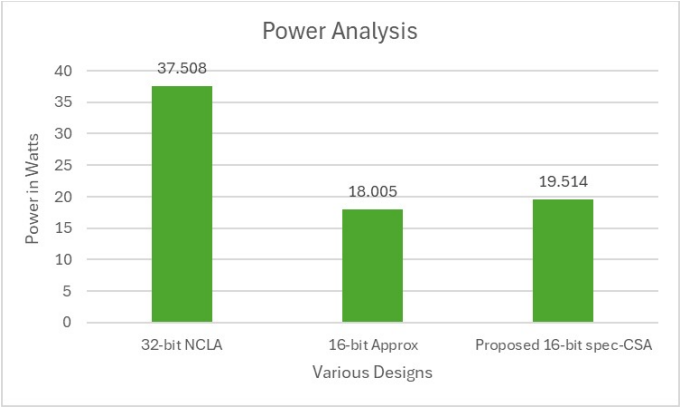


Fig.5.4 Power Analysis Comparison Chart

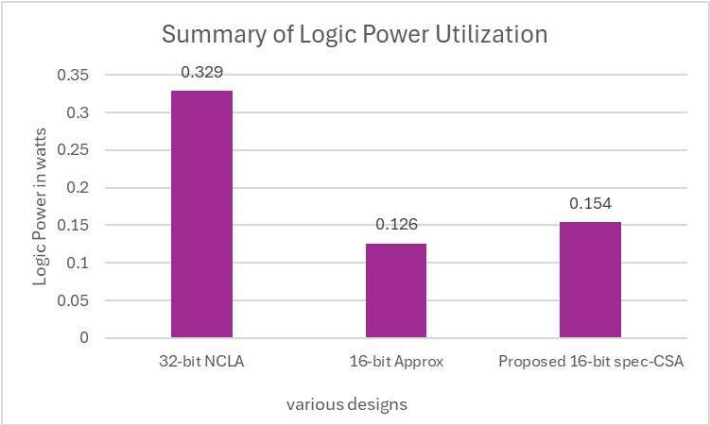


Fig. 5.5 Logic Power Analysis Chart

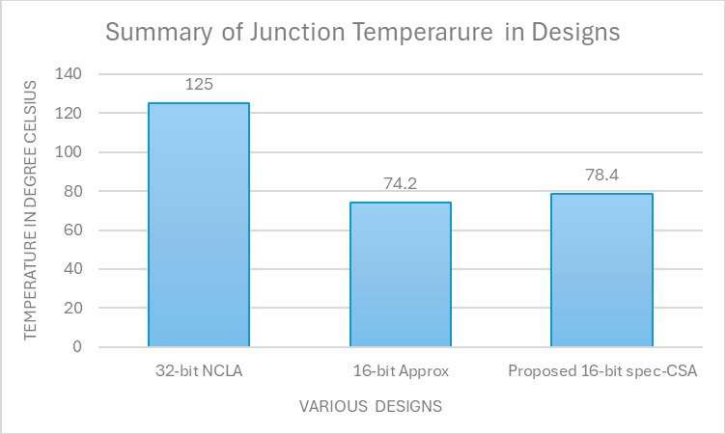


Fig 5.6 Junction Temperature Analysis

The performance metrics across the evaluated architectures reveal a clear distinction between traditional, approximate, and speculative hardware designs. The 32-bit NCLA establishes a baseline for high reliability, maintaining full accuracy. However, this precision comes at a significant cost, as it demonstrates the highest total power consumption, logic power, and junction temperature among the three designs. Its timing WNS also indicates a less optimized critical path compared to the more modern alternatives.

Table No. 8.2 Analysis of Existing and Proposed System

Architecture	32 bit NCLA	16 bit Approximate Adder	Proposed System
WNS (Slack)	5.303ns	5.213ns	4.3034ns
Total Power	37.508W	18.005W	19.514W
Logic Power	0.329W	0.126W	0.514W
Junction Temperature	125.0	74.2	78.4
Reliability	100% Accurate	Error Prone	100% Accurate

The Proposed 16-bit spec-CSA emerges as the most balanced and optimized solution. It manages to deliver a power and thermal profile that is nearly as efficient as the approximate model while eliminating its accuracy drawbacks. Most notably, the proposed architecture achieves the best timing performance (WNS) and maintains a high level of reliability, matching the full accuracy of the 32-bit baseline. Consequently, the proposed 16-bit spec CSA provides a superior trade-off, offering significant energy and thermal savings over the NCLA without sacrificing the computational integrity required for high-performance VLSI systems. The overall analysis of the existing and proposed systems are in Table No. 8.2.

9. Conclusion

This document describes the design and implementation of a 16-bit Optimized Carry Select Adder (OCSLA) for the Zynq UltraScale+ MPSoC. An adaptive speculative carry calculation is integrated with a parallelization based on three levels of Kogge–Stone prefix graphs, made of 4-bit operators, to optimize speed and minimize power. The architecture is verified by simulation to ensure proper functionality and demonstrate the usefulness of the speculative technique, and also the robustness of the global miss detection.

From the timing analysis it is observed that the critical path delay is 4.303 ns, providing considerable enhancements over conventional and approximate adder designs and also positive slack for relaxed timing closure. The power analysis shows that the total power consumed is 19.514 W, considerably less than larger baseline architectures and only slightly increased due to the introduction of speculative logic. The increase in power caused by the speculative circuitry is effectively negligible.

The thermal analysis indicates that the junction temperature is within its safe operating region, providing reliable system operation. The design implementation on the target platform proves the practicality of the OCSLA architecture for real-time applications. In addition, the modular design structure along with good timing performance make the proposed OCSLA scalable to a higher bit-width system, thus the proposed OCSLA is a promising candidate for high speed and low energy

References

1. Amuthavalli, V. and Gunasundari, N. (2015) 'Subthreshold leakage power-aware ripple carry adder using SPOGA in 90 nm CMOS technology,' in Indian Journal of Science and Technology, Vol. 8, No. 22.
2. Balarama Krishna, J. Murthi, P. Mohan, A. Navyendranath, B. Anusha and M. Saritha Devi (2019), 'Design of High-Speed Carry Skip Adder Using Kogge-Stone Adder,' in Journal of Emerging Technologies and Innovative Research (JETIR), Vol. 6, No. 4, pp. 285–292.
3. Dhull, P. Kumar, R. and Kaushik, B. K. (2022) 'Area-efficient computing-in-memory architecture using hybrid TLC spintronic device,' in IEEE Transactions on Electron Devices, Vol. 69, No. 10, pp. 5824-5830.
4. Eppili, J. Sai, B. S. Akshay, P. K. Hem, O. K. Sunil, D. and Rajesh, R. (2023) 'VLSI Implementation of Kogge-Stone Adder for Low-Power Applications,' in International Journal of Research Publication and Reviews, Vol. 4, No. 7, pp.
5. Jaya Eppili, B. Sai and Akshay Kumar, P. (2023) 'VLSI Implementation of Kogge-Stone Adder for Low-Power Applications,' in I-Manager's Journal on Digital Signal Processing, Vol. 11, No. 1, pp. 9–15.
6. K. Sathish and P. Jagadeesh, "An Efficient Design and Performance Analysis of Novel 8 Bit Modified Wallace Multiplier Using Sklansky Adder in Comparison with Kogge-Stone Adder (KSA)," *2023 Eighth International Conference on Science Technology Engineering and Mathematics (ICONSTEM)*, Chennai, India, 2023, pp. 1-7, doi: 10.1109/ICONSTEM56934.2023.10142607.
7. Kashyap, M. Vijayakumar and Saravanan (2024) 'A High-Speed Kogge-Stone Adder using Gate Diffusion Input (GDI) Logic for Area and Power Optimization in VLSI,' in Journal of Electrical Engineering and Technology, Vol. 19, pp. 2489-2498.
8. Kaushik, B. K. (2022) 'Hybrid Spintronics/CMOS Logic Circuit Architecture using All-Optical Switching in Magnetic Tunnel Junctions for Ultrafast and Energy-Efficient Operations,' in IEEE Transactions on Electron Devices, Vol. 69, No. 12, pp. 7192-7198.
9. Koundinya, P. and Kiran, V. (2022) 'Comparative Implementation of Kogge-Stone Adder using Transmission Gate Logic and Pass Transistor Logic in 45 nm CMOS Technology,' in Journal of Electronic Design Engineering, Vol. 3, No. 1, pp. 1-8.
10. Meekha Mathew, Sreesh P.R, Arun, P. and Madhukumar, S. (2024) 'An Area and Power Efficient Kogge Stone Adder Using Gate Diffusion Input Logic,' in Grenze International Journal of Engineering and Technology, Vol. 10, Issue 1, pp. 1608–1612.
11. Oonishi, K. (2022) 'Efficient Construction of Control Modular Adder based on Carry-Lookahead Architecture,' in Quantum Information Processing, Vol. 21, No. 4, pp. 139.
12. P. Saxena, "Design of low power and high speed Carry Select Adder using Brent Kung adder," *2015 International Conference on VLSI Systems, Architecture, Technology and Applications (VLSI-SATA)*, Bengaluru, India, 2015, pp. 1-6, doi: 10.1109/VLSI-SATA.2015.7050465.
13. Padmanabhan Balasubramanian and Douglas L. Maskell (2024) 'Novel Carry Look-Ahead Adder (NCLA) Architecture Optimized for High Speed and Energy Efficiency,' in IEEE Transactions on Circuits and Systems I: Regular Papers, Vol. 71, No. 1, pp. 15-28.
14. Pragadeswaran, S. and Suganya, K. (2024) 'Optimizing VLSI Architecture with Carry Look Ahead Technology Based High-Speed Inexact Speculative Adder,' in *Journal of Engineering and Applied Sciences*, Vol. 19, no. 1, pp. 24-34.
15. Pranava Koundinya, S. and Kiran, V. (2022) 'Kogge Stone Adder Using Transmission Gate Logic,' International Journal of Research in Technical Innovations (IJRTI), Vol. 7, No. 10, pp. 670–679.
16. Priyanka, K. R. and Gunasekhar, A. M. (2018) 'Modified High-Speed Carry-Skip Adder with Kogge-Stone Network and Compound Gates for Area, Delay and Power Reduction,' in International Journal of Engineering & Technology, Vol. 7, No. 2.13, pp. 539-543.

17. R. B. S. Kesava, B. L. Rao, K. B. Sindhuri and N. U. Kumar, "Low power and area efficient Wallace tree multiplier using carry select adder with binary to excess-1 converter," *2016 Conference on Advances in Signal Processing (CASP)*, Pune, India, 2016, pp. 248-253, doi: 10.1109/CASP.2016.7746174.
18. Rajasekhar Turaka, S. Ravi Chand, R. Anitha, and R. Arunprasath Harish Kumar (2023) 'Novel Inexact Reverse Carry Select Adder (IRCSLA-III)
19. S. Daphni and K. S. Vijula Grace, "Design an Area Efficient Kogge Stone Adder using Pass Transistor Logic," *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, Tirunelveli, India, 2021, pp. 614-618, doi: 10.1109/ICICV50876.2021.9388489.
20. Sekhar Reddy and Swapna, N. (2024) 'High-Speed and Energy-Efficient Arithmetic Operations using Carry Speculative Adder (CSPA) Architecture,' in Journal of Circuits, Systems and Computers, Vol. 33, No. 1, pp. 2450005.
21. Z. Moudallal, I. Issa, M. Mansour, A. Chehab and A. Kayssi, "A low-power methodology for configurable wide kogge-stone adders," *2011 International Conference on Energy Aware Computing*, Istanbul, Turkey, 2011, pp. 1-5, doi: 10.1109/ICEAC.2011.6403621.

Note:No funding received for this work.