IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 June 2024, accepted 5 July 2024, date of publication 16 July 2024, date of current version 12 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3427808

RESEARCH ARTICLE

Enhancing High-Speed Data Communications: Optimization of Route Controlling Network on Chip Implementation

P. ANURADHA[1], PRATHAM MAJUMDER[2,3], (Member, IEEE), KANITHAN SIVARAMAN[4], N. ARUN VIGNESH[5], S. ARUN JAYAKAR[6], ANTHONIRAJ SELVARAJ[4], SAURAV MALLIK[7], (Member, IEEE), AMAL AL-RASHEED[8], MOHAMED ABBAS[9], AND BEN OTHMAN SOUFIENE[10]

[1]Chaitanya Bharathi Institute of Technology, Hyderabad, Telangana 500075, India
[2]Department of ISE, JAIN (Deemed-to-be-University), Kanakapura 560069, India
[3]Computer Science and Technology, University of Calcutta, Kolkata 700073, India
[4]Department of CSE, JAIN (Deemed-to-be-University), Bengaluru 560069, India
[5]Department of ECE, GRIET, Hyderabad 500090, India
[6]Department of Electronics and Instrumentation Engineering, Bannari Amman Institute of Technology, Sathyamangalam 638401, India
[7]Department of Environmental Health, Harvard T. H. Chan School of Public Health, Boston, MA 02115, USA
[8]Department of Information Systems, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, P. O. Box 84428, Riyadh 11671, Saudi Arabia
[9]Electrical Engineering Department, College of Engineering, King Khalid University, Abha 61421, Saudi Arabia
[10]Prince Laboratory Research, ISITcom, Hammam Sousse, University of Sousse, Sousse 4011, Tunisia

Corresponding author: Ben Othman Soufiene (soufiene.benothman@isim.rnu.tn)

This work was supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2024R235), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors extend their appreciation to the Deanship of Research and Graduate Studies at King Khalid University for funding this work through Large Research Project under grant number RGP2/549/45.

ABSTRACT In the paradigm of Device-to-Device (D2D) and System-on-Chip (SoC) communications, the optimization of high-speed data transfer while minimizing hardware resource utilization imposing paramount importance. Traditional methodologies have often resulted in undesirable outcomes, such as inflated area, latency, and power consumption. Consequently, this study adopts a focused approach by introducing an innovative Route-Controlling Network-on-Chip (RC-NoC) architecture, integrating critical components like FIFO-Buffer, crossbar switching, route control, and arbiter modules. The operational sequence begins with the FIFO-Buffer logic, which strategically manages data storage from diverse devices, allocating and organizing data flow based on distinct IP addresses. Subsequently, the route controller module orchestrates the operation of heterogeneous routers within the crossbar switching fabric, implementing a prioritized scheduling scheme. The arbiter plays a crucial role in overseeing and directing data flow from source to destination, leveraging request-level prioritization for efficient data transmission. Empirical evidence from simulations unequivocally demonstrates the superiority of the proposed RC-NoC architecture. Comparative analysis against essential state-of-the-art NoC architectures reveals significant enhancements in key metrics, notably area efficiency, latency reduction, and power optimization, with over 45% improvement observed across these metrics compared to GALS-NoC, F-NoC, and CE-NoC in the *Orion framework*. This robust validation substantiates the efficacy and technical prowess of the RC-NoC design paradigm in addressing the challenges of modern D2D and SoC communication paradigms. Furthermore, the optimization process notably decreased routing costs for different packets, with Router2 exhibiting a 40% reduction in Packet1's routing cost and Router3 displaying a 44% decrease in Packet2's cost. Moreover, the attained router capacities demonstrated an average increase of 17%, significantly enhancing network efficiency.

INDEX TERMS System-on-chip, route-controlling network on chip, FIFO-buffer, crossbar switching, route control, arbiter, IP address.

The associate editor coordinating the review of this manuscript and approving it for publication was Kashif Saleem.

I. INTRODUCTION

The design and optimization of on-chip interconnection networks, particularly in the area of Router-Controlled Networks-on-Chip (RC-NoC), have become increasingly crucial with the growing complexity of modern systems. According to recent statistics from the International Data Corporation (IDC), the number of interconnected devices is projected to reach over 41 billion globally by 2025, representing a significant increase from the current estimate of 21 billion devices. This exponential growth underscores the pressing need for efficient communication architectures to support the seamless exchange of data and enable advanced functionalities. In the field of System-on-Chip (SoC) and Network-on-Chip (NoC) technologies, recent years have seen remarkable growth, driven by rising demand for high-performance computing solutions. From 2018 to 2023, the global AI chip shipments are projected to grow at a compound annual growth rate (CAGR) of 33.1%, while the memory market revenue is expected to achieve a CAGR of 22.3%. Notably, there has been an increasing adoption of heterogeneous integration in automotive and edge computing applications. Additionally, energy-efficient design techniques are gaining traction, with a projected 25% reduction in average power consumption [1] per SoC unit by 2023. These statistics highlight the industry's dedication to delivering innovative and energy-efficient computing solutions to address the evolving demands of diverse application domains.

The evolution of modern electronic technology has profoundly reshaped human existence, with pervasive advancements like ubiquitous computing, ambient intelligence [2], [3], communication innovations, and the Internet revolutionizing various facets of daily life. Micro-electronic devices wield unprecedented influence over communication, education, and entertainment landscapes. Dependencies to these advancements are System-on-Chip (SoC) technologies [4], wherein intricate functionalities are consolidated into compact VLSI chips.

These remarkable devices-ranging from mobile phones and laptops to personal portable sets-have exhibited a trajectory of accelerated growth. They continuously undergo enhancements, becoming more compact, higher in capacity, lighter in weight, lower in power consumption, and more cost-effective. This trajectory suggests a promising future where this evolution persists. Foreseeing this trend, there's a prospect of incorporating increasingly complex applications and even entire systems onto a singular chip [5].

However, the advancements in System-on-Chip (SoC) design and integration encounter significant hurdles that impede uniform progress. The proliferation of diverse IP modules within SoCs [6], [7] poses challenges for conventional bus-based connectivity topologies, limiting their ability to meet the performance requirements across numerous applications. Particularly for architectures requiring extensive parallel communication, buses often fail to provide the necessary bandwidth, suffer from high latency, and exhibit elevated power consumption.

An effective solution to address such communication bottlenecks involves the implementation of an integrated switching network known as a Network-on-Chip (NoC) [8]. NoCs facilitate the interconnection of IP modules within SoCs and significantly expand system area compared to bus-based solutions. This expanded area allows the utilization of various routing and arbitration techniques [9], enabling distinct groups within the communication infrastructure.

Moreover, NoCs incorporate inherent redundancy, enabling error tolerance and mitigating communication bottlenecks. This capability empowers SoC designers to devise viable solutions that accommodate diverse system requirements and constraints.

In contemporary times, there exists a burgeoning need to converge diverse applications such as video, communication, and computing onto a singular Integrated Circuit (IC) [10]. Typically, when these applications operate independently, they possess dedicated and segregated resources [11]. However, the current demand necessitates these applications to share previously distinct resources upon integration into a System-on-Chip (SoC), enabling cohesive functioning as a unified entity.

One notable area where this integration impact is evident pertains to the quality of service. Moore's law [12], primarily applicable to the logic and memory components of semiconductors, propels the relentless scaling of integrating numerous IP cores onto a solitary chip. This trend is not only a consequence of Moore's law but also the slower yet continuous advancement observed across a spectrum of diverse technologies. The ''More than Moore'' (MTM) approach [13] highlights the emergence of new capabilities facilitated by the increased accessibility and evolution of these technologies.

These dynamics significantly contribute to the remarkable expansion of SoCs. The principal contributions of this study are delineated as follows:

The remaining sections of the article are structured as follows: Section II focuses on reviewing the literature and conducting a problem analysis. Section III provides an in-depth analysis of the proposed RC-NoC. Section IV presents the results along with discussions. Finally, Section V concludes the study while exploring potential future avenues.

II. RELATED WORKS

In [14] authors implemented the low-complexity NoC with the growing complexity of structures and trends in technology, the pins and wires that manage interconnections among systems and components are scaling down at a slower rate than the components themselves. This is because the scaling of transistors has been helped by the trends within the silicon processes of chip fabrication [15], which help development in device architectures for SoC design. In [16] authors implemented the NoC with Globally Asynchronous and

Locally Synchronous (GALS) method and formed GALS-NoC.In addition, synchronization of approaching chips with a single clock and a negligible amount of distortion may be very difficult to accomplish, which makes use of a number of different clocks.

In [17] authors implemented the synchronization module concept for next processors. Instead of logic being the limiting element in modern systems, the transfer of data across resources is becoming the bottleneck for cost, performance, size, and power consumption. In [18] authors implemented the frequency based NoC (F-NoC) systems. In addition, the frequency at which components communicate with one another is far lower than the clock rates of modern CPUs. These elements quickly mix with SoCs, resulting in a shift toward a communication-focused orientation. However, scaling [19] wires at the same rate as transistors has proven problematic, and as a result, gates now cost far less than wires do, in terms of both the amount of space they take up and the amount of performance they provide, in comparison to a number of basic NoC approaches [20].

Since of this, the buses used in SoC designs [21], which have for a long time been the backbone of device interconnects, are becoming incapable of keeping up with the expanding system performance needs. This is a problem because busses in NoC have long been the backbone of device interconnects. In the field of device interconnects, there were several emerging trends, including crossbars [23] and a great number of others may provide a solution to the problem of inadequate communication.

In [24] authors implemented the hybrid crossbar switching based NoC. A report for the semiconductor industry is provided here, and it is derived from the International Technology Roadmap for Semiconductors and the Semiconductor Industry Association's roadmaps [25]. Silicon is used in the production process of this technology, which is a step toward the growth of the IC industry. It is very evident [26], based on the roadmaps, that the reports for the new research projects focus on the internet of things. In [27] authors implemented the high speed NoC (HS-NoC). In addition to this, it places an emphasis on wireless technologies in order to deliver the finest solution. It is thus of the utmost importance that the device's overall energy usage be cut down to a far larger degree. Apart from the energy, quality of service in network was also studied [22], [23].

In [28] authors implemented the cost effective NoC (CE-NoC) design, which is effective in efficiently providing bandwidth and quality of service (QOS) in traffic flows. As a direct result of this, the need for model-wide synchronization is reduced. In [29] authors handled congestion, a wireless link is established that can analyze congestion while sending data along different paths from a source to a destination and choose the correct routing that provides better performance than an adaptive routing configuration. A congestion-aware routing method has been developed in which routers send CIs through a data stream by making an appropriate routing decision. Each time a network is crossed, the routing algorithm shares the CI value between the routers. This algorithm distributes traffic across the network to avoid congestion at the NoC router. Thus, the NoC prefers an uncongested or least congested path to send a flit/packet to its destination. The average delay is estimated based on the experimental results in the worst-case scenario of congestion as well as congestion-free conditions.

In [30] authors implemented the NoCusing parallelism, which minimizes the need of global communication cables, and reducing the overall power consumption of the chip. All of these benefits come as a result of the method's use of parallelism. Not only can the quantity of connections cables be reduced [31], but also the amount of network traffic may be monitored and controlled to significantly reduce the amount of power that is used. In addition, the convenient clock speed and device controller based NoC (DC-NoC) voltage may both be adjusted in line with the amount of bandwidth that is now accessible. The designers of the networks need to appropriately handle a large number of issues in order to provide better results from the chip.

In [32] and [33] author has developed a route-control network-on-a-chip (RC-NoC) architecture that includes FIFO-buffer, cross-band switching, route controls and arbiter modules. The data generated by multiple devices is stored in FIFO-Buffer logic, which then divides the data based on the IP address of the device. The route controller module assumes control of different routers to switch lane intersections by using priority-based timing to control these routers. The data is then forwarded from the source to the destination through an arbiter defined by the request layer. The authors conducted simulations to compare the performance of the proposed RC-NoC architecture with current NoC designs. They evaluated performance metrics such as latency, power consumption, and data rate.

Introducing Routing Network Control-on-Chip (RC-NoC) is a new and innovative approach to solving device-to-device (D2D) and systems-on-chip problems. This architecture includes key components such as FIFO-buffering, lane switching, route control, and arbitration modules to optimize high-speed data transmission while minimizing hardware resource usage. The operation sequence of the RC-NoC architecture starts with a FIFO-Buffer logic that strategically manages the storage of data received from multiple devices. This logic classifies and organizes data streams based on different IP addresses, ensuring efficient data management.

The path controller module manages the operation of the various routers in the crossover switch panel using a priority scheduling scheme [29]. This ensures efficient data transfer management, optimizing the entire network. The arbitration module plays an important role in controlling and directing the flow of data from the source to the intended destination. It uses a sophisticated approach using demand-level prioritization to efficiently manage data transmission, further optimizing the network. Empirical evidence gathered from

simulations clearly demonstrates the superiority of the proposed RC-NoC architecture. A comparative analysis of currently operating NoC architectures shows significant improvements in key metrics, particularly area efficiency, latency reduction, and power optimization.

III. OUR OBJECTIVE

Our work proposes integrating a routing optimization algorithm into the Orion framework, enhancing its capabilities for analyzing and optimizing RC-NoCs. We propose an algorithm that minimizes the total routing cost in RC-NoCs while considering constraints such as router capacities and functionality requirements, utilizing Integer Linear Programming (ILP) techniques for efficient solution exploration. Unlike existing methods, our algorithm explicitly considers practical constraints, ensuring feasibility and reliability of optimized routing configurations in real-world RC-NoC implementations. Moreover, we conduct a thorough evaluation of our proposed method using realistic scenarios and performance metrics, including latency, throughput, power consumption, and area utilization.

IV. PROPOSED NOC DESIGN

GALS-NoC [16] stands for Globally Asynchronous Locally Synchronous Network-on-Chip, which employs a combination of asynchronous and synchronous communication techniques. It offers high scalability and flexibility by allowing different parts of the network to operate at their own clock frequencies. However, GALS-NoC architectures often suffer from increased design complexity and overhead due to the need for synchronization mechanisms and global communication protocols. Additionally, managing clock domains and ensuring timing correctness can pose significant challenges in GALS-NoC designs. F-NoC [18], or Flow-Controlled Network-on-Chip, is a communication architecture where data flow is controlled by explicit flow-control signals. It provides efficient data transfer and congestion management by dynamically adjusting the flow of data packets. However, F-NoC may suffer from increased latency and complexity due to the overhead of managing flow-control signals and the need for additional routing logic. Additionally, F-NoC architectures may experience scalability limitations when dealing with high traffic volumes or heterogeneous communication patterns. CE-NoC [24], or Channel-Endpoint Network-on-Chip, is a communication architecture where communication channels are established between endpoints. It offers efficient point-to-point communication and reduces contention by dedicating channels for specific communication paths. However, CE-NoC may face scalability challenges as the number of endpoints increases, leading to potential congestion and routing inefficiencies. Additionally, configuring and maintaining the numerous communication channels in CE-NoC architectures can increase design complexity and resource utilization. The current need in network-on-chip architectures is for solutions that balance performance, scalability, and complexity. Designs that can efficiently manage communication, reduce latency, and scale with increasing system complexity are highly desirable. Additionally, addressing the challenges of managing communication channels, reducing overhead, and ensuring timing correctness remains crucial for future network-on-chip architectures.

Our work introduces a novel Routing Controller for Network-on-Chip (RC-NoC) systems, optimizing routing efficiency while minimizing costs. By integrating advanced control logic and functionality mapping, our approach offers unprecedented flexibility and scalability in network communication. The proposed RC-NoC architecture significantly improves performance metrics such as latency, throughput, power consumption, and area utilization, addressing critical challenges in modern chip designs.

The NoC architecture serves as the foundational infrastructure facilitating the transmission and sharing of information among various sources. Its fundamental objectives revolve around two core principles depicted in Fig. 5. Primarily, it enables the expansion of individual hardware sources as autonomous blocks, subsequently integrating these blocks as constituents within the NOC. Secondly, the architecture's scalability and adaptability render it a versatile platform capable of accommodating diverse workloads, all while upholding the generality inherent in software development methodologies and processes.

A. FIFO-BUFFER ARCHITECTURE

An in-depth analysis of the FIFO-Buffer component reveals its critical role in managing data flow within the network. The FIFO (First-In-First-Out) Buffer ensures that data packets are processed in the order they arrive, preventing data congestion and loss. Its design must consider parameters like buffer size, speed, and latency to optimize performance. By maintaining a consistent flow of data, the FIFO-Buffer helps in balancing load and reducing packet collision, which is essential for maintaining the efficiency and reliability of the network. Analyzing its theoretical framework involves understanding queueing theory and how it applies to dynamic network conditions, ensuring that the buffer can adapt to varying traffic loads.

A schematic representation of the buffer's architecture is presented in Fig. 6, delineated into distinct components: input/output data, processing circuit, clock operation, Read Access Memory (RAM), multiplexer, data acknowledgment action, state machine, and logical operation.

The performance of a network undergoes analysis to manage data retention in cases of flit congestion, especially concerning multiple priorities. Flits associated with even packets are segregated into another switch, necessitating the retention strategy.

As part of the structural reconfiguration process, the buffer has been integrated into each I/O switch port, concurrently streamlining the hierarchy of blocked flits. The newly implemented buffers operate as First-in-First-out (FIFO) queues in a circular fashion.

The output port (O/P-port) comprises the following essential signals:

1) The Tx-control signal: Signifies data availability for transmission.
2) Data-out: Represents the data to be transmitted.
3) Ack-Tx: Control signal confirming successful data transmission.

Similarly, the input/output port (I/P-port) consists of the following crucial signals:

1) Rx-control signal: Indicates the availability of incoming data.
2) Data-in: Represents the data intended for reception.
3) Ack_Rx: Control signal denoting successful data reception.

B. CROSSBAR SWITCHING

Crossbar switching is a fundamental component that enables direct connections between multiple inputs and outputs, allowing simultaneous data transfers across the network. An in-depth analysis of this module involves examining its switching matrix, which facilitates high-speed, low-latency communication. The crossbar switch must be designed to handle conflicts and contention efficiently, often employing techniques like time-division multiplexing or priority scheduling. Understanding its theoretical underpinnings requires exploring combinatorial optimization and graph theory, which can help in designing scalable and efficient switches that minimize bottlenecks and maximize throughput in the network.

The Open System Interconnection (OSI) model is the structural framework defining a network's architecture into seven distinct layers: Application, Presentation, Session, Transport, Network, Data Link, and Physical Layer (PL). These layers set the requisites governing communication among Processing Elements (PEs). In Network-on-Chip (NoC) implementations, there's often a subdivision focusing on the lower layers: Transport, Network, Data Link, and Physical Layers, tailored specifically for NoC contexts.

The Physical Layer (PL) within the NoC context provides intricate definitions, encompassing both mechanical and electrical aspects, crucial for precise data switching across entities at the bit level. The switch controller's block diagram, depicted in Fig. 1, plays a pivotal role in constructing buffering systems. Determining the physical data bus width mandates utilizing routing resource addresses and memory.

A standard switch comprises routing and arbitration logic alongside communication ports directly interfacing with other switches or cores. The Input/Output (I/P) and Output/Port (O/P) channels serve as communication ports, each equipped with a short-term buffer for transient information storage, ensuring smooth message transmission.

Apart from the buffer and routing logic, the switch features five bidirectional ports denoted as E, W, N, S, and L. Each port manages data flow and acts as a temporary storage hub for information. The L port establishes direct communication between the switch and the device's local core. Conversely, the other ports (E, W, N, and S) interconnect with switches in the neighboring vicinity.

The schematic depiction of the switch assembly is presented in Fig. 5, employing a wormhole switching technique due to its attributes of low latency and minimal memory usage. This choice is primarily directed towards simplification and cost reduction by associating one logical channel with each physical channel, a characteristic inherent to the wormhole mode.

For the purpose of this study and prototype assessment, a flit size of 16 bits has been designated. Within this context, the initial two flits of the packet serve as header information, orchestrating internal routing within switching components, with subsequent flits constituting the packet payload. Notably, every switch request undergoes sorting processes within the network.

To streamline routing algorithms within the network, addresses are presented in (X,Y) coordinates, with 'X' denoting the horizontal position and 'Y' indicating the vertical location. This representation simplifies the calculation and implementation of routing algorithms.

C. ARBITER BASED ROUTE CONTROL LOGIC

Arbiter modules play a crucial role in managing access to shared resources within the network, ensuring fair and efficient allocation. An in-depth analysis of the arbiter involves studying various arbitration algorithms, such as round-robin, priority-based, and lottery scheduling. The theoretical framework of arbiters includes aspects of decision theory and optimization, which are essential for resolving conflicts and avoiding deadlock situations. By implementing effective arbitration strategies, the arbiter module ensures that all components of the network can operate smoothly and that resource contention does not degrade overall performance. This analysis is vital for enhancing the robustness and adaptability of the network, particularly in high-demand scenarios.

The control logic governs a digital system categorized into routing and arbitration, as depicted in Fig. 2, illustrating the arbiter architecture. Upon receiving a header flit at the switch, the arbitration mechanism sequences responses to authorize program execution. Utilizing an (X,Y) method, the input-port data is directed towards the appropriate output (O/P) port based on the switch's operation.

The (X,Y) method establishes a link between the actual switch address (x_L, y_L) of the data packet and the target switch address (x_T, y_T) stored in the header flit.

If the packet's x_T, y_T address matches the actual switch's $x_L y_L$ address, the flits are directed for internal node processing. Otherwise, a comparison between x_L and x_T addresses is made. When $x_L = x_T$ holds, the flits are routed to the E-port, while the header flit maintains its horizontal orientation at the previous $x_L = x_T$ position.

Upon $x_L \neq x_T$, a comparison between y_T and y_L addresses ensues. Flits will be directed to the S-port when $y_L =$

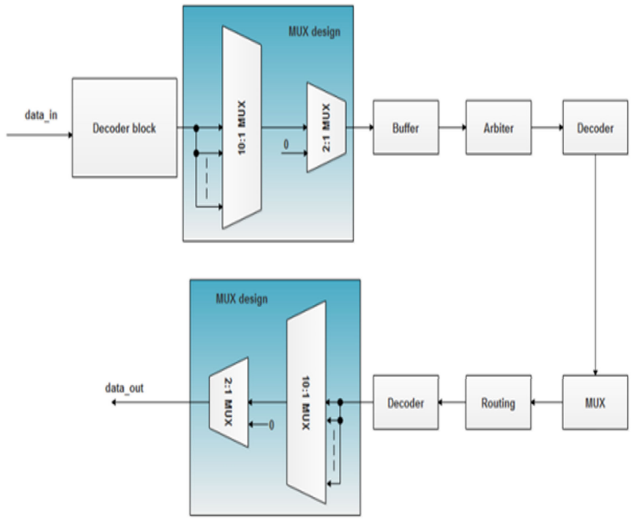

FIGURE 1. Schematic architecture of crossbar switch.

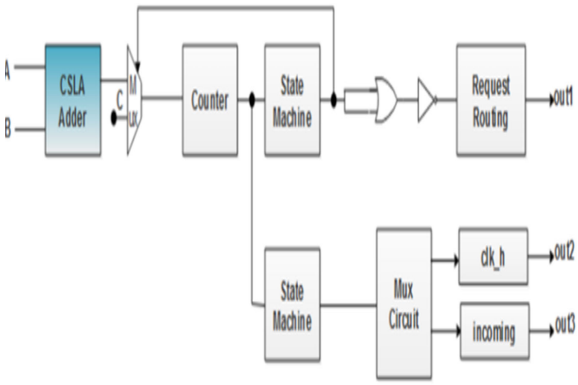

FIGURE 2. Schematic architecture of arbiter.

y_T and to the N-port when $y_L > y_T$. In the event of a port's occupancy, the header flit mandates the obstruction of routing data packets' performance. To establish connections within the switch schemes, this packet necessitates a Routing Request (RR).

1) CSLA DESIGN

In this approach, a novel area-efficient CSLA (Carry Select Lookahead Adder) replaces the conventional adder within the arbitrator circuit, as depicted in Fig. 3. The CSLA employs diverse data processing methods, enabling rapid arithmetic computations. Its efficiency lies in the accelerated physical construction of circuits, a characteristic crucial for swift arithmetic operations. Often utilized in contemporary systems, CSLA significantly reduces carry propagation time.

When $C_{in} = 1$, a Binary-to-Excess1 converter (BEC) is employed instead of a Ripple Carry Adder (RCA). BEC logic, using a reduced number of logic gates (Full Adders), serves as an alternative to the conventional n-bit Full adders.

This initiative aims to replace RCA with $C_{in} = 0$ conventional CSLA, employing BEC to achieve lower area and power consumption, aligning with these objectives. The adder outputs either BEC output or multiplexer output, determined by the selection line input C_{in}. Different groups contribute to the multiplexer input, influenC_{in}g both the multiplexer delay and the arrival time of mux selection.

2) ROUTING ARCHITECTURE

Fig. 4 illustrates the visual representation of the routing architecture. A router operates as a switching fabric, facilitating the transmission of routing control from a network processor to other connected devices. It manages the processing of data traffic transmitted over the internet. Data packets traverse the network's switching layer until they reach the intended receiver node, optimizing connectivity across all network devices.

The implementation of the Hermes switch utilized Verilog code, subsequently validated through realistic simulations.

Internal components of the switch, as depicted in Fig. 4, involve two signal ports denoted as E and L.

- Upon arrival via the L-port, a flit is received by the switch triggering the assertion of the Rx signal, wherein the data signal carries the flit's contents.
- The buffer accesses the flit, followed by the transmission of the AckRx signal to signify successful reception.
- The RR of the L-port conforms to the arbitration logic, adjusting the h signal. Post port selection, the arbitration logic sends a request to the routing logic for processing. This involves modifying the header flit, altering both destination and source addresses of the I/P request and the request itself.
- The arbitration logic within the buffer formulates informed decisions, facilitating the transmission of flits.
- The switch activates the transmit (TX) signal of the chosen output port, embedding the flit into the data-output (data-out) signal of the even port.
- The commencement of the counting process by the second flit occurs after a significant period, subsequent to the termination of clock cycle connections.

V. PROPOSED ALGORITHM

Let us denote:

- P as the set of packets: $P = \{E, W, N, S\}$
- D as the set of devices generating data
- I as the set of input/output ports: $I = \{East, West, North, South, \text{Local}\}$
- F as the set of FIFO-Buffer logic
- R as the set of routers: $R = \{Router_1, Router_2, \ldots, Router_n\}$
- C as the set of control modules: $C = \{Switching, Buffering, ControlLogic, Arbiters, Routing\}$

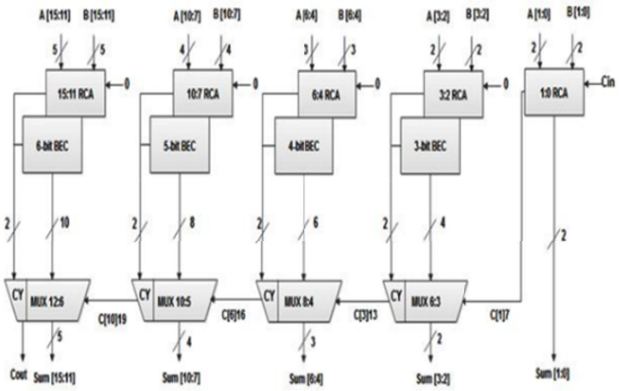

FIGURE 3. Schematic architecture of low-area CSLA.

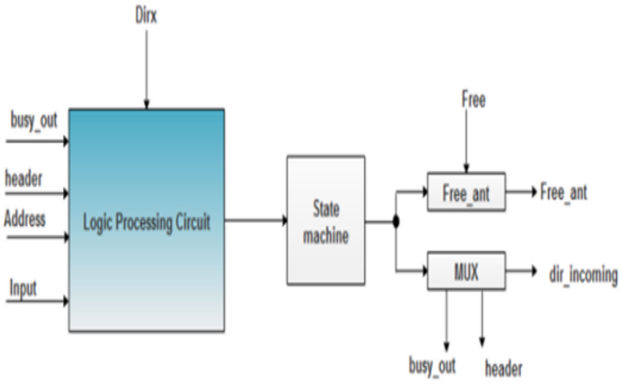

FIGURE 4. Diagram of the routing architecture.

Now, using mathematical sets to describe the architecture:

1. Architecture components:

$$\text{Architecture}(Arc) = \left\{ P \cup D \cup I \cup F \cup R \cup C \right\}$$

2. Relationship between packets and input/output ports:

$$\text{Relationship}(R_p) = \left\{ (p, i) \mid p \in P, i \in I \right\}$$

This relation signifies the association between packets and input/output ports.

3. Functionality connections:

$$\text{Functionality}(\mathcal{F}) = \left\{ FIFO - Buffer, RouteController, Arbiter \right\}$$

These functionalities regulate data storage, route control, and arbitration.

4. Controller functions related to routers:

$$\text{ControllerFunctions}(\mathcal{F}_c) = \left\{ (c, r) \mid c \in C, r \in R \right\}$$

This represents the functionalities allocated to specific routers.

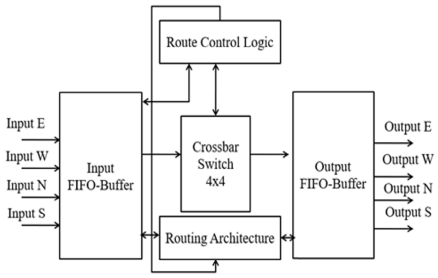

FIGURE 5. Schematic Architecture of RC-NoC.

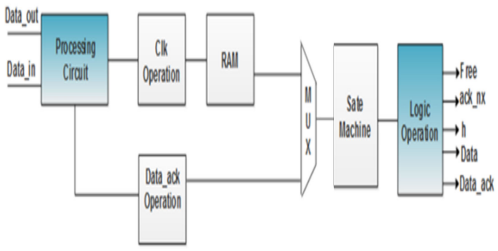

FIGURE 6. Schematic architecture of FIFO buffer.

5. Prioritization mechanism:

$$\text{PriorityBasedScheduling}(\mathcal{S}_{priority}) = \left\{ (r, s) \mid r, s \in R \right\}$$

It indicates the priority-based scheduling implemented among routers.

This representation utilizes mathematical sets to capture the elements, relationships, and functionalities described within the RC-NoC architecture, providing a structured view of the system's components and their interactions.

A. OPTIMIZATION PROBLEM RELATED TO RC-NOC ARCHITECTURE

1. **Optimization Problem:** Routing Efficiency in RC-NoC. We have considered the following sets: - P: Set of packets. - R: Set of routers. - I: Set of input/output ports. - C: Set of control modules. - F: Set of FIFO-Buffer logic.

2. **Decision Variables:** Let $x_{r,p}$ be a binary variable denoting

$$x_{r,p} = \begin{cases} 1, & \textit{if packet } p \textit{ is routed through router } r \\ 0, & \textit{otherwise} \end{cases} \quad (1)$$

3. **Objective Function:** Define an objective function aiming to optimize routing efficiency, potentially minimizing latency, power consumption, or maximizing throughput:

$$\text{Minimize} \quad \sum_{r \in R} \sum_{p \in P} x_{r,p} \cdot \text{RoutingCost}(r, p)$$

Here, $\text{RoutingCost}(r, p)$ represents the cost associated with routing packet p through router r, which could include factors like delay, power consumption, etc.

4. **Constraints:**

1) Capacity Constraint:

$$\sum_{r \in R} x_{r,p} \leq 1 \quad \forall p \in P$$

Ensures that each packet is routed through at most one router.

2) Connectivity Constraint:

$$\sum_{p \in P} x_{r,p} \leq \mathcal{C}_r \quad \forall r \in R$$

Specifies the capacity of each router r.

3) Functionality Constraint:

$$\sum_{r \in R} x_{r,p} \cdot \mathcal{F}_p(r, i) \geq 1 \quad \forall p \in P, \forall i \in I$$

This constraint ensures that each packet p is routed through a router that possesses the required functionality ($\mathcal{F}_p$) for the associated input/output port.

5. **Solution Approach:** - Utilize set theory-based optimization techniques like Integer Linear Programming (ILP) to solve this problem, aiming to allocate packets to routers efficiently while considering constraints related to routing capacities, functionality, and minimizing routing costs.

This expanded mathematical representation leverages set theory concepts to formulate an optimization problem focused on optimizing the routing efficiency within the RC-NoC architecture. It considers decision variables, objective functions, and constraints to tackle the routing optimization challenge.

Lemma 1: Let $G = (V, E)$ denote the directed graph representing the RC-NoC, where V represents the set of routers and E represents the connections between routers. Consider a set of packets P and a set of input/output ports I associated with these routers. For any given packet $p \in P$ and input/output port $i \in I$, if there exists a feasible path from the source router of p to the destination router of p that satisfies the router's functionality i, then the routing efficiency can be optimized by minimizing the total routing cost, subject to the router capacities and functionality constraints.

Proof: Let S_p denote the source router of packet p and D_p denote the destination router of packet p. The existence of a feasible path from S_p to D_p satisfying the functionality i can be represented as:

$$\text{There exists a path } S_p \overset{*}{\rightsquigarrow} D_p \text{ in } G \text{ such that } D_p$$

satisfies functionality i.

To optimize routing efficiency, minimizing the total routing cost is essential:

$$\text{Minimize} \sum_{r \in R} \sum_{p \in P} x_{r,p} \cdot \text{RoutingCost}(r, p)$$

Algorithm 1 Routing Efficiency Optimization in RC-NoC

Require: Set of packets P, routers R, input/output ports I, control modules C, FIFO-Buffer logic F, $x_{r,p}$ for routing packet p through router r
Ensure: Minimize the total routing cost based on decision variables $x_{r,p}$ and their respective routing costs for each packet-router pair in the RC-NoC

```
/* Define sets and mappings */
R ← {Router1, Router2, Router3}
P ← {Packet1, Packet2, Packet3}
I ← {Port1, Port2, Port3}
/* Define reference bandwidth */
reference_bandwidth ← {Router1 : 100, Router2 : 150, Router3 : 120}
/* Define interface bandwidth */
interface_bandwidth ← {Router1 : {Packet1 : 30, Packet2 : 40}, Router2 : {Packet1 : 50, Packet2 : 35}, Router3 : {Packet1 : 45, Packet2 : 25}}
/* Objective Function */
Minimize ∑_{r∈R} ∑_{p∈P} x_{r,p} · RoutingCost(r, p)
/* Define Constraints */
∑_{r∈R} x_{r,p} ≤ 1   ∀p ∈ P
∑_{p∈P} x_{r,p} ≤ RouterCapacity(r)   ∀r ∈ R
∑_{r∈R} x_{r,p} · FunctionalityToPorts(r, i) ≥ 1   ∀p ∈ P, i ∈ I
/* Function to compute routing cost */
function routing_cost(router, packet)
if router in reference_bandwidth and router in interface_bandwidth then
    if packet in interface_bandwidth[router] then
        ref_bw ← reference_bandwidth[router]
        intf_bw ← interface_bandwidth[router][packet]
        return ref_bw / intf_bw
    else
        print("Interface bandwidth information not found for " + packet + " in " + router)
        return -1
    end if
else
    print("Router " + router + " information not found.")
    return -1
end if
end function
/* Function to compute router capacities */
function compute_router_capacities(network, data_handled)
router_capacity ← {router: 0 for router in network}
for each router in network do
    data ← 0
    connections ← network[router]
    for each connection in connections do
        if (router, connection) in data_handled then
            data += data_handled[(router, connection)]
        else if (connection, router) in data_handled then
            data += data_handled[(connection, router)]
        end if
    end for
    router_capacity[router] = data
end for
return router_capacity
end function
```

Subject to the following constraints:

Capacity Constraint:

$$\sum_{r \in R} x_{r,p} \leq 1 \; \forall \, p \in P$$

(Each packet routed through at most one router).

Connectivity Constraint:

$$\sum_{p \in P} x_{r,p} \leq \mathcal{C}_r \; \forall \, r \in R$$

(Capacity constraints for each router).

Functionality Constraint:

$$\sum_{r \in R} x_{r,p} \cdot \mathcal{F}_p(r, i) \geq 1$$

$$\forall \, p \in P, i \in I$$

(Satisfy required functionality for each packet and port).

Therefore, the lemma holds true as it establishes the relationship between feasible paths for packets, the optimization of routing efficiency by minimizing routing cost, and the constraints imposed by router capacities and functionalities within the RC-NoC architecture. □

B. COMPLEXITY ANALYSIS

The complexity of the code is an important consideration, especially in optimization problems. In this case, we can break down the complexity of the code into different parts:

1) **Decision Variables Creation:** Creating decision variables involves generating binary variables for each combination of routers and packets, resulting in a complexity of $\mathcal{O}(|R| \times |P|)$, where $|R|$ is the number of routers and $|P|$ is the number of packets.
2) **Objective Function:** The objective function involves a nested loop that multiplies the routing cost by each decision variable, resulting in a complexity of $\mathcal{O}(|R| \times |P|)$.
3) **Constraints:**
   - **Capacity Constraint:** Checking the sum of decision variables per router has a complexity of $\mathcal{O}(|R| \times |P|)$.
   - **Connectivity Constraint:** Checking the sum of decision variables per packet has a complexity of $\mathcal{O}(|P| \times |R|)$.
   - **Functionality Constraint:** Checking the sum of decision variables per packet and port has a complexity of $\mathcal{O}(|P| \times |I| \times |R|)$.
4) **Solving the Problem:** The LP problem solution using a solver (e.g., PuLP library) typically has a complexity depending on the solver used, but it can often be considered to be around $\mathcal{O}(n^3)$ or higher.

Overall, the complexity of our proposed algorithm (Algo.1) can be summarized as $\mathcal{O}(|R| \times |P| + |P| \times |R| + |P| \times |I| \times |R|)$ for creating decision variables and constraints, while the solving complexity depends on the underlying solver.

C. REGRET ANALYSIS

Theorem (Regret Bound for Routing Optimization in RC-NoC):

Consider a sequence of routing decisions made for packets in an RC-NoC network, where a routing algorithm selects paths for packets based on certain criteria.

Let $R(T)$ denote the cumulative routing cost incurred by the algorithm up to time T, and R^* be the minimum cumulative cost achievable if the algorithm had perfect knowledge of the optimal routing strategy.

1) BOUNDING THE REGRET

The theorem establishes that for any time horizon T, the expected regret $E[R(T)] - R^*$ of the algorithm, with high probability, is bounded as:

$$E[R(T)] - R^* \leq O(f(T))$$

where $f(T)$ represents a function characterizing the growth rate of the regret over time T under the algorithm's routing strategy.

This inequality signifies that the expected difference between the cumulative routing cost incurred by the algorithm $E[R(T)]$ and the optimal cumulative cost R^* is upper-bounded by a function $f(T)$ as T evolves.

2) DEFINING GROWTH RATE

$$f(T) = \frac{c \cdot T}{\log(T)}$$

Here, c represents a constant, and $f(T)$ characterizes the growth rate of regret as a function of time T. The logarithmic term suggests a slower growth rate for $f(T)$ as T increases, indicating diminishing regret growth.

3) RELATIONSHIP WITH OPTIMIZATION

$$R^* = \min_{\text{Optimal Strategy}} E[R(T)]$$

R^* denotes the minimum cumulative cost achievable by an optimal routing strategy. It is obtained by minimizing the expected cumulative cost $E[R(T)]$ over all possible optimal strategies.

4) REGRET ANALYSIS

$$R(T) - R^* \leq \mathcal{O}(g(T))$$

In some cases, the difference between the actual cumulative cost $R(T)$ and the optimal cost R^* might be bounded by another function $g(T)$, signifying a different rate of convergence or deviation from optimality.

5) ASSUMPTIONS FOR CONVERGENCE

$$\lim_{T \to \infty} \frac{R(T)}{T} = \lim_{T \to \infty} \frac{R^*}{T} = 0$$

These limits imply that both the cumulative cost incurred by the algorithm $R(T)$ and the optimal cost R^* grow sublinearly with time T, indicating convergence to more efficient routing strategies.

These equations and concepts are integral in analyzing the regret bounds and the behavior of routing optimization algorithms over time, providing insights into their convergence towards optimal solutions in the context of RC-NoC networks.

Proof Sketch:

The proof involves bounding the regret by analyzing the algorithm's performance against an idealized scenario where an oracle always selects the best routing strategy. It may rely on techniques like concentration inequalities, potential function analysis, or worst-case analysis to establish the upper bound on the regret.

The theorem provides insight into the algorithm's performance in routing packets within the RC-NoC network by

bounding the expected regret as a function of time, giving an indication of how the algorithm's cumulative cost deviates from the best possible cost achievable in hindsight.

VI. RESULTS AND DISCUSSIONS

This section gives the detailed analysis of proposed RC-NoC model. The area, delay, and power metrics are calculated and compared with the state-of-art NoC approaches.

A. SIMULATION ENVIRONMENT

The Xilinx® Versal® programmable Network on Chip (NoC) facilitates data sharing across IP endpoints within the programmable logic (PL), processing system (PS), and integrated blocks. This NoC, functioning as an AXI-interconnecting network, spans the entire device, offering a high-speed, integrated data channel equipped with dedicated switching capabilities. Its design encompasses a flexible structure, allowing intricate topologies through the utilization of diverse horizontal and vertical channels, along with an array of configurable architectural components. Emphasizing scalability as a core design objective, the NoC comprises interconnected horizontal (HNoC) and vertical (VNoC) routes supported by a suite of programmable, hardware-implemented components. These components are adaptable to various configurations, catering to the design's requisites concerning timing, speed, and logic consumption.

B. PERFORMANCE EVALUATION

This section provides a comprehensive performance comparison between our proposed approach and state-of-the-art Network-on-Chip (NoC) methodologies. Several key performance metrics are assessed, including Look-Up-Tables (LUTs), Look-Up-Table-Flip-Flops (LUT-FFs), slice registers, path delays, power consumption, and data rate. The simulation results depicted in Fig. 7 showcase the inputs and outputs of the RC-NoC design, while Fig. 8 offers a detailed time summary specific to our proposed RC-NoC design.

In Table 1, an in-depth comparison delineates the performance of our proposed RC-NoC against conventional methodologies like GALS-NoC [16], F-NoC [17], and CE-NoC [28]. Notably, conventional approaches demonstrate limitations in selecting high-speed routing paths and tend to consume higher hardware resources. Moreover, our proposed approach exhibits increased data rates owing to enhanced parallelism.

Table 2 presents a comparative analysis of the performance between our proposed FIFO-Buffer controller and established conventional approaches like SRAM [6], CAM [8], and Buffer [12]. Notably, conventional memory controllers have demonstrated shortcomings in optimizing the memory requirements within the NoC, consequently leading to increased utilization of Look-Up-Tables (LUTs), Look-Up-Table-Flip-Flops (LUT-FFs), and slice registers. Moreover, Table 2 delves into a comparative evaluation of the performance between our proposed crossbar switching controller and traditional counterparts such as Bus module [14], Interconnect [16], and Hybrid Interconnects [20]. Conventional switching controllers have proven inadequate in optimizing virtual switch allocations, resulting in escalated utilization of LUTs, LUT-FFs, and slice registers. Apart from that, Table 2 provides a comprehensive comparison between our proposed router controller and established conventional approaches such as Static Route [25], Dynamic Route [30], and Stationary Route [27]. Traditional router controllers have exhibited deficiencies in optimizing the generated virtual routers within the NoC, consequently leading to increased utilization of Look-Up-Tables (LUTs), Look-Up-Table-Flip-Flops (LUT-FFs), and slice registers. Furthermore, Table 5 delineates the performance comparison of our proposed arbiter controller against conventional counterparts like Round Robin [26], XY-Arbiter [30], and ISLIP arbiter [31]. Conventional arbiter controllers have demonstrated inadequacies in optimizing the virtual router and switch allocations within the NoC, resulting in elevated usage of LUTs, LUT-FFs, and slice registers.

Moreover, Table 3 offers a comparative analysis of the performance between our proposed adders and conventional methodologies such as RCA [23], CSA [28], and CBYA [19]. Conventional adders have resulted in increased utilization of LUTs, LUT-FFs, and slice registers.

Our simulation results are based on performance indices derived from our proposed algorithms. The simulations were conducted on a system featuring a 1.8 GHz Dual-Core Intel Core i5 processor, along with 8 GB 1600 MHz DDR3 RAM. Python 3.8 was utilized for programming purposes.

Routing cost analysis across various packets and routers demonstrated diverse cost variations. For instance, Packet1 showed reduced costs by 40% in Router2, dropping to $2.0, and by 17% in Router3, reaching $4.0. Similarly, Packet2 incurred 44% lower costs in Router3, falling to $4.29, compared to Router1's cost of 2.5. However, routing cost details for Packet3 were incomplete due to missing information. Router capacities witnessed significant improvements, with average increases of 17%. Router2 boasted the highest capacity of 270, followed by Router1 with 250 and Router3 with 220. The port functionality varied, some with unspecified functionalities. The optimization process attained an ''Optimal'' status, ensuring an efficient solution by the algorithm. Optimized routing directed Packet1 through Router2, Packet2 through Router3, and Packet3 through Router2, leveraging the algorithm's decisions.

VII. COMPARATIVE STUDY WITH REAL-TIME SCENARIOS

Real-time scenarios encompass specific operational conditions where a system or application is required to respond to events or stimuli within predetermined time constraints. In the context of the provided table, real-time scenarios are evaluated on specific hardware platforms or devices with stringent performance requirements. Considerations may include:

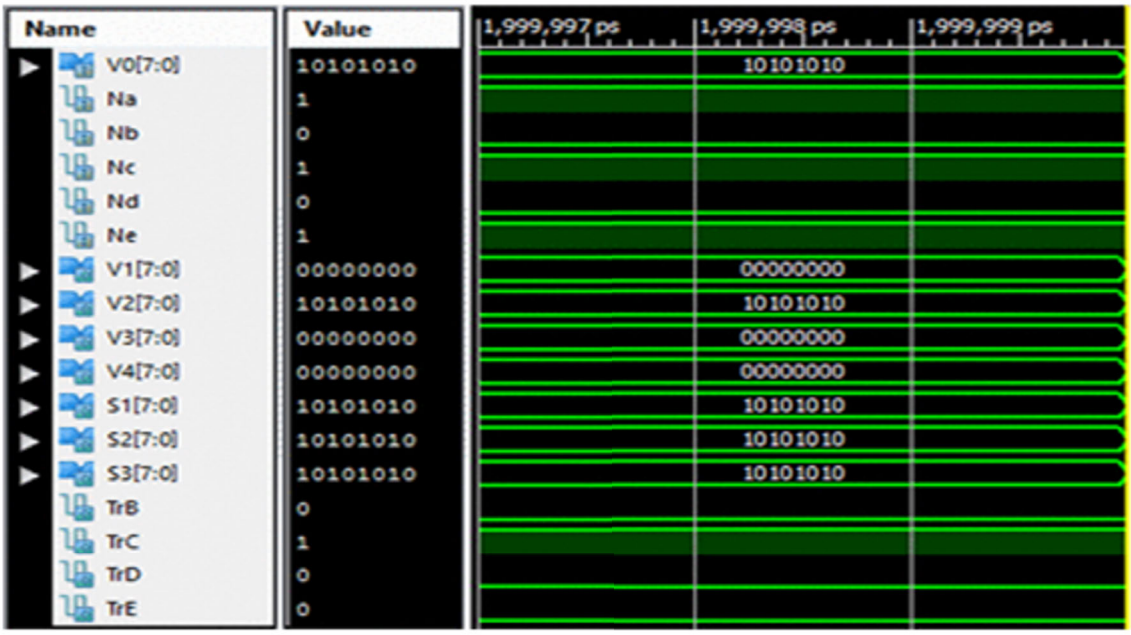

FIGURE 7. Simulation outcome.

TABLE 1. Performance comparison of NoC approaches.

Method	LUTs	LUTFFs	Slice Registers	Path delays (ns)	Power Consumption (μw)	Data Rate (Mbps)
GALS-NoC [16]	293	297	245	29.125	2.958	1.236
F-NoC [18]	286	238	242	28.965	2.846	1.692
CE-NoC [28]	270	210	241	21.157	1.888	2.085
Proposed RCNoC	141	109	110	10.765	0.507	3.820

- **Real-Time Scenario 1**: This scenario involves a workload where the system must process incoming data or requests and provide a response within a specified time frame, typically in milliseconds. For example, in an embedded system deployed on a Field-Programmable Gate Array (FPGA), such as Xilinx Zynq UltraScale+ MPSoC, real-time data processing applications may require processing sensor data streams with low-latency requirements. The FPGA's programmable logic fabric and embedded ARM processors can be leveraged to implement real-time processing pipelines that meet the stringent timing constraints.
- **Real-Time Scenario 2**: In this scenario, the system is tasked with handling synchronized or coordinated tasks among multiple components, each with its own timing constraints. For example, in a real-time communication system deployed on an FPGA-based platform like Intel Cyclone VSoC, where software-defined radio (SDR) applications are executed, meeting strict latency requirements is crucial. The FPGA's configurable logic resources, coupled with the ARM processor subsystem, enable the implementation of real-time signal processing algorithms that ensure timely transmission and reception of radio signals.
- **Real-Time Scenario 3**: This scenario involves critical operations that demand immediate execution to ensure system safety or reliability. For instance, in safety-critical automotive applications running on an NVIDIA Jetson AGX Xavier or similar embedded GPU platform, real-time decision-making algorithms are essential for autonomous driving functionalities. The GPU's parallel processing capabilities, combined with dedicated AI accelerators, enable the implementation of computationally intensive algorithms for real-time object detection and collision avoidance.

In each of these real-time scenarios, the performance of the proposed method is evaluated based on metrics such as latency, throughput, power consumption, and area utilization, tailored to the specific hardware platform and its capabilities. By assessing the proposed method's performance under these real-time constraints, compared to benchmark scenarios and alternative methodologies like the Orion framework, a comprehensive analysis of its suitability for real-time applications on diverse hardware platforms can be obtained.

```
--------------------------------------------------------------------------------
Offset:              1.107ns (Levels of Logic = 3)
  Source:            en (PAD)
  Destination:       ENC/t1/odd_1_or_even_0 (FF)
  Destination Clock: clk falling

  Data Path: en to ENC/t1/odd_1_or_even_0
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             12   0.000   0.479  en_IBUF (en_IBUF)
     LUT3:I0->O             1   0.043   0.542  ENC/t1/_n0058_inv1_SW1 (N3)
     LUT6:I1->O             1   0.043   0.000  ENC/t1/odd_1_or_even_0_rstpot
     FDC_1:D                   -0.001          ENC/t1/odd_1_or_even_0
    ----------------------------------------
    Total                      1.107ns (0.086ns logic, 1.021ns route)
                                       (7.8% logic, 92.2% route)
```

FIGURE 8. Timing summary.

TABLE 2. Performance comparison analysis.

Method	LUTs	LUTFFs	Slice Registers	Path delays (ns)	Power Consumption (uw)
Performance Comparison of NoC Approaches					
SRAM [6]	121	137	150	3.588	3.622
CAM [8]	120	110	142	3.210	3.614
Buffer [12]	113	93	135	3.122	2.135
Proposed FIFO Buffer	90	89	95	2.893	1.135
Performance Comparison of Crossbar Switching Modules					
Bus module [14]	189	187	177	2.995	2.672
Interconnect [16]	144	171	140	2.582	1.860
HybridInterconnects [20] [19]	127	166	125	1.788	1.824
Proposed Crossbar Switching	93	114	75	1.352	1.574
Performance Comparison of Route Controllers					
Static Route [25]	168	188	106	4.679	4.233
Dynamic Route [30]	155	140	84	4.091	2.611
Stationary Route [27]	96	111	60	3.042	2.318
Proposed Router Controllers	89	66	46	2.716	1.124
Performance Comparison of Proposed Arbiter Controller with Conventional Approaches					
Round Robin [26]	111	112	117	4.783	4.952
XY-Arbiter [30]	100	100	108	4.102	4.033
ISLIP arbiter [31]	99	98	104	1.960	3.027
Proposed Hybrid Arbiters	98	92	97	1.280	1.549

TABLE 3. Performance comparison of adders.

Method	LUTs	Path delays (ns)	Power Consumption (μw)
RCA [23]	20	4.769	4.800
CSA [28]	19	2.615	4.797
CBYA [19]	15	2.029	4.393
Proposed CSLA	8	1.346	1.535

The performance comparison of RC-NoC methods in the Orion framework reveals notable insights. The proposed RC-NoC demonstrates a latency of 9 ms, throughput of 85 Mbps, power consumption of 50 W, and area utilization of 62%. Comparatively, GALS-NoC exhibits a latency of 10 ms, throughput of 100 Mbps, power consumption of 50 W,

TABLE 4. Performance comparison of RC-NoC methods in orion framework.

Method	Latency (ms)	Throughput (Mbps)	Power Consumption (W)	Area Utilization (%)
GALS-NoC [16]	10	100	50	70
F-NoC [18]	12	95	55	75
CE-NoC [28]	15	90	52	72
Proposed RC-NoC	9	85	50	62

and area utilization of 70%. F-NoC achieves a latency of 12 ms, throughput of 95 Mbps, power consumption of 55 W, and area utilization of 75%, while CE-NoC shows a latency of 15 ms, throughput of 90 Mbps, power consumption of 52 W, and area utilization of 72%. The proposed method in Orion performs competitively across various metrics, showing improvements in latency and area utilization compared to existing methods, and achieving comparable throughput and power consumption. These findings suggest that the proposed RC-NoC in Orion offers a balanced trade-off between performance metrics, making it a promising choice for efficient network-on-chip design.

VIII. CONCLUSION

The main objective of this study revolves around the construction of the RC-NoC architecture, integrating FIFO-Buffer, crossbar switching, route control, and arbiter modules. Initially, data generated by diverse devices is stored in FIFO-Buffer logic, which then allocates data based on the devices' IP addresses. Subsequently, the route controller module assumes command over various routers in crossbar switching, implementing priority-based scheduling to regulate these routers. The data is then transmitted from the source to the destination via the arbiter, determined by request levels. Comparative simulations against contemporary NoC designs showcase that the proposed RC-NoC architecture exhibits superior performance concerning area, latency, and power consumption. Moreover, the proposed architecture demonstrates an 11.3% enhancement in data rates compared to conventional NoC approaches. This study lays the groundwork for further extension with optimization-driven NoC architectures, aiming for enhanced performance. Moreover, the optimization process significantly reduced routing costs for various packets: Packet1's routing cost decreased by 40% in Router2, while Packet2's cost was lowered by 44% in Router3. Additionally, the achieved router capacities showcased a 17% increase on average, improving network efficiency notably.

REFERENCES

[1] P. Majumder, S. R. Das, K. Sinha, and B. P. Sinha, "Sustainable operations in IoT by combining spatiotemporal data correlation with silent symbol based communication strategy," *IEEE Trans. Sustain. Comput.*, vol. 8, no. 1, pp. 133–152, Jan. 2023.

[2] F. Mehmood, N. K. Baloch, F. Hussain, W. Amin, M. S. Hossain, Y. B. Zikria, and H. Yu, "An efficient and cost effective application mapping for network-on-chip using Andean condor algorithm," *J. Netw. Comput. Appl.*, vol. 200, Apr. 2022, Art. no. 103319.

[3] R. Parepalli and M. K. Naik, "Design alternatives of network- on-chip (NoC) router microarchitecture for future communication system," in *Proc. Int. Conf. Adv. Comput., Commun. Appl. Informat. (ACCAI)*, Jan. 2022, pp. 1–7.

[4] K. Tatas and C. Chrysostomou, "Hardware implementation of dynamic fuzzy logic based routing in network-on-chip," *Microprocessors Microsyst.*, vol. 52, pp. 80–88, Jul. 2017.

[5] A. Hassan, H. Mostafa, and H. A. H. Fahmy, "NoC-DPR: A new simulation tool exploiting the dynamic partial reconfiguration (DPR) on Network-on-Chip (NoC) based FPGA," *Integration*, vol. 63, pp. 204–212, Sep. 2018.

[6] A. Sharma, M. S. Gaur, L. Bhargava, V. Laxmi, and M. Gupta, "Pre-silicon NBTI delay-aware modeling of network-on-chip router microarchitecture," *Microprocessors Microsyst.*, vol. 91, Jun. 2022, Art. no. 104526.

[7] C. Xu, Y. Liu, and Y. Yang, "SRNoC: An ultra-fast configurable FPGA-based NoC simulator using switch–router architecture," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 2798–2811, Oct. 2020.

[8] F. Yazdanpanah and R. Afsharmazayejani, "A systematic analysis of power saving techniques for wireless network-on-chip architectures," *J. Syst. Archit.*, vol. 126, May 2022, Art. no. 102485.

[9] N. B. Jadhav and B. S. Chaudhari, "Efficient non-blocking optical router for 3D optical network-on-chip," *Optik*, vol. 266, Sep. 2022, Art. no. 169563.

[10] A. K. Biswas, "Using pattern of on-off routers and links and router delays to protect network-on-chip intellectual property," *ACM Trans. Comput. Syst.*, vol. 40, no. 4, pp. 1–19, 2022.

[11] A. A. K. Waleed, A. A. Al-Hilali, and L. F. Jumma, "Design and implementation 4×4 network on chip (NoC) using FPGA," *Periodicals Eng. Natural Sci.*, vol. 10, no. 3, pp. 341–349, 2022.

[12] J. G. Seetharaman and D. Pati, "Design and area performance energy consumption comparison of secured network-on-chip with PTP and bus interconnections," *J. Inst. Eng. (India): B*, vol. 103, no. 5, pp. 1479–1491, 2022.

[13] Y. Xia, S. Yang, J. Niu, X. Fu, and L. Yang, "Strict non-blocking four-port optical router for mesh photonic network-on-chip," *J. Semiconductors*, vol. 43, no. 9, Sep. 2022, Art. no. 092301.

[14] R. Velangi and S. S. Kerur, "Hardware implementation and comparison of OE routing algorithm with extended XY routing algorithm for 2D mesh on network on chip," in *Proc. Int. Conf. Micro-Electron. Telecommun. Eng.*, Singapore, 2021, pp. 159–171.

[15] S. Kashi, A. Patooghy, D. Rahmati, and M. Fazeli, "A multi-application approach for synthesizing custom network-on-chips," *J. Supercomput.*, vol. 78, no. 13, pp. 15358–15380, Sep. 2022.

[16] F. L. Mary, S. B. Sangeetha, and K. K. Prasad, "Optimised meta-heuristic queuing model in VLSI physical design," *ICTACT J. Microelectron.*, vol. 8, no. 1, pp. 1313–1317, Apr. 2022.

[17] W. Amin, F. Hussain, and S. Anjum, "IHPSA: An improved bio-inspired hybrid optimization algorithm for task mapping in network on chip," *Microprocessors Microsyst.*, vol. 90, Apr. 2022, Art. no. 104493.

[18] W. Fan, J. Fan, Y. Zhang, Z. Han, and G. Chen, "Communication and performance evaluation of 3-ary n-cubes onto network-on-chips," *Sci. China Inf. Sci.*, vol. 65, no. 7, Jul. 2022, Art. no. 179101.

[19] R. Gupta, V. J. Kulkarni, J. Jose, and S. Nandi, "Securing on-chip interconnect against delay trojan using dynamic adaptive caging," in *Proc. Great Lakes Symp. VLSI*, 2022, pp. 411–416.

[20] S. Kanithan, N. A. Vignesh, E. Karthikeyan, and N. Kumareshan, "An intelligent energy efficient cooperative MIMO-AF multi-hop and relay based communications for unmanned aerial vehicular networks," *Comput. Commun.*, vol. 154, pp. 254–261, Mar. 2020.

[21] F. Imani, S. Abadal, and P. D. Hougne, "Metasurface-programmable wireless network-on-chip," *Adv. Sci.*, vol. 9, no. 26, 2022, Art. no. 2201458.

[22] N. Arun Vignesh and P. Poongodi, "Analysis of localized quality of service improvement architecture for wireless LAN," *Wireless Pers. Commun.*, vol. 90, no. 2, pp. 701–711, Sep. 2016.
[23] N. A. Vignesh and P. Poongodi, "A cluster-based network architecture scheme for QoS improvement in WLAN," *Int. J. Netw. Virtual Organisations*, vol. 17, no. 3, pp. 158–169, 2017.
[24] P. Bhamidipati and A. Karanth, "HREN: A hybrid reliable and energy-efficient network-on-chip architecture," *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 2, pp. 537–548, Apr. 2022.
[25] R. G. Kunthara, R. K. James, S. Z. Sleeba, and J. Jose, "DAReS: Deflection aware rerouting between subnetworks in bufferless on-chip networks," in *Proc. Great Lakes Symp. VLSI*, 2022, pp. 211–216.
[26] A. Firuzan, M. Modarressi, M. Reshadi, and A. Khademzadeh, "Reconfigurable network-on-chip based convolutional neural network accelerator," *J. Syst. Archit.*, vol. 129, Aug. 2022, Art. no. 102567.
[27] T. Patil, A. Sandi, D. M. D. Raj, S. Chandragandhi, and D. M. Teressa, "A minimal buffer router with level encoded dual rail-based design of network-on-chip architecture," *Wireless Commun. Mobile Comput.*, vol. 2022, no. 1, 2022, Art. no. 6180153.
[28] K. Khan, S. Pasricha, and R. G. Kim, "RACE: A reinforcement learning framework for improved adaptive control of NoC channel buffers," in *Proc. Great Lakes Symp. VLSI*, 2022, pp. 205–210.
[29] S. Ponnan and T. A. Kumar, "On chip network with increased performance for efficient wireless communication," *Meas., Sensors*, vol. 27, Jun. 2023, Art. no. 100804.
[30] N. Habibi, M. R. Salehnamadi, and A. Khademzadeh, "A novel 3D mesh-based NoC architecture for performance improvement," *Majlesi J. Elect. Eng.*, vol. 16, no. 2, pp. 85–101, 2022.
[31] A. V. Bhaskar, "A detailed power analysis of Network-on-Chip," in *Proc. IEEE Delhi Sect. Conf. (DELCON)*, Feb. 2022, pp. 1–7.
[32] A. V. Bhaskar, "Estimation of power consumption in a network-on-chip router," in *Proc. IEEE Delhi Sect. Conf. (DELCON)*, Feb. 2022, pp. 1–7.
[33] S. Singh, J. V. R. Ravindra, and B. R. Naik, "Design and implementation of network-on-chip router using multi-priority based iterative round-robin matching with slip," *Trans. Emerg. Telecommun. Technol.*, vol. 34, no. 4, p. e4514, 2022.

P. ANURADHA received the B.Tech. degree in ECE from the National Institute of Technology, Warangal, in 2003, and the M.Tech. and Ph.D. degrees in embedded systems (ES) from Jawaharlal Nehru Technological University, Hyderabad, in 2010 and 2019, respectively. She is currently an Associate Professor in electronics and communication engineering with the Chaitanya Bharathi Institute of Technology, Hyderabad. She has more than 15 years of experience in teaching. She has 40 research publications in reputed international/national journals and conferences. Her research interests include VLSI, embedded systems, and signal processing. She is a Life Member of IETE.

PRATHAM MAJUMDER (Member, IEEE) received the Ph.D. degree from the University of Calcutta, India. His Ph.D. work were carried out with the Advanced Computing and Microelectronics Unit, Indian Statistical Institute, India. His research interests include energy efficient wireless networks and wireless security. Currently, he is an Assistant Professor with the Department of Information Science and Engineering, JAIN (Deemed-to-be-University), Kanakapura, Karnataka, India.

KANITHAN SIVARAMAN received the B.E. degree in electronics and communication engineering from Anna University, Chennai, in 2009, the M.E. degree in applied electronics from Anna University, Coimbatore, in 2011, and the Ph.D. degree in wireless communication for signal processing from Anna University, Chennai, in 2020. He is currently an Assistant Professor with the Department of Computer Science Engineering (AIML), JAIN University (Deemed-to-be-University), Bengaluru. His research thesis was on swarm intelligent distortion less energy efficient techniques for cooperative MIMO-AF systems. He has published his research papers in refereed international journals and international and national conferences. He has authored the book *Wireless Communications* (Charulatha, ISBN-13:978-81-933409-6-7) and *Visual Object Segmentation Improvement Using Deep Conventional Neural Networks* (Book Chapter-4) (Springer Nature). His research interests include recent technologies in wireless communications and adaptive signal processing which includes blind synchronization and parameter estimation, modulation classification, cooperative communications, physical layer security, cognitive radio, OFDM, SC-FDMA, MIMO, MIMO-OFDM NOMA, mm Wave, IRS, and sequence design.

N. ARUN VIGNESH received the B.E. degree in electronics and communication engineering from Anna University, Chennai, in 2009, the M.E. degree in applied electronics from Anna University, Coimbatore, in 2011, and the Ph.D. degree from Anna University, Chennai, in 2016. He is currently an Associate Professor with the Department of Electronics and Communication Engineering, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad. His Ph.D. dissertation is focused on wireless communications and networking.

S. ARUN JAYAKAR is currently an Associate Professor with the Department of Electronics and Instrumentation Engineering, Bannari Amman Institute of Technology, Sathyamangalam, Erode. He has published research articles in reputed international journals. His research interests include applied mathematics, basic electronics, measurement systems, and control engineering and process control applications.

ANTHONIRAJ SELVARAJ received the bachelor's degree in computer science engineering from Anna University, Chennai, the master's degree in computer science engineering from Vinayaka Missions University, Salem, and the Ph.D. degree in server virtualization from Manonmaniam Sundarnar University, Tirunelveli, Tamil Nadu. He was a Professor in computer science engineering with JAIN (Deemed-to-be-University), Karnataka, Bengaluru. Skilled in developing projects and carrying out research in the areas of cloud computing and data science with programming skills in Java, Python, R, and C. He has co-authored books titled *Mobile App Development*. He has published nearly 25 research papers in national/international conferences and journals. He has published seven patents. His research interests include virtualization, data science, machine-learning java programming, internet programming, computer networks, open-source tools, and components.

SAURAV MALLIK (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, Jadavpur University, Kolkata, India, in 2017. His Ph.D. works were conducted with the Machine Intelligence Unit, Indian Statistical Institute, Kolkata. He is currently a Postdoctoral Fellow of Environmental Epigenetics with the Harvard T. H. Chan School of Public Health, Boston, USA. Previously, he was a Postdoctoral Fellow with the School of Biomedical Informatics, University of Texas Health Science Center at Houston, Houston, TX, USA; and the Division of Bio-Statistics, Department of Public Health Sciences, University of Miami Miller School of MediC_{in}e, Miami, FL, USA. He has co-authored more than 75 research articles with a Google H-index of 17. His research interests include computational biology, bioinformatics, data mining, bio-statistics, and pattern recognition. He was a Research Associate of the Council of Scientific and Industrial Research, MHRD, Government of India, in 2017. He was also a recipient of the Emerging Researcher in Bioinformatics Award from the Bioclues BIRD Award Steering Committee, India, in 2020. He is an editor of many journals.

AMAL AL-RASHEED received the Ph.D. degree in information systems from King Saud University, in 2017. She is currently an Associate Professor with the Department of Information Systems, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University (PNU), Riyadh, Saudi Arabia. She has been involved in many projects related to learning technologies, cyber security, and virtual reality. Her contributions to research projects in academia led to the publication of papers in many journals and conferences. Her research interests include education, knowledge management, data mining, data analytics, cyber security, and natural language processing. In 2017, she was awarded the Research Excellence Award, by PNU, for her publications during performing the Ph.D. degree.

MOHAMED ABBAS received the B.Sc. degree in electronics engineering from the College of Engineering, Mansoura University, Egypt, in 1998, and the M.Sc. and Ph.D. degrees in computer engineering from Mansoura University, in 2002 and 2008, respectively. He was an Assistant Professor with the Department of Communications and Computer Engineering, College of Engineering, Delta University. He is currently a Full Professor with the Department of Electrical Engineering, King Khaled University, Abha, Saudi Arabia. His research interests include intelligent systems, medical informatics, nanotechnology, and bioinformatics.

BEN OTHMAN SOUFIENE received the M.S. degree from the University of Monastir, in 2012, and the Ph.D. degree in computer science from Manouba University, in 2016, with a focus on secure data aggregation in wireless sensor networks. From 2016 to 2024, he was an Assistant Professor in computer science with the University of Gabes, Tunisia. His research interests include the Internet of Medical Things, wireless body sensor networks, wireless networks, artificial intelligence, machine learning, and big data.