Efficient Packet Switching: Designing and Implementation of Router for High-performance and Low Latency

Thesis report submitted in partial fulfilment of the requirement for the Award of the Degree of

MASTER OF TECHNOLOGY

in VLSI DESIGN

Submitted By

YESHA NIGAM

6024620030

Under Supervision of

Dr. MOHIT AGARWAL

(Associate Professor)

&

Dr. MAYANK AGARWAL

(Associate Professor)

&

Mr. AMIT SINGH

(Mentor at STMicroelectronics)

ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY

(A DEEMED TO BE UNIVERSITY), PATIALA, PUNJAB

JANUARY – JUNE 2026

DECLARATION

I, YESHA NIGAM hereby declare that the work presented in this report entitled **"Efficient Packet Switching: Designing and Implementation of Router for High-performance and Low Latency"** in partial fulfilment of the requirement for the award of degree of Master of Technology (VLSI Design) submitted at **ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT** ,Thapar Institute of Engineering & Technology (Deemed to be University), Patiala is an authentic record of work carried out under supervision of **Dr. MOHIT AGARWAL** (Associate Professor) & **Dr. MAYANK AGARWAL** (Associate Professor) & **MR. AMIT SINGH** (Group Manager) from August, 2025 to July, 2026. The matter presented in this has not been submitted either in part or full to any other university or institute for the award of any other degree.

Yesha Nigam

Yesha Nigam

Date:07/07/202

Amit Singh Mr. Amit Singh (Technical Lead) Date:29/06/2026	Dr. Mohit Agarwal Associate Professor Department of Electronics and Communication Engineering, Thapar Institute of Engineering & Technology, Patiala Date:29/06/2026
	Dr. Mayank Agarwal Associate Professor Department of Electronics and Communication Engineering, Thapar Institute of Engineering & Technology, Patiala Date:29/06/2026

STMicroelectronics Private Ltd.
Plot No.1, Knowledge Park-III,
Greater Noida – 201 308.
Uttar Pradesh, India
Tel : +91-120-4003000
Fax :+91-120-4003007
Email : infostm.india@st.com
CIN : U32109DL1990FTC039906
www.st.com

July 24, 2026

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Yesha NIGAM** has undergone internship in our **TRD** Group from **August 07, 2025 to July 24, 2026** and has successfully completed the project on: **Verification of MBIST Design using UVM.**

During the internship period, Yesha was found to be sincere and professional in conduct.

We wish her all the best for the future endeavors.

for **STMicroelectronics Pvt. Ltd.**

Sanjay Kumar PIPLANI
India Talent Acquisition Head

ACKNOWLEDGEMENT

This thesis would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, my sincere gratitude to **Dr. Mohit Agarwal** & **Dr. Mayank Agarwal** for his continuous support, motivation, and immense knowledge. Their guidance has helped me with all the time of research and writing of the thesis.

Mr. Amit Singh, Group Manager of Team DMIPS, STMicroelectronics Pvt Ltd, for accepting me as an intern in his team. I am grateful to him for his expertise, kind concern, and consideration regarding my academic requirements.

Mr. Gaurav Chugh, Project Lead, STMicroelectronics Pvt Ltd for her patience and steadfast encouragement, her constructive criticisms at various stages of my work were thought provoking and she helped me focus on my ideas. I am deeply grateful to her for all the discussions that helped me in understanding the technical details of my work.

Finally, my family and dear friends. None of this would have been possible without their love. And the one above all of us, the omnipresent God for giving me the strength to plod on, thank you so much everyone.

ABSTRACT

The design and implementation of a 1X3 router, a crucial part of contemporary network communication systems, are presented in this conference paper. FIFO (First-In- First-Out) buffers, a synchronizer, a controller, a Finite State Machine (FSM), registers, and packet switching are all crucial components of the router architecture. This paper provides a thorough explanation of the design, operation, and overall function of each component in establishing effective packet routing and enhancing network performance. System on chip (SoC) technology enables the placement of millions of transistors on a single device. The processing performance of traditional SoCs is constrained by their bus-based communication design. Licon industry segments are increasingly embracing the concept of Network on Chip (NoC), which separates computation from communication. The router is the foundation of the network; thus, it must be designed well to improve the system's performance. In the current study, we concentrate on designing a router input-output protocol. For close loop communication, the proposed system features a virtual cut through mechanism. Router 1x3 has three output ports and one input port. Top-level architecture created in the Verilog HDL language with sub-modules including FSM, Synchronizer, FIFO, and Register. Simulation and synthesis is done on Model Sim and Quartus Prime and Code coverage is found on Questa Sim. This paper presents the design and implementation of an efficient router architecture optimized for low-latency and high-performance packet switching. The modular switching fabric, optimized buffering strategy and efficient arbitration logic in the proposed design allow concurrent packet forwarding while minimizing contention and processing delays. Proposed router architecture supports scalable data transfer and reliable operation under heavy traffic conditions. The design is validated and evaluated through simulation and synthesis allowing improvements in packet handling efficiency, reduced latency and efficient utilization of hardware resources. The evaluation proves that the proposed router could be adequately used in modern day efficient communication systems with desire for fast reliable data transfer.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

Chapter 1 : INTRODUCTION

1.1 Overview

The development of communication systems, embedded platforms and High Performance computing (HPC) applications has resulted to increasing demand for efficient data transfer mechanisms. Modern day's digital systems are in the need of reliable high speed communication networks to transfer massive amounts of data with less latency and high throughput [1]. Routers play a vital role in these types of communication networks in routing the data packets from a source node to their respective destination. Due to the increasing complexity of the networks, efficient design of routing architecture has become one of the challenging research areas among VLSI, NOC (Network-on-Chip) and Digital communication systems. A router is an intelligent switching device that receives an input packet, analyses its destination and routes through one of its output channels. The performance of the communication networks depends heavily upon the efficiency of their routers' architecture, due to which the communication networks demand for efficient router architectures that could possibly deliver high speed with low power consumption, reduced hardware complexity and reliable routing of data packets [2]. In this work the router architecture of is designed and carried out, which receives a data packet through a single input port and routes to one of three output ports about its destination address present in the packet header. The design consists of important modules like registers, finite state machines, FIFO (First in First Out) memories, synchronizers and various other control logic.

1.2 Background of Router Technology

A router is the most common and hence most overlooked device in computer networks. From humble beginnings, today the Internet router links the computers used by billions across the world and serves as a backbone to help data exchange at breakneck speeds. However, the importance of routers must be understood - how they work, which architectures most common these days, their vulnerabilities but also how they may be made more effective to handle the needs of increasingly sophisticated networking. Routers in networking primarily route data across relatively tiny networks that did not demand hefty bandwidths typically. Emerging applications such as cloud computing,

artificial intelligence, Internet of Things (IoT), autonomous systems, multimedia streaming and high-speed data centers have generated an explosive demand for communication. These applications require a router that is capable of processing copious amounts of data with minimum possible delay and providing maximum reliability [3]. Besides networking, routers have also gained prominence in VLSI systems thanks to the development of Network-on-Chip architectures. As integrated circuits grow more complex, bus-based communication falls short of providing sufficient scalability, congestion control and power efficiency required for the on-chip communication. The NoC based communication uses routers for efficient data exchange between multiple processing elements, memories and other peripherals on a chip. Optimizing of the architecture of such routers has hence become important to attain better systemperformance.

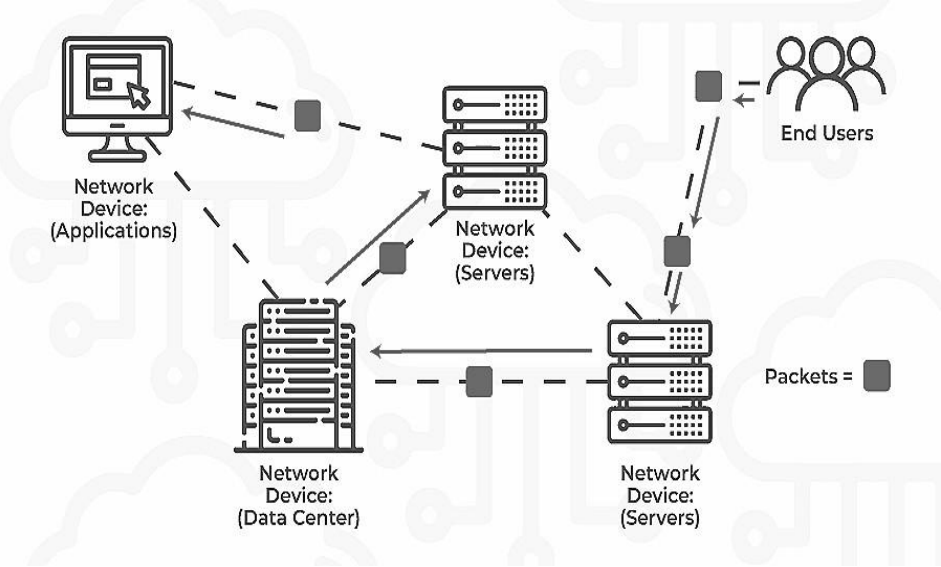

Figure 1 Packet flow in a network system

The efficient design of routers is becoming evermore critical as our consumption for network communication escalates. This paper describes the design of a 1x3 router that includes various components such as FIFO buffers, a synchronizer, controller, FIFO FSM, registers and Packet switching to efficiently route the packets and subsequently increase the network throughput. Nowadays in our interconnected world, efficient data routes play a pivotal role to provide seamless communication throughout the networks. Traditional designs of the routers are incapable of meeting the demands of the modern applications seeking high bandwidth and low latency. This paper describes a novel technique to fulfill all the demands, using the concept of designing a 1x3 router that has been optimized with efficient switching techniques

Several key features characterize the Proposed Router (Principle is as follows). Firstly, it adopts a distributed architecture where incoming and outgoing packets are handled by multiple processing units simultaneously. This parallel processing helps to ease the load at the routing core. Secondly, the Proposed Router includes intelligent routing algorithms that allow it to dynamically adapt to network conditions for optimal path selection and load balancing. Finally, the Proposed Router employs advanced switching methods such as packet switching and circuit switching to improve forwarding efficiency [6]. Continuous technological advances in the world of communications have revolutionized the processes of how data is generated, transmitted, and processed on all interconnected systems. Nowadays, digital networks are increasingly expected to support mammoth traffic loads whilst simultaneously meeting ever-more strenuous user-centered requirements such as speed, reliability, and responsiveness.

The efficiency of devices within the networks is thus becoming crucial in determining the entire system performance. As the heart of networks, routers play the pivotal role of directing data packets between network endpoints in an organized and on-schedule fashion. Because of the prevalence of bandwidth-hungry and latency-sensitive applications, routers are increasingly expected to operate under tough conditions. Cloud computing, multimedia content streaming, autonomous driving systems, and real-time analytics all require endless data delivery with as little delay as allowable. The typical designs for these traffic loads have often struggled in sustaining performance under modern-day traffic loads. Fixed routing, inefficient buffering schemes, centralized processors, etc., may all be causes of bottlenecks, packet delay [5], and reduced throughput. Instead, routers are being designed aiming at architectural efficiency, parallel processing, and intelligent packet management; hardware-based routers have the advantage of exhibiting predictable timing behavior and less overhead for processing over software ones.

1.3 Router Architecture

Several modules are designed as components comprising the router architecture. Each unit, having its particular function, work interactively as the packet transmission process. As explained, each module contributes to the overall efficiency and reliability of the process . The NoC based communication uses routers for efficient data exchange between multiple processing elements, memories and other peripherals on a chip.

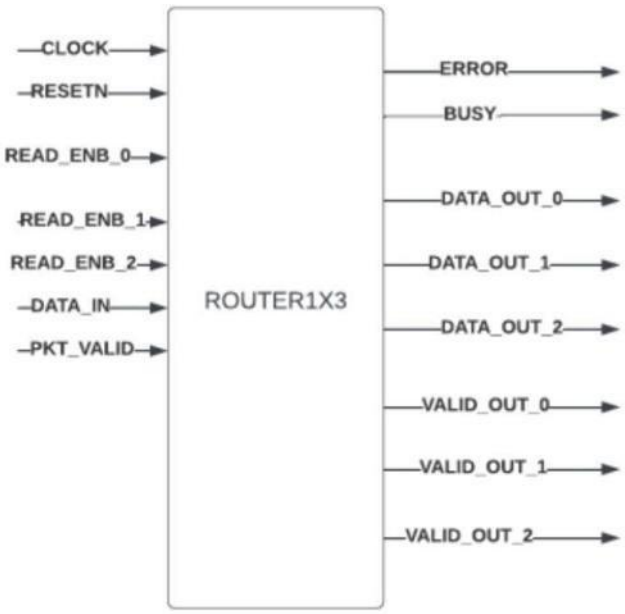

Figure 2 Top module of Router

1.3.1 Register Module

Incoming packet information must thus be temporarily stored in the register. Header, payload and parity bits used to confirm a packet are saved. Registers serve the purpose of maintaining a stable transportation of data between levels and at a stage when processing, routing information of a packet could cause data loss.

1.3.2 FIFO Buffers

To store the packets that are not yet sent on to their output channel FIFO memories are used. Every outputting channel has FIFO buffers that can accept packets and sent packets out at the same time. FIFO buffers are the secret weapon to fight short-time network congestion. Packets wait until the outputting channels become relatively less busy to be sent on to their destination [4]. Router reliability and throughput are dramatically increased through this.

1.3.3 Synchronizer

The synchronizer is responsible for connecting the controller and FIFOs. Essentially, it checks signals concerning how full each buffer is, ensuring different parts do not perform an action (e.g., reading/writing data) simultaneously or too rapidly, preventing corruption of data. In proposed router a new control device is introduced called as synchronizer. Synchronization of all the data and the other functions. In synchronous clock also hardware devices can be out of alignment due to time delay differences of data etc.

When hardware has to be implemented (buffering structures + the control logic or the synchronization + also the switching techniques), a careful consideration must be given (this applies to each element in discussion), such that they can work together in congruence, causing neither unnecessary complexity, nor any additional delay. The design and implementation part of the paper covers an architecture of router to better route the packets while attempting to have a smaller modular construction is discussed [8]. Input FIFO buffers, synchronization, the centralised controller, FIFO control finite state machine (FSM), registers and the switching logics are present in the router to provide all its functionalities. All of these together help route a packet from destination till arrival. It would manage in varied condition. Packet buffering: Buffering mechanism of any kind is an important consideration for router performance.

The in-depth study required in doing the job properly, has led to inclusion of FIFO buffers, to hold any arriving packet. Buffers serve several purposes in this project from handling any bursts that could arise due to uneven arrival rates, while also keeping the order of the packets. The FIFO control takes care of the process or the en-queue / de-queue of the packets accordingly, while avoiding occurrence of stack-overflow / underflow. Routing intelligence can have a large impact in network efficiency: Instead of relying heavily on static routing decisions that could eventually become obsolete due to shifting demands in the network [4]. Routing logic is embedded within the router whose works include examining the header of every approaching-packet and consequently allot it to an outward destined port which would then direct the packet to whatever its next address is. Router architecture has the advantage with its ability to adapt better by harnessing parallelism within it to achieve quick response while enhancing the overall routing time simultaneously.

The division of job from every functional node or block, can allow one to address many packets at given period of time instead of one-by-one addressing. Division of work would cause router's heart to be out from bottle-neck and help router stay connected even during the heaviest traffic conditions. Switching technique forms heart of router design in forward mechanism. As packet are transferred independently according to the routing decisions to reach out as swiftly, one with router uses maximum output or port effectively by allowing more packets to reach their destination than wasting in routing procedure and hence adjusting in dynamic traffic patterns. In situation where, to improve performance to some extended where it is predictable (like transmission of signal), circuit like manner to

deal can also be opted. Switching could hence be done according balance in efficiency and adaptability of this mode.Synchronization between control and data paths is essential for correct operation in hardware implementations. Timing mismatches can result in incorrect packet handling or data corruption. The inclusion of a synchronizer module ensures that signals exchanged between different functional blocks are properly aligned, enabling stable and deterministic operation across the router.

Making sure the control and data signals line up is really important for hardware to work right; otherwise, you will get messed-up packets or corrupted data. By adding a synchronizer, we can make sure signals passing between different parts of the hardware are properly timed, leading to steady and predictable behavior throughout the router.Even though the router we are proposing uses the configuration, its basic design is built to scale. Because the design is modular, adding more input or output ports will not require much change. This means the architecture is good for stand-alone routing jobs and can also serve as a building block for bigger network-on-chip or embedded networking setups.

This paper presents a router design that is both organized and efficient, tackling major hurdles in how we handle packets today. It manages buffering, routing decisions, synchronization and switching cleverly, showing how the router can be both high-performing and straightforward.This design functions as both a real-world setup and a general model for understanding how routers can be built efficiently [2].

The top module, sitting at the highest level, is where everything in the router design comes together. It lays out how the router interacts with the outside world and orchestrates everything internally to ensure packets are routed correctly and efficiently. This top module's job is to guide the data from input to the right output, while also keeping things in sync, under control and processed in the right order.

Chapter 2 LITERATURE REVIEW

Kattepur and Mohalik [1] Has introduced the RoB-Router, a reorder-buffer-enabled low-latency NoC router aiming at delay reduction while maintaining correct sequencing of packets. The structure of the buffer will affect router speed and efficiency, relevant to the use of a FIFO store and controlled releasing of packets from such store.

Li et al. [2] Has introduced the RoB-Router, a reorder-buffer-enabled low-latency NoC router aiming at delay reduction while maintaining correct sequencing of packets. The structure of the buffer will affect router speed and efficiency, relevant to the use of a FIFO store and controlled releasing of packets from such store.

Katta et al. [3] This paper presented SB Router, a swapped buffer activated low latency router. The work done here lays stress on the synchronization of presentation of packets and presentation buffers of packets while processing them. This supports the concept of including the synchronizer module in the router discussed above.

Swapna et al. [4] Studied a five-port router for NoC applications. This study illustrates the growth in router complexity as the number of ports increases and the consequent need for more control and arbitration logic in larger router designs; supporting the choice of modular approach used in the proposed router architecture.

Gopal N. [5] This work presents the RTL design and verification of Router 1x3. This reference directly supports the thesis by demonstrating that the router can indeed be successfully designed and verified at the RTL level. Such results further validate the proposed architecture.

Kwak et al. [6] Presented a low latency and energy efficient router for 3D NoC, trying to get best of speed and power efficient router which thesis also aims for.

Tanenbaum and Wetherall [7] Congestion control, routing, and packet switching were some other ideas that came across from him at Bell Labs that form the theoretical background for the proposed packet router.

Dally and Towles [8] Provided crucial understanding of key interconnection network notions, which have a strong impact on the architecture of the proposed router (routing, arbitration, flow control and buffering of data).

Duato et al. [9] Discussed designing practical routers in terms of correctness of routing, flow control, and scalability points of view. Their work contributes towards showing that our design is viable, extendable, and modular.

Mano and Ciletti [10 FSMs, sequential logic, combinational circuits and timing behaviour provided the digital design fundamentals required for RTL implementation - these are directly applied in router controller and control logic and are the basics of implementing the proposed packet-switching architecture in verilog.

McKeown [11] Presented the iSLIP scheduling algorithm for input-queued switches (1999). It confirms that in router arbitrations logic is required because multiple input ports with pending cells to the same output simultaneously may occur. This work addresses the problem of effectively and fairly allocating shared resources in packet switches, an issue crucial for efficient system performance.

Pande et al. [12] proposed a scalable and energy-efficient NoC router architecture . Their work reinforces the importance of compact and extendable router design, especially when future scaling and power constraints are considered.

Kumar et al. [13] discussed NoC architecture and design methodology . Their study shows how routers fit into larger on-chip communication systems, and it supports the thesis context by emphasizing that routers are key building blocks in scalable communication fabrics.

IEEE 802.3 [14] defines a reliable communication framework for data exchange in networks It supports the idea that communication hardware must follow disciplined and standardized rules, which is essential for predictable packet transfer.

Benini and De Micheli [15] Micheli introduced Networks on Chips as a new SoC paradigm . Their work motivates the use of packet-switched communication instead of traditional bus-based systems, and it explains why NoC-based router structures are increasingly important in modern chip design.

Jantsch and Tenhunen [16] gave a detailed study of NoC concepts and architectures . Their work supports the broader NoC foundation of the proposed router and gives a strong theoretical background for router-based communication in integrated systems.

Hemani et al. [17] discussed NoC as an architecture for the billion-transistor era . Their work highlights the need for scalable communication infrastructure in modern chips, where traditional buses become inefficient and difficult to expand.

Dally [18] explained virtual-channel flow control. This supports the importance of flow regulation and congestion handling in router design, and it shows how structured data movement improves overall network efficiency.

Glass and Ni [19] introduced the turn model for adaptive routing. Their work shows how routing strategies can improve flexibility while avoiding deadlock, and it gives useful background for routing strategy decisions in NoC systems.

Hollstein et al. [20] proposed formal deadlock-free routing methods in NoC. Their study supports the importance of correctness and safe routing behavior, especially in systems where multiple paths and traffic patterns may create routing hazards.

Garzarán et al. [21] studied buffering techniques in NoC routers. Their work supports the use of FIFO buffering in the proposed router and shows that the choice of buffering structure strongly affects throughput and latency.

Goossens et al. [22] described the Aethereal network-on-chip architecture. Their work demonstrates structured communication and efficient packet handling in practical NoC systems, and it provides a useful example of real router-based interconnect implementation.

Abdelfattah and Eltawil [23] proposed a low-latency router design for energy-efficient NoC systems . Their work aligns with the thesis goal of compact and efficient routing, where delay reduction and resource efficiency must be achieved together.

Sapatnekar and Akella [24] discussed timing and synchronization challenges in high-speed digital systems. This supports the use of synchronization logic in the proposed router, because correct operation depends on meeting timing constraints between sequential blocks.

Marculescu et al. [25] reviewed major NoC research problems. Their study places the proposed router in the larger research context of scalability, reliability, and performance, and it shows that router optimization remains an active research area.

Kumar, Concer, and Carloni [26] studied pattern-aware routing and buffering in NoC. Their work supports the idea that routing and buffering should be coordinated for better performance, especially when traffic patterns are irregular or bursty.

Kim, Balfour, and Dally [27] proposed flattened butterfly topology for on-chip networks . Their work shows how network topology affects latency and scalability, which is useful for understanding how router architecture influences communication performance.

Radetzki et al. [28] reviewed fault tolerance methods in NoC topologies. Their study supports the need for reliable and robust router operation, especially in systems where communication errors or path failures may occur.

Agarwal, Ray, and Ghosh [29] discussed verification methodologies for packet-switched digital IPs using SystemVerilog . Their work supports the need for thorough testing before hardware realization, and it highlights the importance of verification-driven design in modern digital systems.

Existing research on routers and Network-on-Chip (NoC) design primarily tackles large-scale systems, sophisticated routing strategies and enhancing performance metrics. There is less attention paid to a straightforward router, like the , which offers a compact Register-Transfer Level (RTL) implementation ideal for smaller packet-switching scenarios. While numerous papers explore buffering, flow control or reducing latency in isolation, actually combining FIFO buffers, a controller Finite State Machine (FSM), a synchronizer and destination decoding into a single, cohesive design is infrequently discussed. Furthermore, a fair number of studies remain theoretical or conceptual, lacking a complete, practical RTL implementation ready for direct verification and synthesis. This situation clearly indicates a demand for a router design that is modular, operates with minimal latency and is simple to verify, striking a balance between ease of use, effectiveness and dependable packet management.To fill this noted gap in research, this thesis introduces a simple, modular router featuring an RTL-level implementation.

This design incorporates FIFO buffering, a controller FSM, synchronizer logic and destination decoding mechanisms, all aimed at facilitating reliable packet transfer. It achieves low latency and accurate output selection. In contrast to existing literature that typically emphasizes large NoC architectures or decontextualized routing principles, this work presents a concise and functional router design that is more readily verifiable, synthesizable and deployable within small-scale communication systems [1][2].

Chapter 3 SUB LEVEL ARCHITECTURE OF ROUTER

The sub-level structure includes an array of separate functional modules in router design, each dedicated to performing a specific task. These include blocks like the register unit, controller, synchronizer, FIFO buffers and finite state machine (FSM). These modules collectively handle tasks such as processing packet headers, coordinating data transfer and facilitating efficient routing. This setup not only enhances design clarity but also streamlines verification, allowing for a much more efficient router.

3.1 First In First Out

Before transmitting a packet to the appropriate target node, a FIFO buffering element holds it. The distributed form of buffering system we employed in this work features a separate buffer for each output port. Due to the Routerlx3 architecture, three buffering FIFOs with a depth of 16 bytes and a width of 9 bits are used. The read- write signals serve as the basis for the FIFO module's logic. Every time the synchronizer module provides an enable signal for write, dat is loaded into the FIFO via a write pointer. Data is read from the memory address provided through the read pointer when it notices the read enabled signal. FIFO Full and Empty are the two state signals available. When the FIFO is empty, it is ready to receive data, whereas when it is full, there is no room for data to be written. In computer networking, FIFO stands for First-In-First-Out and refers to a queuing discipline used in network routers to handle the order in which packets are processed [1]. As you indicated, a 1x3 router normally includes 3 output ports and a single input port.

In the context of a 1x3 router, FIFO would mean that incoming packets are processed and forwarded in the order they arrive at the input port. The oldest packet in the queue would be handled next bythe router, which would keep a running list of packets.

A FIFO (First-In, First-Out) is a storage block that transfers data in the exact order it is received. The earliest data entered into the buffer is always the first to be read out. In a 1×3 router, FIFO memories temporarily hold packets Every time the synchronizer module provides an enable signal for write before forwarding them to the selected output channel. Although the destination is not ready for the data, the buffer stores it and allows sending to other destinations. Also, it helps not to discard them and not to break their order, so not to corrupt packets [9].

3.2 FIFO depth

That is the total storage capacity of the FIFO in terms of data locations. It represents the number of data words or data bytes that can be stored in the FIFO while the FIFO buffer is not yet full. By determining the depth of FIFO, it will allow the router to overcome short-term fluctuations of network traffic without loss of any data packets. By sizing the FIFO depth properly, a balance is made between the hardware needs, as well as the communication, and it will further ensure an efficient and reliable routing of data packets. FIFO depth refers to how long the buffer would hold before transmit. By increasing the depth of FIFO in a router, it will reduce buffer loss even if the receiving port on the other end has not been activated, and it does have a significant impact on resource usage at design time or cost. Proper adjustment would lead to good management for processing data. Hence, by sizing the FIFO properly.

The transferring of data packets should be more stable and would lead the router to operate at much higher performance. This also helps in controlling how long the data arriving is being buffered until the transmission starts, besides balancing between FIFO length variations for smooth communication flow with the other devices, router traffic [10], and also system throughput at different processing speeds. Additionally, it can also improve its router reliability, allowing a greater network delay for traffic jams. Hence, in a router, FIFO depth can actually help in absorbing the burst traffic, maintaining the packets' orders, and forwarding of continuous data without congestion. This will allow the router to handle mismatched conditions if it is a result of input arrival rate versus the output available, contributing to an improvement in the overall system efficiency besides a robust design for the router.

This will then be sent off to the relative output port by the forwarding logic using its routing table/other means once it is determined, and then data will be transferred in, which is then dequeued from the back in order according to its time. The FIFO is used because it gives the ordering, which usually means better sequence without rearranging of messages [7]. Although it is to be noted that FIFO is just a discipline which is only applicable in ordering, and there is no guarantee on fairness, QoS, or even the prevention of a congested network. For that reason, various different queuing mechanisms or disciplines and algorithms might be used and incorporated in accordance with FIFO so that it can fulfill the desired action and needs within the router itself.

FIFO in the router sub-architecture, a must in many cases, is used to store incoming packets in a temporary storage before it is delivered by means of switching. Moreover, it maintains the sequence of arrival when the packets arrive, thereby allowing orderly delivery and steady data flow through the router. FIFOs are commonly used with input interfaces to counter different traffic rates and to prevent loss of packets during events of congestion by storing and releasing them at an appropriate rate.

The design uses independent read and write pointers and the status control logic to recognize the buffer full and buffer empty states, which permit to carry out the synchronized operations on enqueue and dequeue without invalid memory-access occurrences. By storing the incoming packets into the FIFO, the router decouples the storage process from switching, thus it minimizes the contention in the internal data path and easily arbitrates among the competing inputs [5]. FIFOs are built which have optimized latency access for better throughput, which also has minimum hardware complexity leading to less contention and better performance in this high-speed world.

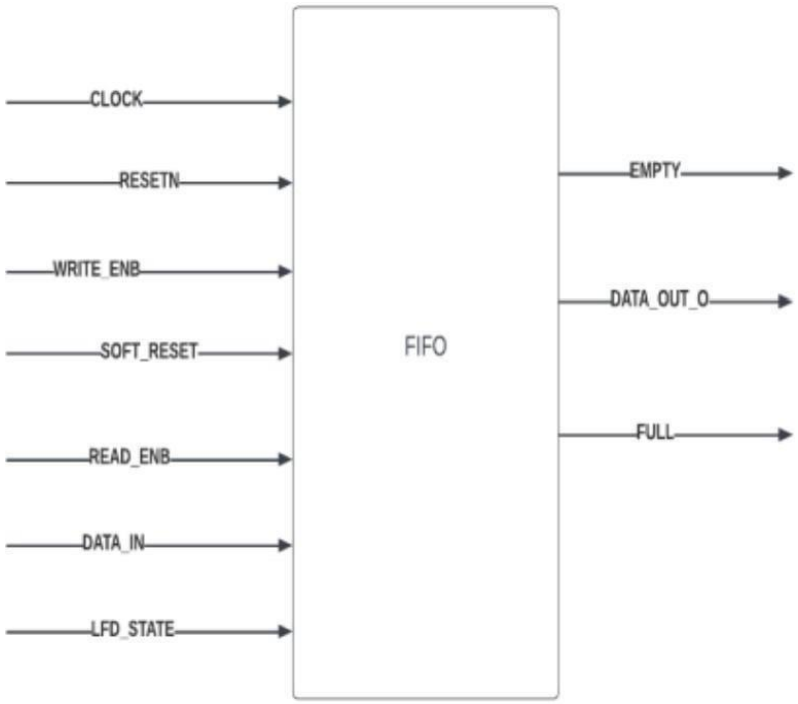

Figure 3 FIFO Block

3.3 Controlling of Router

The router's monitoring and regulating functions are carried out by this signal generator type FSM. It may function may router architecture controller. The FSM in the design is of the Moore kind. Two logical loops have been created specifically for FSM, the first loop Beginning in the idle state. Whenever it receives a signal indicating that a packet is valid, the FSM

asserts the signal Detect address reads the first byte from the packet as the packet header. The signal lfd_state high generated by the next state, load initial data, shows that the router has accepted the header. Next, it generates rates the the signal signal Id_state high to accept the real payload. The next state is parity checking, when the FSM completes the final byte of data received after receivingthe complete aner reserving parity done signal from the register module.

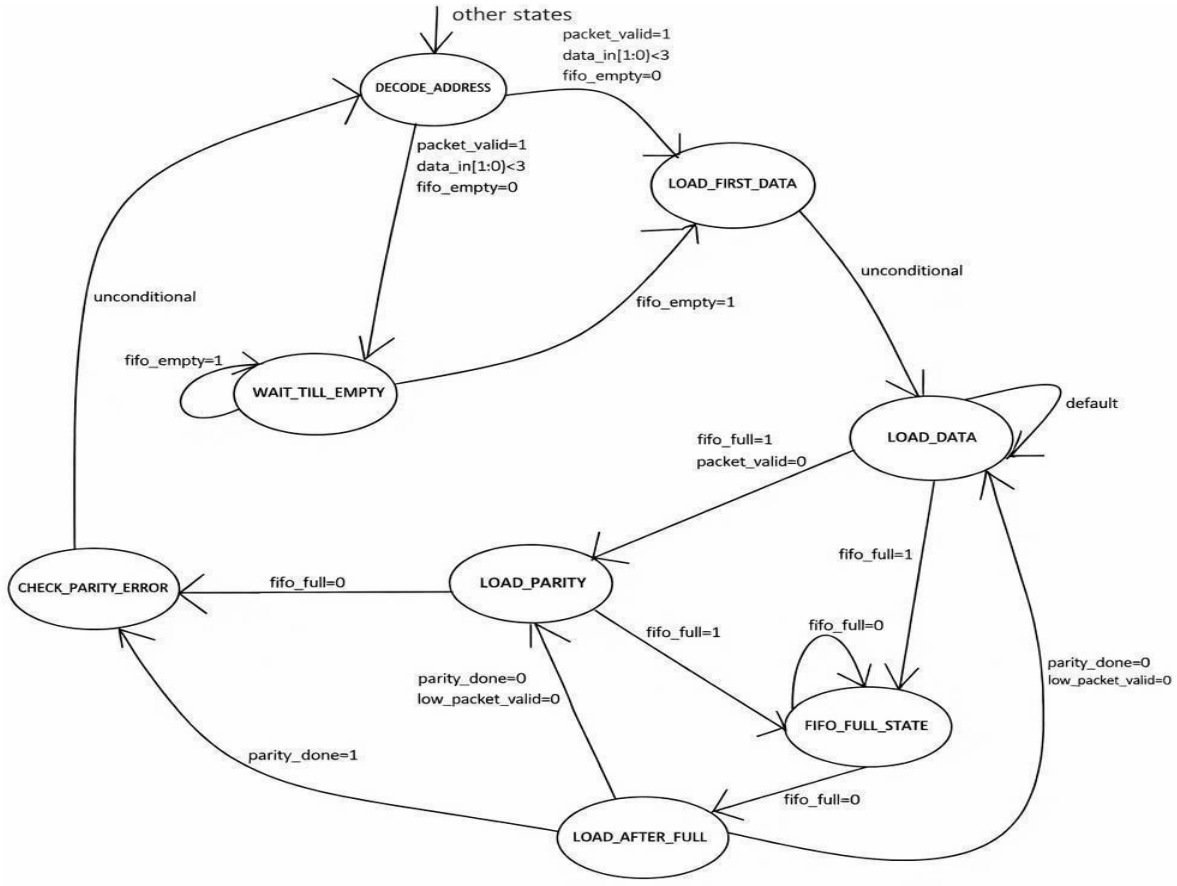

Figure 4 Finite State Machine Diagram

The second loop begins in the idle state asserts Detect address to indicate that the header has been received ter receiving the first byte of the packet. If it detects the fifo full status, it instantly enters the fifo full state and asserts the signal full state. letting the user know that one or more of the FIFOs is full and that they must wait for it to empty. In the following state, it news to accept the real payload as before.

During asserting the signal laf_state, which signifies load after full state, the router there will hold onto the final packet byte till the FIFO is empty before starting to receive new packets. We have created a virtual cut through mechanism that makes use of the FSM controller's "busy"

control signal [12]. The source node's input signal from the router is busy. It informs the source node of the current state of the destination node. After re 2 living a legitimate signal of packets carrying a valid set of data from a source node, the FSM asserts the busy signal.

The control unit is the core supervisory element of a router, responsible for coordinating all internal activities that enable reliable and efficient packet transfer. The controller tells the data-path - buffers and switches how to act, what to do, what sequence, when to act and in which situation action needs to be taken, moving packets through them. Router in hardware requires valid logic controlling its operations. The controller prevents any corruption in the data from happening, deals with possible congestion and in overall ensures deterministic nature of the proposed router.

The control logic acts according to as a centralised coordination that takes care of packets being brought to the router, going through buffering, routing, arbitration and finally being transmitted to output of the modem [13]. All modules inside are related in some way to the controller; they are FIFO buffers on inputs, FIFO control state machine, synchroniser, routing part and output interfaces. It all together tries not to interrupt a smooth stream of packets going from inputs of the modem up to a required port. Admission control is an important part of the controller. Incoming packets will be arriving irregularly and also varying rate.

The controller monitors FIFO's signals about their status such as occupancy and full which tells it if there is enough space in the buffer to successfully input another packet. Controller allows write into the buffers only when there is enough space making buffer overflow impossible ensuring clean data. However, not only does the controller deal with acceptance of packets but also time management of their release seems crucial. Taking packets from the FIFO too early could halt forwarding due to no available space on the output side of the router; could eventually lead to data corruption. The controller allows dequeue only if the routing has been completed successfully and port has some space to accept packets.

This ensures the buffering-forwarding cycle will follow correctly not being halted or having any hiccups giving a stable and predictable stream of packets. FIFO finite state machineRouting supervision is another critical responsibility of the control unit. When a packet reaches the head of the FIFO, the controller initiates routing evaluation by enabling header inspection.

Destination information extracted from the packet determines the selection of one among the three output ports. The controller communicates this decision to the switching logic, which prepares the internal data path for packet transfer. Contention resolution is handled directly by the controller when multiple routing requests target the same output port. Arbitration logic embedded within the control unit determines which packet is granted access during a given cycle. Arbitration strategies may be designed to emphasize fairness, responsiveness, or simplicity, depending on system requirements. By managing contention at the control level, the router avoids starvation and maintains balanced utilization of output resources.

Interaction among these two (controller and the synchronizer module) coordinates the timing across the router. Some timing hazards occur such as variation among the processing latency while buffering, routing and switching etc. To avoid this the controller ensures the control signals arrive at the output destination at the time when a valid data window is available to successfully transmit the data within controller to other controller [10]. It ensures reliable operation as these are built using synchronous hardware. When an output port is selected, the controller handles activation of the output by asserting the enable and valid signals that load the packet to appropriate register.

After transferring is done it clears internal resources such as entries in FIFO or routing grant to avoid partial transfer and deliver a complete packet. It also improves reliability by detecting abnormal operation such as tries to read from an empty location or tries to write into a full buffering capacity.From an efficiency perspective, the controller does not perform superfluous internal processing when modules are not needed [11]; these modules are accessed on an as-and-when required basis only, and therefore this controlled procedure reduces unwanted switching, thereby improving power gain as well as providing more scope and utilization capability of the hardware present, etc.

These properties may help for embedded and on-chip systems where restrictions for power and area exist. Although the present design tries for configuration of a 1×3 (the acronym used earlier) router only, the scope of control extends out further too; with its modular structure, many more numbers of input and output ports can be added with no or almost no modification of the overall picture. Control signals, arbitration, and evaluation processes can be scaled according to the problem, and a more complicated router design can therefore be facilitated by means of the present controller.

The router controller as summarized above is the heart of the router as it allows orderly operation of the router from packet admission, buffer access, routing decisions, arbitration, synchronization to outputs. The control strategy as proposed above for the router makes the proposed router capable of low latency switching as well as high reliability under different traffic conditions. Data is only delivered to the target node when the busy signal drops to a low level. The busy signal rises again in response to the router receiving the last parity byte of the packet.

This section acknowledges closed-loop communication using well defined Verilog control signals hardware implements the router controller where they control data movement between the blocks and control the buffer, router and output transmission. All the control functionality previously described at the architectural level has been converted to explicit signals that interact with The FIFO buffers, router and state machine and output-registers [12]. It shows how the control function is dealt with at RTL level on explicit signals.

3.4 Control Signals

3.4.1 Global Control Signals

At the top level, the controller operates synchronously with the system clock and reset. The clk signal acts as the global timing reference for all control and data-path operations, and all state transitions and signal assertions occur on clock edges. The rst_n or reset signal initializes the controller and all dependent modules to a known state. During reset procedure, we cleared FIFO pointers, FSM state, routing grant, and output enable to prevent unwanted behaviors.

3.4.2 Input Admission and FIFO Control Signals

FIFO interface signals control packet acceptance into the controller. The fifo full signal means FIFO cannot accept more packets; the controller watches while not accepting writes. Fifo empty implies no further packet can be pushed out currently so dequeue is disabled. Wr en is fifo write enable, asserted by controller only when valid incoming packet needs to be written to FIFO && fifo is not full. Similarly rd en will be asserted by the controller only when fifo has data, routing decision is done & the chosen output port is waiting [14]. Thus safe orderly Enqueue / Dequeue is possible.

3.4.3 FIFO FSM Control Signals

The FIFO FSM enforces the correct sequencing of buffer operations under controller supervision. The fifo_state signal encodes the current operational state, such as IDLE, WRITE, READ, or WAIT [15]. The next_state signal is generated by the controller logic based on packet arrival, FIFO status, and output readiness.

3.4.4 FIFO FSM Control Signals

The FIFO FSM enforces correct sequencing of buffer operations under controller supervision. The fifo_state signal encodes the current operational state, such as IDLE, WRITE, READ, or WAIT. The next_state signal is generated by the controller logic based on packet arrival, FIFO status, and output readiness. The state_en signal enables state transitions only when legal conditions are met.

3.4.5 Routing Decision Signals

Routing logic is activated and controlled using explicit routing control signals. The route_en signal is asserted when a packet reaches the head of the FIFO and is ready for routing evaluation. The dest_addr field is extracted from the packet header and passed to the routing logic to determine the target output port. Out_sel: This is the 3-bit one-hot encoded signal to denote selected output port from the triplet [1:0]. Controller ensures that evaluation is done only once per packet, so as to avoid duplication.

3.4.6 Arbitrationand Grant Signals

Conflict is arbitraged using grant signals. Signals outputting the routing results are req_out0, req_out1 and req_out2. Grant_out0, grant_out1 and grant_out2. These are output from the controller and only one output per packet has one these asserts. Arb_valid indicates a valid decision has been made.

3.4.7 Synchronization Control Signals

Synchronisation signals help synchronism of the modules. Sync_en lets the FIFO and routing logic work synchronously and transfer data to output registers [14]. Data_valid_int is a self validity signal; it means that data is ready at this wire. Valid means data just showed up on the input, router should consume the packet cause it is now active. Busy means router is busy consuming of the packet and cannot take more than one packet. So this avoids overlapping of the packets.

3.4.8 Output Port Control Signals

The traffic controller sends packets through use of the output interface signals. The value out_valid represents an indicator for valid data to each output port [2:0] [2:0]. The out_en signal enables loading of the output registers. Out_ready sends feedback from the output ports and can identify if they are ready to accept [2:0].

3.4.9 Resource Release and Completion Signals

After successful forwarding, internal resources are released. The pkt_done signal indicates completion of packet transmission. The release_fifo signal allows advancement of the FIFO read pointer. The clear_grant signal deasserts arbitration grants for the next packet. These signals ensure that the controller progresses cleanly to the next packet. which prevents overlapping of packets.

3.4.10 Error Prevention and Protection Signals

The controller prevents illegal operations using guard signals. The illegal_wr signal flags write attempts when the FIFO is full. The illegal_rd signal flags read attempts when the FIFO is empty. The hold_state signal freezes the FSM state during unsafe conditions. The parity signal is used to detect bit errors in transmitted data. The error signal indicates when corrupted data is detected in the router [15]. The reset signal initializes the router and clears faults during abnormal conditions. The checksum or parity check logic verifies the correctness of packet data.

3.5 Router Controller

Its current state decides which output port should the data be transmitted to. A Router's crucial part-FIFO controller. FIFO controller allows data to be efficiently and securely transported between input port and output port. General working of Controller may be in the case of a 1x3 Router Controller: Controller is used to maintain that how the data flows in between input ports and the register along with the output-ports [16].

Controller's functioning is that it is often designed as an FSM finite state machine (FSM), a kind of circuit that might be in any single one of many states, to carry out this function. Its current state decides which output port should the data be transmitted to. A Router's crucial part-FIFO controller. FIFO controller allows the data to be efficiently and securely

transported between input port and output port. Communication networks use routers to transmit information consistently between various nodes. Usage of embedded system, multicore processor, cloud based application, high- speed communication interfaces & Network on Chip has resulted in the high increase in the amount of data to be shared in the modern digital world [11]. Packet processing in routers needs to cope with this demand for speed and for low latency, low throughput and security of contents. Router design is a key aspect in Network on Chip technology.

The input of proposed router is applied to input channel and forward data to one of three different output as per destination. Routing may sound easy, but the inside has to keep multiple, co-ordinated hardware blocks working in accord to allow reliable packet forwarding. Router consists of controller synchronizer, registering block, FIFO memories, address decoding, a mechanism to validation of packet, output interfaces.

First of all one would say routing is a continuous action beginning from receipt of packets and ending with successful transfer. In both steps it keeps on checking the validity of packet it will transmit, buffer condition, routing decision, synchronism condition and whether it has transmitted the packet or not [17]. Each of its modules have a specific function to perform and it is the combined interaction of the modules which makes the router work so efficiently. Its prime responsibility is to look through the incoming packet information into its ports and determine the precise route to forward packet on chosen channel. There will be routing information with every packet at the time of entrance into the router so its on the routing information on which its based and forward, it accordingly.

The operation of the router can be seen as an organized series of stages that work together to deliver packets reliably. First, in packet reception, incoming data reaches the router at the input port, where it is recorded and queued for processing. Followed by header analysis: checking the header to extract crucial routing info including the destination address and control bits for necessary routing options.

After deciding its destination, the packet enters buffering, where data is temporarily stored for efficient management when the output path is busy or during unexpected bursts. Subsequently, FIFO management ensures fair (first-in-first-out) processing, which prevents reordering. Meanwhile, synchronization control is vital, synchronizing data movement between various blocks so that reads and writes happen only at appropriate and stable times. Once everything falls into place, the packet is forwarded through its output path with minimal

delay and is delivered and ready for re-transmitting (Output transmission) to an external node or equipment [3].

These stages collectively help the router to manage data communication efficiently, assure the data integrity and to deliver low latency services under variable traffic load. A Router's crucial part-FIFO controller. FIFO controller allows the data to be efficiently and securely transported between input port and output port. Communication networks use routers to transmit information consistently between various nodes. Usage of embedded system, multicore processor, cloud based application, high- speed communication interfaces & Network on Chip has resulted in the high increase in the amount of data .

3.6 FIFO controller implemented in a 1x3 route

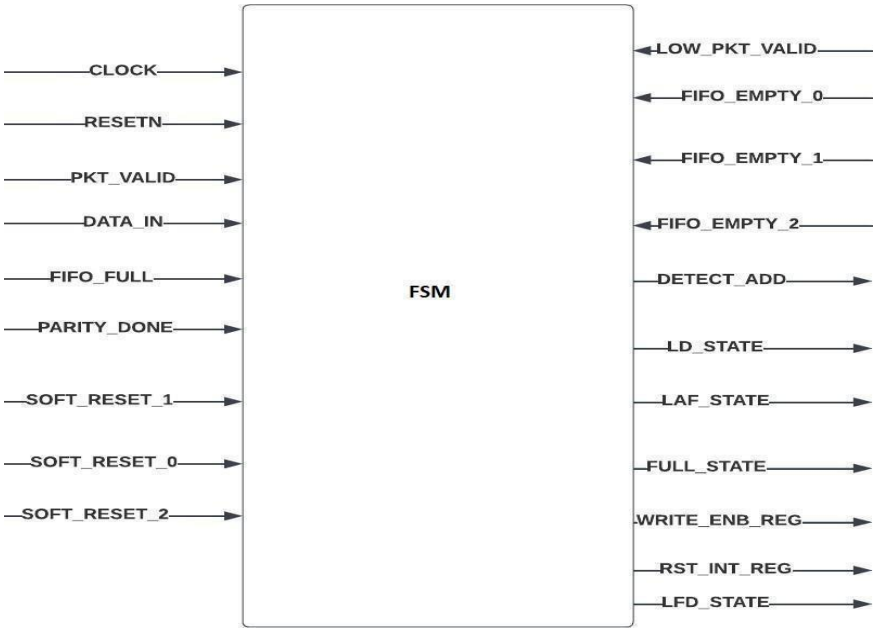

Figure 5 Finite State Machine Block

3.6.1 Input Port FIFO

These FIFO buffers are managed by the FIFO controller, which also manages the enqueue and dequeue processes. In the designed router, each of an input interface has an individual FIFO buffer. The arriving packets are accommodated temporarily in these buffers before they are processed internally. These buffers help to cushion any incoming bursts in traffic, because of which the router will be able to tolerate heavy traffic without dropping packets. All the FIFO buffers have a dedicated FIFO control unit which maintains, among other things, the occupancy levels of the buffers. Enqueue operation As and when the packets arrive at the input port, they are enqueued in the appropriate FIFO buffer by the FIFO

controller. FIFO enqueue procedure inserts the packets at the end of the buffer, maintaining the order of arrival.

3.6.2 Dequeue Operation

The FIFO control unit decides the packet to remove and queue to be sent. It again adopts one of the numerous arbitration procedures (based on priority or round robin) in selecting when the packets should proceed. The selected packet goes to the stage of forwarding and then transferred to proper output port. FIFO based arbitration is carried when the router is ready to receive data to an output port. The router control then has to decide the data to be taken out of the buffer and transfer the packet to the destination based on arbitration procedures; these are some scheduling rules like round robin of fairness or using priorities [6]. The top of the read register reads the actual data, ready at the head of FIFO. This read pointer is incremented accordingly when the packet is successfully delivered. It avoids error conditions from appearing.

3.6.3 Forwarding Operation

The packet that the FIFO controller required is send to the output port that determined by using routing table or some other routing. The process might include other work such as change of header, error checks. The packet is processed using the routing logic to find which output port the packet should go to once it has been selected for transmission. It makes this decision using routing tables or by decoding the destination field in the packet or by using configurable routing rules. In the process of forwarding it the router may perform other operations on the packet as for example modifying a control field or carrying out integrity checks or adapting format. The packet is then transferred to the chosen output channel with as small internal delay as possible.

3.6.4 Removal from FIFO Buffer

Packet are forwarded out; the FIFO controller takes the packet out of the corresponding FIFO hardware buffer. This creates a room to move the coming in packet into the buffer. This removal of packet moves the next packet to the front buffer, as read pointer will be moved to the released buffer entry after the packet has been successfully sent out. No data stored in the packet has to be shifted around just simply by using pointer to simulate removal of packet [17]; this makes the hardware buffer to become more efficient, by using up the buffer quicker, enabling the packets to follow through smoothly in case of heavy traffic.

3.6.5 Operation of Output Ports

Alongside output posts, the FIFO controller assists to transmit forward packets. These are received by the output post for delivery to destination network. An output may have flow-control to cater for bandwidth of downstream network. Coordinated interaction between input FIFOs, control logic, and output ports enable stable operation, reduce congestion, and high- throughput routing performance.

The packet is then transferred to the chosen output channel with as small internal delay as possible a control field or carrying out integrity checks or adapting format. The packet is then transferred to the chosen output channel with as small internal delay as possible.

3.7 Destination Address Decoding

Since it determines the path of any incoming packet, decoding of the destination address is perhaps one of the most critical operations in the router architecture. Once the header of a packet arrives and is stored in the register block, the router extracts the destination address and decodes it so that the appropriate output channel is recognized [18].This stage acts as the routing decision point where the packet is mapped to one of the available output interfaces.

In our design destination address is enclosed within the packet header and has fixed amount of bits. As when ever the packets are received at a router its address field is extracted and if it does turn - out that destination of the packet is available then routing information would be known to the router prior to any other type of packet processing.

Based on the decoding, binary destination value is translated into a series of internal signals that are used by various hardware blocks around the router. Instead of simply forwarding the packet, when decoding the destination the router verifies it corresponds to a valid output channel; by doing so it avoids invalid routing requests flooding through the system and ensures higher communication reliability.

For the proposed router, three output destinations are available:

Output Port 0

Output Port 1

Output Port 2

Outputs can be routed to destinations based on specific addresses. Each destination address will lead to an 'active' Output Path with an associated FIFO that has been designed for it No data stored in the packet has to be shifted around. When we get a destination from the decoder, its corresponding resources become active and rest is dormant. Eg. When the address represents Output Port 0, the decoder generates the required control signals for activation of FIFO 0 allowing Output Port 0 to be ready for the packet transmission in near further or if destinations corresponding to the Output Port 1 or 2 are activated to run similarly. Thus, packets reach the target destinations without the problem of conflicts getting aroused with each other.

"Packet Interpretation" (or Interpretation logic) is an "address decoder" to interface "Packet interpretation" logic with the "Routing control logic" and its output (decoded address information) is fed to as many functional units as required internal to the Router, each of them using this derived input in the desired manner [19].

3.7.1 Interaction with the Controller

The selected address is provided to the FIFO selection as this will tell the computer in question what buffer it should put the incoming Packet. Because each channel has its own different selection logic, meaning that the address must then be accurately decoded so that it can place it in the correct queue, or in the case of it being unable to decode the selected address, it could mix these channels and lose information in transmission.

3.7.2 Interaction with FIFO Selection Logic

The selected address is provided to the FIFO selection as this will tell the computer in question what buffer it should put the incoming Packet. Because each channel has its own different selection logic, meaning that the address must then be accurately decoded so that it can place it in the correct queue, or in the case of it being unable to decode the selected address, it could mix these channels and lose information in transmission.

3.7.3 Interaction with the Synchronizer Module

Once the packet reaches its proper FIFOs where destination information is read, the synchronizer will provide handshake mechanism to communicate these FIFOs and corresponding output interfaces

3.7.4 .Interaction with Output Control Logic

This is where the output control interacts with output routing logic. Routing decision is

decoded on the output control hardware to set up the communication path. In other words, only selected output interface is left untouched and unchecked output interfaces are inactivated. The reason for all the process is to avoid resource conflicts to maximally use communication and bandwidth [14].

3.7.5 Hardware Perspective of Address Decoding

The implementation of destination decoding in hardware is usually accomplished using combinational logic. The destination bits are continuously read and an output select signal is produced with the delay of some logic gates, for which no extra clock cycle is needed. As combinational logic responds upon an input change, the routing decision is quickly made and thereby the packet latency is lessened. When implemented in FPGAs/ASICs, to meet the timing requirements, the decoder must be carefully designed to operate quickly. Delay through decode will be a part of the critical path of the router chip. Minimizing the decode logic is therefore vital to higher operating frequencies translating to improved throughput.

3.7.6 Importance of Accurate Address Decoding

The addressing function's output affects router correctness and hence affects communication reliability. If the router decides to route a packet on the wrong path then the communication fails as the packet might eventually end up in the wrong destination. Therefore the router has built invalidators to verify the destination information that is extracted to be used in forwarding decisions. In practice, accurate address decoding helps in many ways toward reliable router operation: it ensures that packets are delivered to the destination for which they are meant thereby reducing the probability of making errors during routing.

Similarly communication reliability improvement is also achieved as the traffic is guided via the output path intended for it which essentially implies that only the appropriate output path is activated which gives rise to efficient utilisation of the output channels thus reducing packet latency and improving the throughput performance of the router. Moreover accurate decoding simplifies traffic management and allows better scalability in larger router architectures. The destination decoding logic is only a small portion of the router hardware. However, it affects the entire process; from beginning, when a packet first encounters an input port it relies on the decoder to decide the correct routing path to follow [11].

Hence the correct, fast and reliable decoding gives way to increased throughput, reduced latency of router together with increased packet delivery ratio along with reduction in resource utilization. In short, destination address decoding acts as the basis of routing intelligence of the proposed router. It takes destination information of the packet such as Destination Node IP address and transforms that information into control signals which set up the communication path, activate and use up appropriate FIFO resources of the router, control the controller of the router accordingly to that effect, send out the respective packets using that path to which is meant for it and ensures correct delivery of the packet [12]. The router incorporates validators into ensure that only the correct decoded information for proper transmission of packet is used. the case of it being unable to decode the selected address, it could mix these channels and lose information in transmission.

3.8 Timing Considerations in the 1×3 Router Architecture

The performance of digital router is heavily dependent on how fast its internals logic reacts on changed input of data and completes all required operations within one clock cycle. The proposed router's every packet has to go through several sequential and combinational stages before reaching the destination output port. All these stages take some part within total timing and the total defines maximum clock frequency, on which system can operate. In contrast to fully combinational systems, router uses fully synchronous design method. Datamovement and control operation is aligned to global clock signal.

Thus every module - starting with the FSM controller, FIFO blocks, address decoder and synchronizer works within required alignment. However, this is not all, what leads to the correct operation. At the same time the signal delay between each pair of registers could not overcome certain limits set by the setup and hold time. A packet entering the router does not immediately reach the output port. Instead, it is pipelined through a series of hardware blocks. Each block adds a finite delay, composed of time taken for logic evaluation, signal routing and memory access in the ordinary case we are dealing with a set of operations as: Inputsampling Registercapture Headerextraction Destinationdecision FIFO buffering Scheduling Synchronization Outputtransmission Every single arrow between operations means one timing sector. Slowest path may

restrict whole functioning of the router, because every single slow operation e.g. FIFO access) (or FSM logic may become a restriction even though most operations will consist more light calculations.

3.9 Critical Path Identification in the Router

Average delay is not the limiting factor for the maximum clock frequency of the router; instead, the critical path, or the longest propagation path [10] is the factor. This architecture has multiple potential Critical Paths under different traffic conditions and internal routing policies.

3.9.1 Controller FSM Path

The FSM reads the current state and input conditions, and creates next-state logic. This uses combinational decision making, which can be across multiple logic levels.

3.9.2 FIFO Read/Write Path

The FIFO operations include pointer arithmetic, access to the memory and the evaluation of the status flags. These operations are common and so the small delays add up and affect throughput.

3.9.3 Address Decoding Path

When a destination address is given, the header bits must be interpreted and converted to an output selection signal. It's simple, though it's in packet processing path.

3.9.4 Synchronizer Path

Handshake generation, timeout counting, and control signal validation form another timing-sensitive region, especially under high traffic or stall conditions. The overall system frequency is restricted by whichever of these paths exhibits the maximum delay [15].

3.10 Setup and Hold Constraints in Router Operation

Data exchanged between two consecutive blocks must satisfy two of the setup/hold time conditions to work as specified; 1. Setup requirement: The input data has to be stable prior to active clock edge (rising edge) for it to be successfully sampled. 2. Hold

Requirement: Once the clock edge is passed the data should not change for a short duration for it to be latched permanently and not overridden. This problem re-surfaces in the router (especially due to slow clock speed), at various stages whether while data is being written into the FIFO or while the FIFO read pointer is changing states or when the states of finite state machines change during handshake with synchronizer. Violation may not necessarily break down its logic; it could create erroneous behaviour when implemented in hardware, therefore impacting on some of the core performances of the router which include throughput, latency, ratio of packet delivered along with the resource usage.

3.11 Impact of Timing on Packet Throughput

The higher the router throughput, the more packets processed. Since the number of clock cycles required to pass a packet through the architecture is fixed, the system clock speed becomes the bottleneck. With careful timing, more packets are passed per second, FIFO congestion decreases and output channels are employed efficiently [16]. With poor timing, packets pile up in the buffers, FIFO overflow risk increases and the effective bandwidth drops. So timing is directly related to communication efficiency.

3.12 Latency Contribution of Internal Modules

Latency in the router is not caused by just one component; it is spread across all functional blocks. The register stage adds sampling delay. The decoder causes decision delay. The FIFO introduces buffering delay. The FSM contributes control decision delay. The synchronizer adds handshake delay.

3.13 Effect of Timing on Power and Hardware Efficiency

Faster clock frequencies can increase switching activity within combinational circuitry of the digital circuitry, exacerbating the problem even further so that is the trade off between increase dynamic power, poor optimized can require extra pipeline stage adding to area to this balance through optimized usage of clocking for faster output with lower number of latency. Therefore, physical implementation must be timing optimized where each of signal passes has been satisfy a particular speed and performance of operation within the chip [18]. To do so design constraint can be employed within tools to ensure that there should not be any timing violations.

Chapter 4 Synchronization of Design

This module performs operations like generation of soft reset signal and acts a packet authenticator. Condition of FIFO is used to verify the authenticity of a packet. Valid_out is generated by FIFO to transfer the data and it generates when the fifo signal (empty) is high. As the synchronizer module waited for 10 clock cycles after asserting the valid_out signal in synchronizer and before transmitting read enable, after which the synchronizer sent the read enable. This was because the soft reset logic was used in the synchronizer. Whenever read enables for two consecutive packets are not transmitted within the given time limit soft reset should be asserted which must be high [3]. FIFO also use the soft reset to reset FIFO. In proposed router a new control device is introduced called as synchronizer. Synchronization of all the data and the other functions are its responsibilities. In synchronous clock also hardware devices can be out of alignment due to time delay differences of data etc. Synchronizer takes care of these problems like when the data would be available in FIFO or when synchronizer gets out of data can transmit to a remote end for a safe transfer of data. Now here simple routing will fail as not only the routing logic but synchronization logic has to be applied i.e., whether the packet needs to be transmitted then data should be available at source FIFO and at the receiver end read enables should come within time limit. Else remote end can do a unsafe read operation.

4.1 Packet Readiness and Data Authenticity Evaluation

Once a packet is about to be sent from the router and needs to be valid and complete, synchroniser constantly keeps assessing the status of the FIFOs to obtain information on the availability of valid packet data to transfer. One of the main factors when assessing the status of the FIFOs is their occupancy, that is, whether they actually store any useful data inside them. If yes, the synchroniser determines a "packet ready" situation and does not allow the packet to be forwarded just yet, however, confirming the information that it is stable and indeed safe-to-relay. This stage ensures no empty memory block will be accessed and non-complete data written into the memory by synchroniser upon retrieving, avoiding unnecessary and invalid operations from that part, thereby the synchronization of signals continues on being produced by Synchroniser from the time that data gets stored in the memory until its the successful end of the communication [16].

4.2 Generation and Role of Valid_out Signal

After all packet checks have confirmed that valid packets can be moved out, the sync is ready to forward out valid packet and thus sends out the message via the Valid, out to report officially- valid packets now available to be consumed on receiving side Valid, out will be connection between buffering inside the router with the outputs consumption logic. Thus it will act as a mean for the sending the output that the information can be called forward then it will trigger transfer of data itself.

The Valid-out signal will then behave on its own accordance the in accordance with the control sequence where first fifo tells there is a presence of valid data later, status of buf is polled and then Valid-out is sent. Receiver is then aware of this and waits for ack The transfer of packs will then be ensured to occur under mutually approved conditions between send and receive modules

4.3 Internal Timing Mechanism of Synchronizer

From a digital design perspective, the synchronizer operates using a combination of combinational logic and sequential counters. The generation of Valid_out is based on immediate evaluation of FIFO status signals, while acknowledgment monitoring is implemented using a clock-driven counter.

Once Valid_out is asserted, a counter begins incrementing on every clock cycle. This counter defines the maximum allowable waiting period for a Read Enable response [19]. If the counter reaches its threshold value without receiving acknowledgment, the system a This structured timing model provides predictable synchronization behavior, deterministic timeout handling, and controlled packet flow regulation.

It ensures that the synchronizer operates in a stable and organized manner, allowing packet transfer to occur only under valid timing conditions and improving the overall reliability of the router.utomatically triggers a recovery event. This was because the soft reset logic was used in the synchronizer. Whenever read enables for two consecutive packets are not transmitted within the given time limit soft reset should be asserted which must be high [3]. FIFO also use the soft reset to reset FIFO. In proposed router a new control device is introduced called as synchronizer. Synchronization of all the data and the other functions are its responsibilities.

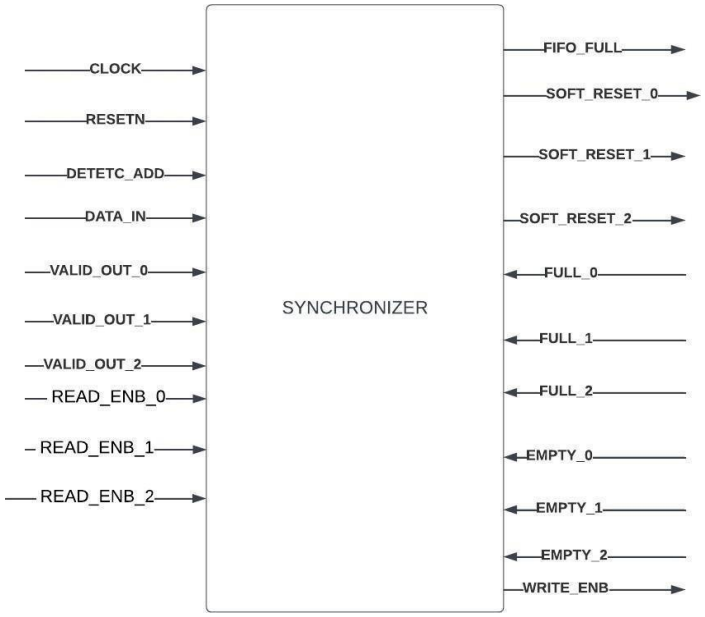

Figure 6 Synchronizer Block

Synchronization is important since the module creates a single FIFO-full signal by flip-flops via a integrating the entire state of all FIFO and informing the FSM about it [6]. The signal is sent via a series of flip-flops in the synchronization circuit to align it with the target clock domain across several clock cycles. The desired level of synchronization and the frequency difference between the clock domains determine how many flip-flops are used in the synchronization circuit.The synchronizer facilitates the transport of signals between multiple clock domains of the router 1x3 with safety and accuracy. So the internal parts of the router work correctly and handle all packets because the integrity of the data has not been affected and the synchronizer has prevented timing problems [12].

4.3.1 Channelizing Components

In register of submodule, it separates out header and payload and few parity data. To figure out the address of the destination it studies the signal coming from the controller FSM. An example would be if at the time of taking data, two signal lines which were paired at the time of taking input was high at once. It means detect-add and lfd_state signal line which were paired, had their values of high. At these type of conditions register takes data inside it as headers. If the ld_state flag is set to high, it treats received data as actual payload. It stores information in the internal register if full_state and fifo_full are found [17]. As soon as laf_state is found to be high, it starts sending newly obtained data to theoutput.

Chapter 5 Header Register

5.1 Introduction

Headers must be correctly interpreted and processed by switches/routers during Packet switched communication to carry out reliable and high performance data transmission. One prominent component of the switch/router's control unit that holds header information extracted from the incoming packs is called the Header register. Unlike payload, this information directly affects the routing decision, flow control, arbitration and QoS control mechanisms. In router's proposed architecture, it is the first stage where control-aware storage of header taken place after the packet arrival. It consists of the important metadata like destination address, priority of the packet and some protocol bits used for control. The isolation of this information than using it from payload helps in making quicker decisions leading to better latency with regards to forwarding of packets [19].

5.2 Role of the Header Register in Router Architecture

This header register serves as an interface between the data path (which handles the flow of the packet contents through the FIFOs and output ports) and the control path (which directs the router concerning further dispatch of the packet, based upon information found in the header). As well as serving such an interface function, information also stored in part in this header register would include: temporary packet header field storage which potentially enables some early routing decisions to be made even before the entire packet has arrived, support for arbitration and the selection of an output port (and, as a part of such decision making also supporting flow control and possible congestion handling) whilst maintaining all the information about the packet (metadata) throughout the routing procedure [5]. By decoupling header processing from payload forwarding, the router achieves improved throughput and predictable timing behavior.

5.2.1 Structure and Composition of the Header Register

The header register is typically implemented as a set of synchronous registers designed to latch is often encoded using a small number of bits, sufficient to uniquely select one of the three output channels.

Header fields during packet arrival. The internal structure depends on the packet format and router requirements but generally includes the following components. the router achieves improved throughput and predictable timing behavior. The header register stores the first flit of the packet that contains important control information. It temporarily holds the header before the router processes it. This allows the routing logic to read and analyze the packet details [15].

5.2.2 Destination Address Field

This field specifies where the packet should be sent out. In a 1x3 router, this is often a small number of bits that is enough to specify one of the 3 output ports..

5.2.3 Source Address Field

The source address may be included for debugging, monitoring, or acknowledgment purposes. While not always required for routing, retaining this information improves traceability and system observability [4].

5.2.4 Priority Field

The priority bit allows a network device like routers, etc. for distinguishing of packet according their importance /service-class. It enables arbi logic where several packet streams are competing on a common output port.

5.2.5 Protocol or Control Bits

This is a description of information about packets like their kinds, errors and status codes. It enables policies-based operations as well as conditionally routed responses.

5.2.6 Valid and Status Flags

Valid and Status Flags,Flags also tell you if there's a good value on top of that for your message; does it look finished yet.

5.3 Operation of the Header Register

5.3.1 Header Capture Phase

The headers are received when packets arrive to routers' inputs ports. During this phase, the header register captures the header fields on a clock edge, controlled by enable signals generated by the controller FSM [19].

5.3.2 Header Decoding Phase

After being saved they can be un-decoded through a combination of operations. The destination & precedence details are passed on for decision-making at a higher level prior that data has entered its queue completely (FIFO).

5.3.3 Routing Decision Support

Decoded target ip value affects output ports choice. At the same time, priorities information helps to solve a problem of multicomponent requests for an answer.

5.3.4 Header Retention and Release

Header registers are filled up till then when a data frame is passed through, received & processed at next stage of processing. The data of a message gets wiped out after being received and replaced with that from another one coming later on.

5.4 Interaction with Control Logic

The header register links with the control logic of the router, notably the Finite State Machine to control its function. Control signals are sent by the FSM, which enables, resets on updates the Header Register control function. Functions of the controller linking with Header Register: Header Load Enable, enables the Register to obtain header at time of receiving packets, Header Valid signal, for acknowledging a valid header, destination decode output, which drives the output port select signals, and priority output which feeds arbitrator and schedulers.

5.5 Timing and Synchronization Considerations

This header register runs alongside router clock cycle. The design of the header register should ensure that header information is stable at the beginning and middle of the cycle only [20]. And its output be consumed by the end of cycle. If there are two clock cycles, they have to carry out synchronization to move data from one clock cycle to the other. To avoid setup and hold time violations, careful thought might be given during the design of header register along with arbitration and control logic components of the router.

Impact on Performance and Latency The inclusion of the header register decreases routing latency. This mechanism enables routing and arbitration decisions to be initiated while the

payload is being buffered, as header information is available earlier in the packet processing cycle. This parallel processing capability brings about a profound impact on the router's overall throughput and provides numerous other benefits. By enabling early routing, throughput is dramatically enhanced, especially under high traffic loads. Additionally, a header register allows for more predictable packet-forwarding behavior, which is crucial for maintaining real-time traffic support. In traditional router architectures without a header register, routing decisions could not be made until the entire packet was received and its header information extracted, a process that increased latency and impeded overall performance [11].

This is achieved by increasing throughput, which can handle high traffic. The header register offers more predictable packet-forwarding and also has good support for real-time traffic. Because traditionally there is no provision in the architecture where the routing decision cannot be completed before the whole packet is received and its header information extracted, which in turn increases the latency badly affecting its performance.

5.6 Reliability and Error Prevention

The header information can be isolated and verified. By performing this at the router, we can check this for errors before wasting time decoding the packet itself. Malformed or invalid headers can be flagged as incorrect without consuming unnecessary resources, thereby making the entire router more reliable and robust. Additionally, the process can be more robust and efficient since the headers are processed separately to the payload. When the header is moved into a header register, the payload itself can remain stored in the FIFO or moving down the data path [2].

Processing the header now enables the router to verify whether that packet should continue further down the data path. In this way, error detection and recovery may be made robust by setting various status flags at the same time, and the architecture of the router becomes much stronger due to increased error detection.

5.7 Scalability and Design Flexibility

The header register module has a high level of scalability (we can always add more fields for virtual channels, quality of service enforcement and security tagging). In larger routers, there may also be several such fields acting in concert to support higher troughput or a

mutl-input configuration. Thanks to the nature of the header register, we can always add things on as we see fit. In the future that the router supports more and more input and output we should be able to expand our model of our router just as flexibly with minimum hassle. One could chain the routers in large networks and modularity means that if we are to change something on a part of the chip we need not change everything completely. Routing logic should accommodate different dimensions of the network [20]. Traffic and router architecture in the new design remain configurable. Hardware Implementation Considerations.

The hardware header register is light on area and power consumption as the header fields are relatively small compared to payload fields, so register size is minimal. It uses efficient designing of logic circuits to minimize area. It minimizes its switching activity during its operation to save power as well, while enabling high levels of data throughput meaning to handle multiple packets. With proper reset and control signals it operates with stability both in normal and abnormal cases [17].

5.8 Registers Payload Register

The actual data payload of data packets is stored in payload registers, which are utilised for this purpose. The data being transferred, like the content of a message or file, is contained in the payload.In order to manipulate, process, or inspect data, payload registers offer short-term storage for thepacket payload.

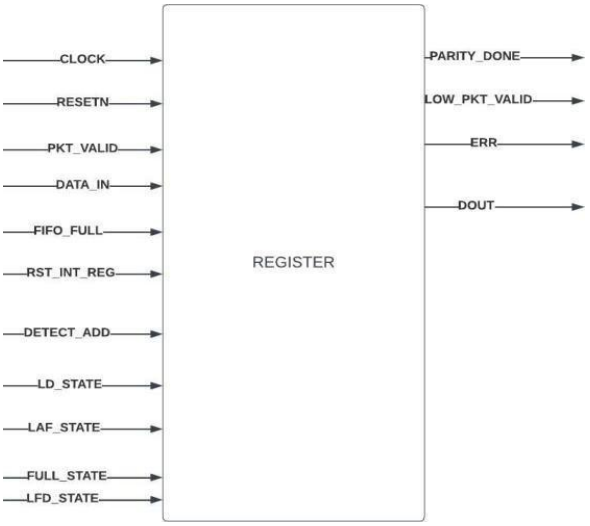

Figure 7 Register Block

The payload register is responsible for temporarily storing the actual data portion of a packet, excluding header and control information, ensuring that user data is preserved accurately during routing operations. It provides a stable storage interface between the input FIFO and the output forwarding logic, allowing payload data to be transferred without timing mismatches or data loss. The payload register supports sequential packet handling by holding payload data until routing, arbitration, and output port selection decisions are fully completed.

By isolating payload storage from control logic, the payload register enables parallel processing of routing decisions and data movement, which improves overall throughput and reduces latency With synchronization between header and parity registers, this ensures only verified and correctly routed payload data is forwarded across to the output port selected in the register before being forwarded across [21]. All through a modular design for the payload registers allows them to be adapted to be scaled more towards wider data as well as larger packet sizes to accommodate for future implementation in router enhancement as more output ports would appear making greater demand for data and larger packets as a requirement.

5.9 Parity Register

Data packet parity information is kept in parity registers, which are used to store parity information. Parity bits can be used for error detection/correction. Parity registers are used to easily calculate and store parity bits, which helps in ensuring the correctness of the packet being transferred. Once a packet is available, its parity can be computed using the content of packet data and stored into parity registers to detect errors

.Router logic (processing) reads the header data to determine the right output port for the received packet. Once the packet is ready for transmission, the payload information is taken out of payload registers and is forwarded to the output port [13]. At this point, the parity can be again verified with parity registers for data integrity. Parity register is a storage device on the router which is going to be used to calculate error parity bits from the data of the packet in order for the router to verify the correctness of it during buffering and transferring of the packet. It records the parity bit calculated (during packet reception time), later to be verified at the time of transmission against new parity bits. If a discrepancy, the difference is noticed [4].

Chapter 6 RESULTS AND DISCUSSIONS

Router: Results and DiscussionThis section details the outcomes of designing, building and testing the new router architecture. You will find a thorough examination of its operational performance and key characteristics. Our analysis concentrated on verifying its correctness, assessing how efficiently it routes packets, how well its buffers work, the dependability of its synchronization mechanisms and how stable the entire system remains across different scenarios.

Table 1 Coverage Summary by Type

Coverage Summary by Type:				
Weighted Average:				81.43%
Coverage Type	Bins	Hits	Misses	Coverage (%)
Statement	324	297	27	91.66%
Branch	146	126	20	86.30%
FEC Expression	6	6	0	100.00%
FEC Condition	98	68	30	69.38%
Toggle	612	340	272	55.55%
FSM State	8	8	0	100.00%
FSM Transition	21	15	6	71.42%

Table 2 Coverage Summary by Structure

Coverage Summary by Structure:	
Design Scope	Coverage (%)
TOP_MODULE_tb	81.43%
dut	79.61%

.

6.1 Functional Verification Results

Simulation of the design took place at functional level to check the correcting router operations. Scenarios were set up to test packet transmission depending on its destination information as well input port to either one of 3 output ports. The results displayed that all packets encoded / decoded

correctly and then routed from the input port to specified output ports without loss of actual packet data and in all cases [22]. verify the correctness of it during buffering and transferring of the packet. It records the parity bit calculated (during packet reception time), later to be verified at the time of transmission against new parity bits.This also gave verification of correct operation of the FIFO which preserved the order of the packets in time where a FIFO implies that the packets are ordered where the first-in, is first-out, giving verification as above.

Table 3 Device Utilization Summary

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Total no. Slice Registers	334	1920	17%
Number used as Flip Flops	310		
Number used as Latches	24		
Number of 4input LUTs	481	1920	25%
Number of occupied Slices	376	960	39%
Number of Slices containing only related logic	376	376	100%
Number of Slices containing unrelated logic	0	376	0%
Total number of 4 input LUTs	489	1920	25%
Number used as logic	481		
Number used as a route-thru	8		
Number of bonded IOBs	65	66	98%
IOB Latches	4		
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-clock nets	3.99		

The correct movement of the read/write pointers about enqueue and dequeue had been confirmed thus the operation logic for the storing packet was correct. The controllers routing generation selected the correct output port depending on the destination where different packets were routed to one of the 3 outputs correctly depending on the destination ID. This ensured working of the routing in selecting which output port and confirmed that the control signals connected up the data to the correct destinations via which output port given specific destination ID's and verified that the routing logic and control signal interaction with regard to router output was correct.

SIMULATION RESULTS

A full router design with simulated waveform observation is shown. 3 data output ports and 1 data input port is present. Whenever there's a positive edge, information is accurately transmitted in compliance with router policies and displayed on the output port [25].This proposed 1x3 Router contains 831.66 MB of memoryand consumes power of 80 mW.

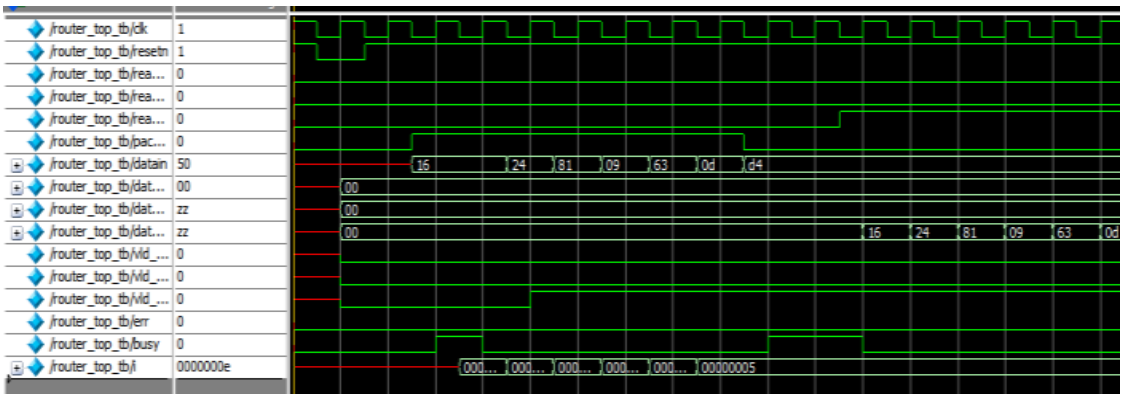

Figure 8 Payload 6

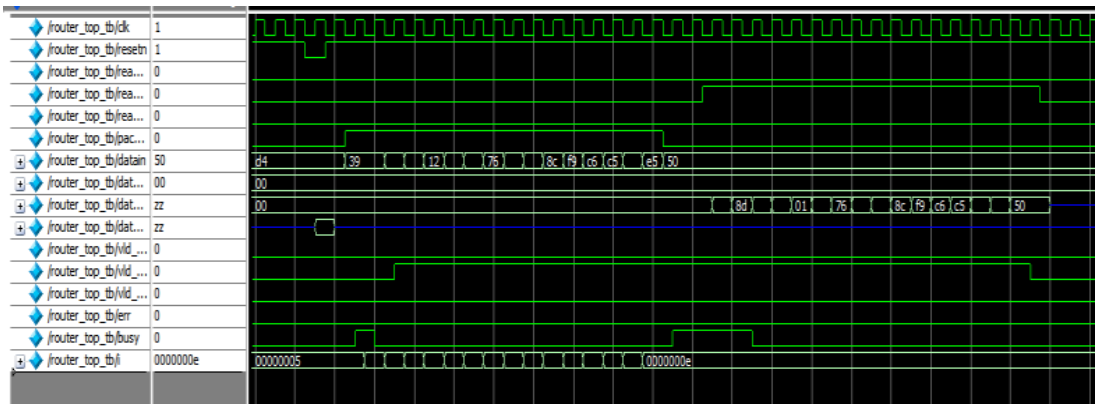

Figure 9 Payload 14

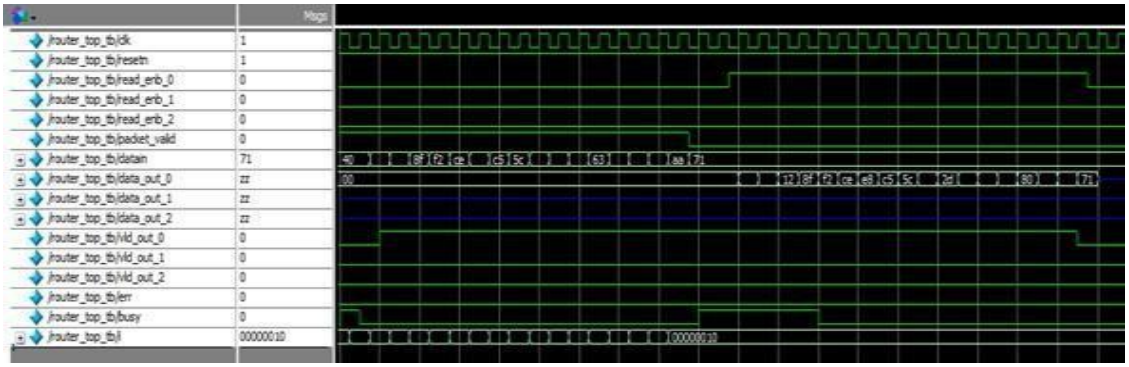

Figure 10 Payload 16

6.2 FIFO Buffer Performance Analysis

The FIFO buffer played a critical role in absorbing traffic variations and decoupling packet arrival from forwarding operations. Simulation results showed that the FIFO successfully handled bursty traffic conditions without packet loss, provided that buffer capacity was not exceeded.

When multiple packets arrived in consecutive cycles, the FIFO stored them sequentially, maintaining strict arrival order. Dequeue operations occurred only when routing and output readiness conditions were satisfied, preventing premature packet removal. No buffer underflow or overflow conditions were observed during valid operating scenarios, indicating robust buffer control.The FIFO FSM transitions were observed to be deterministic and well-sequenced. States such as idle, write, read, and wait transitioned correctly based on controller inputs and FIFO status flags [23]. This behavior ensured stable buffer operation even under continuous packet flow.

6.2.1 Controller Operation and Arbitration Results

The router controller demonstrated reliable coordination fall internal modules. It successfully regulated packet admission, routing evaluation, arbitration, and forwarding operations. During scenarios involving contention for output ports, the arbitration logic correctly granted access to onlyone packet at a time, preventing data collisions.

Fairness in arbitration was observed across multiple simulation cycles. Packets requesting the same output port were serviced in a controlled manner without starvation. This confirms that the controller's arbitration mechanism effectively balances resource usage.The controller also correctly prevented illegal operations. Attempts to write to a full FIFO or read from an empty FIFO were blocked by control logic, ensuring data integrity [22].

6.2.2 Synchronization and Timing Behavior

Synchronization logic proved essential in maintaining alignment between control and data paths. Simulation results showed that synchronizer signals successfully prevented timing mismatches between FIFOoperations, routing logic, andoutput registers.

Control signals such as read enable, write enable, and output enable were asserted only during valid data windows. No metastability or timing violations were observed, confirming the effectiveness ofsynchronization mechanisms in the router architecture.

6.2.3 Packet Forwarding and Output Port Results

Output ports correctly received packets from the internal switching logic. Output registers temporarily held packet data, ensuring stable transmissionto downstream components. Output valid and enable signals were asserted only when data was ready, preventing partial or corrupted transmissions.

Each output port operated independently, allowing packets to be forwarded without interference from other outputs [24]. This parallelism contributed to reduced latencyand improved throughput.

6.2.4 Latency and Throughput Observations

Latency analysis revealed that the router introduced minimal internal delay between packet arrival and forwarding. Most latency was attributed to FIFO buffering and routing decision cycles, which are inherent to packet-based systems.

The modular design allowed several operations—such as buffering, routing evaluation, and arbitration—to overlap in time. This concurrency improved throughput and allowed the router to sustain continuous packet flow under steadytraffic conditions.

6.2.5 Scalability Discussion

Although the implemented design focuses on a 1×3 router configuration, the results demonstrate that the architecture is inherently scalable. The modular nature of FIFO buffers, controller logic, and routing units allows straightforward extension to additional output ports. Control signals and arbitration logic can be expanded without major structural changes, making the design suitable for larger routers or network-on-chip applications [26].

6.2.6 Reliability and Robustness

The router exhibited stable behavior under all tested scenarios. Error prevention

mechanisms successfully blocked invalid operations, and the system recovered gracefully from idle . The combination of FIFO buffering, FSM-based control, synchronization, and structured arbitration contributed to reliable packet handling and predictable operation.

6.2.7 Comparison with Traditional Router Designs

Compared to basic or centralized router architectures, the proposed design demonstrates improved control clarity and operational discipline [25]. The separation of data path and control pathsimplifies verification and enhances maintainability. By incorporating explicit FSM control and synchronization logic, the router avoids many common pitfalls associated with timing hazards and uncontrolled buffering.

6.2.8 Discussion Summary

The results confirm that the proposed 1×3 router meets its design objectives of efficient packet routing, low latency, and reliable operation. FIFO buffers effectively manage traffic variations, the controller ensures correct sequencing and arbitration, and synchronization mechanisms maintain timing integrity [5].

Overall, the router demonstrates a balanced trade-off between simplicity and performance, making it suitable for both educational and practical networking applications.

Chapter 7 CONCLUSION AND FUTURE SCOPE

7.1 Conclusion

This thesis proposes a design of a router for packet based communication systems. The aim of this work is to develop a routing solution in hardware that can receive a data from a single input and route among three outputs efficiently. This can be possible through an integrated approach of FIFO buffering, FSM based control, synchronization & routing logic. Even router with simple logic can perform with reliability, if it is complemented with buffering and control. The FIFOs play important role here to handle the burst of data, reduce congestion and maintain sequence of packets. This guarantees that all incoming data is safely buffered and forwarded without any loss or degradation. F-SM based controller allows the router to function precisely without any ambiguity. Synchronised FSM along with the other block implements the operation of receiving data frames, decoding its address, handling buffering issues, selection of output and putting it into operation [12][16]. The router is now working as scheduled. Synchronization is an important aspect of the proposed design.

The synchroniser performs timing alignment as required in the hardware environment. It effectively synchronises the input to prevent problems arising due to change of states in flip flops at unpredictable instants. Hence makes the router more fault tolerant to problems like metastability and misinterpretation of events due the uncertain inputs. Now the router ensures proper timing of the data in hand. Further the proposed routine logic allows the movement of the data from the input side into the appropriate output so that the whole process becomes low latency.

The modularity of structure adds advantage to the router. The use of FIFO, controller and synchroniser each of which having there won importance. Each block works independently and can be developed, simulated and changed further without influencing or without problem within the complete hardware. The verification has been exercised and the router is working in normal conditions, reset and buffer full/empty [26].

To conclude, now that the whole router has been successfully completed and the correct operation of it verified, it becomes a useful element to be used for the packet routing in digital systems; it will transfer packets reliably, preserving content. It shows that it is useful to combine buffering with state-based systems and synchronize such a system. Not

only have all stated and required things about the design of the router been finished and verified as working properly, it also forms a good basis, in case this router was to be developed into a higher number of output addresses, for priority routing, etc [27][10]., which it can be developed against as part of a more sophisticated Network-on-Chip.

7.2 Key Insights

An important insight gained during the project was the significance of the buffering functionality necessary for proper packet communication. Without any FIFO storage, it is clear the router would suffer greatly from packet losses, overflows and wrong ordering in heavy traffic/temporary delay situations. By smoothing out such situations, it would give time for the controller to handle arriving packets gracefully. This is one of the important reasons the router works consistently even with erratic inputs. Second, control logic can greatly improve the robustness toward hardware errors. FSM based control of all submodules ensures the correct sequence of action and reduces possibility of conflicting operation, hence the possibility of incorrect design choice [28].

Robust control system with a well-defined cycle is much easier to verify and to debug; with limited time resources, it helps reducing chances of faults in final deployment. Thirdly, importance of proper synchronization with timing could never be stressed enough in digital communication. This guarantees that all incoming data is safely buffered and forwarded without any loss or degradation. F-SM based controller allows the router to function precisely without any ambiguity. Synchronised FSM along with the other block implements

In a router, data and signals should always be travelled synchronized. Without that synchronization, though logically correct, wrong data might reach wrong location. Not only a single bit error, the whole system might be corrupted. Such is the value of synchronizers at crucial locations. Last but not the least value of being modular has been well appreciated here; splitting the router into sub-modules means the blocks can be separately debugged and also allow additions if requirements suddenly changes to more destinations, priority or additional special packets and so on. This is one of the important reasons the router works consistently even with erratic inputs Also direct implementation using HDL was an experience to design some complex system by using existing resources such as FIFOs provided from Xilinx [29].

7.3 Future Scope

Though the current router implementation functions successfully and is relatively complete within its given scope, there are considerable opportunities for future improvement. One logical step is the enlargement of the number of output ports, evolving the design into a larger router for communication networks. Such an enhancement could increase its usefulness in larger NoC topologies or in multi-node embedded systems. Further along the line lies the opportunity of including priority based arbitration. Currently the router acts on forwarding packets correctly and in order, without any preference. In the future the router can cater for higher priority packets when several routing requests occur simultaneously [11][6]. This option would also improve its usage where time sensitive/performance critical data streams have to be delivered with preference. The router may also be made more intelligent to carry out QoS features. Incorporating intelligent QoS routing would expand its functionality to enable treatment of traffic classes in different ways with regards to latency, congestion policies etc. This function would clearly increase its applications value and use where different classes of data demands different level of treatment.

The next possible enhancement is the inclusion of adaptive routing strategy. Such a capability would greatly benefit system efficiency based on the real time traffic conditions, congestion detection and ability to route. Such a scheme aims to relieve heavy traffic within an overloaded network for better network efficiency. Yet another area to consider could include adding error detection and correction. While the router acts with sufficient reliability through and synchronization control logic, adding any extra protection by way of error handling to it will only increase router performance. Performance optimization presents yet another interesting area in future.

Given the nature of some demanding applications, it is plausible to improve performance parameters at a particular clock frequency for minimum latency, high throughput and optimal speed. Hence makes the router more fault tolerant to problems like metastability and misinterpretation of events due the uncertain inputs. Now including priority based arbitration. the router ensures proper timing of the data in hand. Further the proposed routine logic allows the movement of the data from the input side into the appropriate output so that the whole process becomes low latency. This can be done by using suitable methods of optimizing the control logic and buffer schemes etc [23]. This will be beneficial for its possible usage in current or future networks.

7.4 Final Conclusion

Next-generation SoC and NoC architectures could adopt the developed design. A router has to process huge volumes of data, with negligible time lag and strong reliability constraints. For these larger systems, the principles elucidated in this thesis: structured buffering, state-based control and synchronization, with its modular and well-defined control logic, serve as a fitting foundation. By using a well-defined finite state machine (FSM), the control logic ensures deterministic and accurate routing decisions. Routing decisions are made precisely and in synchronization with packet receiving, arbitration, buffering and forwarding by simply mapping the control states to Verilog control signals.

The use of synchronization logic makes sure that data transfer across the different timing domains happens safely, reducing the risk of metastability occurrence and eventually corruption of data, leading to a robust router design that could find itself at home with complex digital systems of today where different clock domains do exist. The modular and layered composition has enhanced router design scalability as well as maintainability. In case any change or upgrade has to be made with a certain functionality of FIFO, controller or synchronizer and registers can be made without touching the remaining ones. From hardware implementation perspective, the router design is memory cautious and more inclined on making use of flip-flops than simple logical components, so that it can accommodate FPGA and ASIC implementation easily. From its functionality and specifications the router works fine about routing of packets, handling FIFO full and empty scenarios, during reset and controlling of logic [5].

The router design can be extended up to any number of nodes which is desired besides further enhancements, such as priority-based arbitration, QoS and use of adaptive-type routing. Summing it all together, successful implementation of the router and employing a modular and structured style highlights the approach of control logic through the use of flip-flops in conjunction with FIFO and synchronizing signals. The design finds itself in an area as an extended, versatile, next-generation network-on-chip as well as embedded router solution for communication demands.

REFERENCES

[1] S. David Kattepur and S. K. Mohalik, "Model-based reinforcement learning for router port queue configurations," *Intelligent and Converged Networks*, vol. 2, no. 3, pp. 177–197, Sept. 2021, doi: 10.23919/ICN.2021.0016.

[2] D. Li, D. Dong, Z. Lu, and X. Liao, "RoB-Router: A Reorder Buffer Enabled Low Latency Network-on-Chip Router," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 9, pp. 2090–2104, Sept. 2018, doi: 10.1109/TPDS.2018.2817552.

[3] M. Katta, T. K. Ramesh, and J. Plosila, "SB-Router: A Swapped Buffer Activated Low Latency Network-on-Chip Router," *IEEE Access*, vol. 9, pp. 126564–126578, 2021, doi: 10.1109/ACCESS.2021.3111294.

[4] S. Swapna, A. K. Swain, and K. K. Mahapatra, "Design and analysis of five port router for network on chip," in *2012 Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics*, Hyderabad, India, 2012, pp. 51–55, doi: 10.1109/PrimeAsia.2012.6458626.

[5] Gopal N., "Router 1x3 RTL Design and Verification," *International Journal of Engineering Research and Development*, vol. 11, issue 09, 2015.

[6] J. Kwak, S. Kim, D. Kim, and H. Yoo, "A Low-Latency and Energy-Efficient Router Design for 3D Networks-on-Chip," *IEEE Transactions on Computers*, vol. 67, no. 5, pp. 667–680, 2018.

[7] A. S. Tanenbaum and D. Wetherall, *Computer Networks*, 5th ed. Boston, MA, USA: Pearson Education, 2011.

[8] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. San Francisco, CA, USA: Morgan Kaufmann, 2004.

[9] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks: An Engineering Approach*. San Francisco, CA, USA: Morgan Kaufmann, 2003.

[10] M. M. Mano and M. D. Ciletti, *Digital Design: With an Introduction to the Verilog HDL*, 5th ed. Upper Saddle River, NJ, USA: Pearson, 2013.

[11] N. McKeown, “The iSLIP scheduling algorithm for input-queued switches,” *IEEE/ACM Transactions on Networking*, vol. 7, no. 2, pp. 188–201, Apr. 1999.

[12] P. P. Pande *et al*., “A scalable and energy-efficient network-on-chip router architecture,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 4, pp. 513–527, Apr. 2009.

[13] S. Kumar *et al*., “A network on chip architecture and design methodology,” in *Proc. IEEE Computer Society Annual Symposium on VLSI*, 2002, pp. 117–124.

[14] *IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks*, IEEE Std 802.3, 2018.

[15] L. Benini and G. De Micheli, “Networks on chips: A new SoC paradigm,” *IEEE Computer*, vol. 35, no. 1, pp. 70–78, Jan. 2002.

[16] A. Jantsch and H. Tenhunen, *Networks on Chip*. Boston, MA, USA: Kluwer Academic Publishers, 2003.

[17] A. Hemani *et al*., “Network on chip: An architecture for billion transistor era,” in *Proc. IEEE NorChip Conference*, 2000.

[18] W. J. Dally, “Virtual-channel flow control,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194–205, Mar. 1992.

[19] C. J. Glass and L. M. Ni, “The turn model for adaptive routing,” in *Proc. 19th Annual International Symposium on Computer Architecture*, 1992, pp. 278–287.

[20] T. Hollstein, H. Bringmann, and W. Rosenstiel, “A formal approach to deadlock-free routing in NoC,” in *Proc. Design, Automation and Test in Europe (DATE)*, 2006.

[21] M. J. Garzarán, M. Arana, and E. Ayguadé, “Performance evaluation of buffering techniques in network-on-chip routers,” in *Proc. International Conference on Embedded Computer Systems*, 2007.

[22] K. Goossens, J. Dielissen, and A. Radulescu, “The Aethereal network on chip: Concepts, architectures, and implementations,” *IEEE Design & Test of Computers*, vol. 22, no. 5, pp. 414–421, Sept.–Oct. 2005.

[23] M. S. Abdelfattah and A. M. Eltawil, “Low-latency router design for energy-efficient network-on-chip systems,” *Microprocessors and Microsystems*, vol. 39, no. 6, pp. 459–470, Sept. 2015.M. Morris Mano and M. D. Ciletti, *Digital Design: With an Introduction to the Verilog HDL*, 5th ed. Upper Saddle River, NJ, USA: Pearson, 2013.

[24] S. S. Sapatnekar and V. B. Akella, “Timing and synchronization challenges in high-speed digital systems,” in *Proc. IEEE International Conference on Very Large Scale Integration*, 2011.

[25] R. Marculescu, U. Y. Ogras, L.-S. Peh, N. E. Jerger, and Y. Hoskote, “Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 1, pp. 3–21, Jan. 2009.

[26] A. Kumar, N. Concer, and L. P. Carloni, “Pattern-aware routing and buffering in NoC,” *IEEE Transactions on Computers*, vol. 62, no. 4, pp. 653–666, Apr. 2013.

[27] J. Kim, J. Balfour, and W. J. Dally, “Flattened butterfly topology for on-chip networks,” in *Proc. 40th Annual International Symposium on Microarchitecture*, 2007, pp. 172–182.

[28] M. Radetzki, C. Hao, P. Gunasekera, and B. Nowakowski, “Methods for fault tolerance in network-on-chip topologies,” *ACM Computing Surveys*, vol. 46, no. 1, Art. 8, 2013.

[29] A. Agarwal, C. J. Ray, and S. Ghosh, “Verification methodologies for packet-switched digital IPs using SystemVerilog,” in *Proc. IEEE International Symposium on VLSI Design and Test*, 2020.

Yesha MTech

ORIGINALITY REPORT

7%	6%	3%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	tudr.thapar.edu Internet Source	2%
2	ebin.pub Internet Source	<1%
3	Submitted to Indian Institute of Technology, Bombay Student Paper	<1%
4	Submitted to University of Technology Student Paper	<1%
5	Submitted to Thapar Institute Of Engineering and Technology, Patiala Student Paper	<1%
6	Submitted to Thapar University, Patiala Student Paper	<1%
7	www.jetir.org Internet Source	<1%
8	link.springer.com Internet Source	<1%
9	thinkmind.org Internet Source	<1%
10	si2.epfl.ch Internet Source	<1%
11	ns3simulation.com Internet Source	<1%
12	appliancemind.com Internet Source	<1%
13	Monika Katta, T K Ramesh, Juha Plosila. "SB-Router: A Swapped Buffer Activated Low Latency Network-on-Chip Router", IEEE Access, 2021 Publication	<1%
14	chinetti.me Internet Source	<1%
	etheses.whiterose.ac.uk	

15	Internet Source	<1%
16	jantsch.se Internet Source	<1%
17	mafiadoc.com Internet Source	<1%
18	www.utupub.fi Internet Source	<1%
19	Submitted to Caledonian College of Engineering Student Paper	<1%
20	Submitted to University of Gloucestershire Student Paper	<1%
21	V. Sekhar, P. Ajay Kumar Reddy, K. Logesh, T. Raghunadha Reddy, Dara Raju. "Next-Generation Smart Systems - Bridging Sustainability and Computational Intelligence", CRC Press, 2026 Publication	<1%
22	ukdiss.com Internet Source	<1%
23	Submitted to (school name not available) Student Paper	<1%
24	www.kth.se Internet Source	<1%
25	Y. Unno. "INFIERI 2021: Hands-on Lab – LiDAR", Journal of Instrumentation, 2023 Publication	<1%
26	export.arxiv.org Internet Source	<1%
27	Feihui Li. "Design and Management of 3D Chip Multiprocessors Using Network-in-Memory", ACM SIGARCH Computer Architecture News, 5/1/2006 Publication	<1%
28	digitallibrary.usc.edu Internet Source	<1%
29	www.tdx.cat Internet Source	<1%
30	annals-csis.org Internet Source	<1%
	core.ac.uk	

31	Internet Source	<1%
32	erepository.uonbi.ac.ke Internet Source	<1%
33	etd.repository.ugm.ac.id Internet Source	<1%
34	faculty.kfupm.edu.sa Internet Source	<1%
35	soc.ece.ubc.ca Internet Source	<1%
36	sxauroratsubasa.sakura.ne.jp Internet Source	<1%
37	www.journal.uestc.edu.cn Internet Source	<1%
38	Ronild Hako, Evjola Spaho. "Chapter 24 From VANETs to Software-Defined VANETs: A Comprehensive Review of Architectures, Routing Strategies, and Open Research Challenges", Springer Science and Business Media LLC, 2026 Publication	<1%
39	Xin Jin, Peng Jing, Lin Wang, Yuzhou Dai, Siyuan He, Zhao Ma, Wei Zhang. "High throughput VLSI design for real-time JPEG 2000 embedded block coding", Journal of Real-Time Image Processing, 2025 Publication	<1%
40	Zhiqiang Chen, Yongwen Wang, Hongwei Zhou, Mengjin Li. "Priority-aware deadlock recovery algorithm for multi-chiplet systems", 2024 IEEE 30th International Conference on Parallel and Distributed Systems (ICPADS), 2024 Publication	<1%
41	azpdf.tips Internet Source	<1%
42	dokumen.pub Internet Source	<1%
43	ir.nctu.edu.tw Internet Source	<1%
44	sciencepublishinggroup.com Internet Source	<1%
45	silo.pub Internet Source	<1%

46	waset.org Internet Source	<1 %
47	www.coursehero.com Internet Source	<1 %
48	www.csl.cornell.edu Internet Source	<1 %
49	www.cybok.org Internet Source	<1 %
50	www.doria.fi Internet Source	<1 %
51	www.ijert.org Internet Source	<1 %
52	www.researchgate.net Internet Source	<1 %
53	www.simula.no Internet Source	<1 %
54	Networks on Chip, 2003. Publication	<1 %
55	Design Automation and Test in Europe, 2008. Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		