AN ADAPTIVE ROUTING ALGORITHM FOR 3-D NETWORK-ON-CHIP (NOC)

By

SAPNA TYAGI

(SAP ID-27037)

SCHOOLOF COMPUTER SCIENCE

SUBMITTED

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF THE

DEGREE OF

DOCTOR OF PHILOSOPHY

TO

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

DEHRADUN

(March 11 , 2019)

UNDER THE GUIDANCE OF

DR. AMIT AGARWAL	DR. VINAY AVASTHI	DR. PIYUSH MAHESHWARI
SUPERVISOR	C0-SUPERVISOR	EXTERNAL SUPERVISOR
PROFESSOR (on leave)	ASSOCIATE PROFESSOR	DEAN – ENGINEERING, TECHNOLOGY AND ARCHITECTURE
UPES DEHRADUN	UPES DEHRADUN	AMITY UNIVERSITY DUBAI UAE

ACKNOWLEDGEMENT

All praises to Almighty God for the strengths and blessing during the duration of this PhD study . The duration of this journey have all moments of ups , downs , joy, frustration , sorrow , confusing and happiness. So it is important for me to express my gratitude and appreciation to various people who have inspired , encouraged and supported me in completing this thesis work.

I would like to express my deep and sincere gratitude to my supervisor Dr. Amit Agarwal for patient guidance, unremitting effort, professional advice ,useful critiques at all stages of my research work. I thank him for believing in me and giving me the chance under his supervision to complete my Ph.D.

I would also like to express my deep appreciation to my co-supervisor Dr. Vinay Avasthi for his valuable academic advice, inspirations, assistance and guidance that helped me through my Ph.D. thesis.

I would like to acknowledge the advice and support from my external supervisor Dr Piyush Maheshwari . He has always given me valuable advice and discussions.

Many thanks to Dr. Manish Prateek (Professor & Dean School of computer science) for his valuable suggestions and comments during my synopsis and abstract presentation that helped me in refining my work a lot .

Last but not least , Special thanks to my Parents , Husband and my lovable son who always supporting me to pursuit my dream and always by my side to give moral support . Sincere thanks to all my friends.

[illegible]

Sapna Tyagi

CANDIDATE'S DECLARATION

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

[illegible]

Sapna Tyagi

Date: March 11, 2019

UPES
UNIVERSITY WITH A PURPOSE

THESIS COMPLETION CERTIFICATE

This is to certify that the thesis on "An Adaptive Routing Algorithm for 3-D Network-on –chip (NoC)" by Sapna Tyagi in Partial completion of the requirements for the award of the Degree of Doctor of Philosophy (Computer Science) is an original work carried out by her under our joint supervision and guidance.

It is certified that the work has not been submitted anywhere else for the award of any other diploma or degree of this or any other University.

[illegible]

Dr. Amit Agarwal
Supervisor
Professor (on leave)
UPES Dehradun

[illegible]

Dr. Vinay Avasthi
Co-Supervisor
Associate Professor
UPES Dehradun

Date: March 11, 2019

AMITY UNIVERSITY DUBAI

THESIS COMPLETION CERTIFICATE

This is to certify that the thesis on "**An Adaptive Routing Algorithm for 3D Network on Chip (NoC)**" by **Sapna Tyagi** in partial completion of the requirements for the award of the Degree of Doctor of Philosophy (Computer Science) is an original work carried out by her under my supervision and guidance.

It is certified that the work has not been submitted anywhere else for the award of any other diploma or degree of this or any other University.

External Guide

Piyush Maheshwari
24 Feb 19

Prof. Dr. Piyush Maheshwari

Amity University Dubai

United Arab Emirates

EXECUTIVE SUMMARY

Future, smart devices will contain billions of transistors that will make up tens to hundreds of IP cores. These devices with built- in IP cores implement emerging and complex multimedia and networking applications to offer a rich range of network and multimedia services. In order to maximize available resource use, effective collaboration (e.g. data transfers) between the IP cores is required. There are many difficulties in designing systems that make up of a large number of cores. Another challenge is to make effective use of the available hardware resources and to make compatible application mapping with them. Network- on- Chip(NoC) architecture is an architecture that can fit all these cores and satisfies the need for communication and data transfer. For these reasons, Networks- on- Chip is a popular choice in terms of designing and supporting the on- chip interconnection for new MPSoCs (Multi Processor System on chip). A typical NoC application consists of several components, such as general - use CPUs, specialized cores and embedded hardware that are connected together over a complex communication architecture. The NoC architecture is functionally implemented as layered network protocol stack similar to the OSI model (Open system Interconnection) .NoC has outperformed traditional shared bus communication approach by providing packet switched data transfer across all the processing peripheral in the high-performance embedded systems. As the technology advances and the number of IP cores increased on a single chip, the two-dimensional NoC are no longer exhibiting desired performance. The performance improvement resulting from the architectural advantages of NoCs will be significantly improved if 3D Integrated Circuits (ICs) are used as the primary method of manufacturing. The 3-D ICs can achieve significant performance advantages over 2-D ICs based on the electrical and mechanical properties of the new geometric arrangement. The 3-D integration approach and process technologies of VLSI enables innovative design opportunities in 3-D NoC . There exist a lot of technical challenges which needs to be addressed to build a viable three-dimensional on-chip network. These challenges are apparent due to the scaling of transistor size and increasing number of cores in 3D NoC. These 3D NoC s are generally more susceptible to various types of reliability problems and must therefore deal with data communication challenges like fault, congestion, deadlock and livelock . The 3D NoC must maintain its functionality and graciously degrade its performance in the presence of above mentioned challenges in order to provide a reliable communication service. One of the aspects is to design a routing algorithm to deal with these challenges. This leads to the need for research and the development of better adaptive NoC routing algorithm to meet the growing demands of the larger NoC, and that will be capable of handling data communication challenges while keeping the optimum performance.

In this thesis, we have classified the work in three phases:

- **Empirical Investigation of Two-Dimensional NoC**: This study is presented in chapter five, where 2D NoC topologies like mesh, torus, fattree and cmesh are investigated. The topology parameters like node degree, diameter, and routers required and network cost of the topologies are presented. The comparative study of topologies are done with the help of Booksim 2.0 simulator. The latency, throughput, number of hops comparison for all the topologies are evaluated and analysed with respect to varying range of injection rate. The algorithm used for this analysis is one of the very popular routing algorithm dimension order routing i.e. XY routing. After the comparative study of all topologies, it is concluded that mesh and torus can be the good choice for NoC architectures. The impact of virtual channels on throughput is also presented for all the four topologies. The virtual channels are used to assure the data flow in the deadlock and congestion situations. As we increase the number of virtual channels it increases the throughput. Despite the advantage the virtual channels are costly to implement. Afterwards odd even and adaptive algorithm are also implemented for the 2D mesh using Booksim 2.0 simulator. The adaptive algorithm works in dynamic manner if there is traffic congestion adaptive approach is used otherwise the algorithm routes deterministically. Odd Even algorithm is very common turn model routing used for avoiding deadlock, livelock and congestion situation. The performance metrics defined as the ratio of average latency to average throughput is also calculated and compared for the three routing algorithm XY, Odd Even and adaptive. The performance metric of adaptive algorithm is higher than Odd Even and XY. This topology and algorithm analysis helped us to have elaborative and better understanding of two-dimensional NoC.
- Empirical Investigation of Three-Dimensional NoC: This study is presented in chapter six, where 3D Mesh based NoC is investigated. The major reasons for adopting 3D NoC is to accommodate larger number of IP cores for upcoming smarter devices. Therefore it is extremely compulsory in the presented work to understand the data communication among IP cores in 3D NoC. The Existing routing algorithms like XYZ, adaptive and turn model "Negative-First" are implemented with the help of Access Noxim simulator. XYZ algorithm is easy to implement and deadlock free. Adaptive algorithm avoids congestion and improves latency by using extra logic for determining the network state before doing routing. Adaptive algorithm uses virtual channel while routing which are pretty costlier to implement. Therefore "Negative-First" turn model is also implemented. Turn model algorithms are useful approaches for avoiding deadlock, livelock and congestion situation. Similar to 2D

NoC , the latency and throughput for all the three algorithms are evaluated against injection rate and compared for a better understanding of 3D NoC.

- Proposed Work: This is presented in chapter seven, where we have proposed TFRF routing algorithm to route the data in 3D mesh topology. The 3-D mesh is scalable and recursive structure with constant node degree. The mesh topology is grid based structure where each node can be identified as (x,y,z) coordinate which makes the implementation of communication algorithm easier. The TFRF routing algorithm is implemented majorly in two aspects. First, how to implement the routing model to improve all possible parameters like end-to-end delay, throughput, latency, overhead and buffer occupancy to enhance the fault tolerance while routing in 3D NoCs; another aspect, how to enhance the model's traffic flow to ensure the effecient execution and unwavering quality of 3DNoCs. The TFRF routing model is accompanied with five turn activation rules
  that is capable of handling challenges like deadlock, congestion, and fault while keeping performance optimum. The performance parameters are evaluated and compared with the latest techniques present for 3D NoC. The algorithm's complexity is O(n). The routing algorithm proposed is compared with the AFRA, HamPa & HARS algorithm and the results of the comparison show that the routing algorithm TFRF has better performance with respect to latency and throughput .

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ITRS	International Technology Roadmap for Semiconductors
PCs	Processing Cores
SoCs	System on Chip
DSM	Deep Sub-Micron
NoC	Network-on-Chip
IP	Intellectual Property
PE	Processing Elements
NI	Network Interfaces
ASNoC	Application-Specific Networks-on-Chips
MAC	Medium Access Control
TDMA	Time Division Multiple Access
DTDMA	Dynamic Time Division Multiple Access
OCP	Open Core Protocol
VCI	Virtual Component Interface
AEI	Advanced Extensible Interface
DTL	Device Transaction Level
VC	Virtual Channel
QoS	Quality of Service
GS	Guaranteed Service
BE	Best Effort
GS	Guaranteed Service
D	Diameter
ND	Node Degree
CLICHÉ	Chip-Level Integration of Communicating Heterogeneous Elements
SPIN	Scalable, Programmable, and Integrated Network
DOR	Dimension Order Routing
TTL	Time to live
DoA	Degree of Adaptiveness
OE	Odd-Even
PR	Pseudo-Receiver
NMOE	Non-Minimal Odd-Even
EDAR	Efficient Dynamic Adaptive Routing
EBL	Effective Buffer Length

VPM3NoC	Vertical Partial Mesh-based 3D NoC
CH	Cluster Head
TSV	Through Silicon Vias
XNoTs	Xbar-associated Network-on-Tiers
MIRA	Multi-layered on-chip Interconnect Router Architecture
BBVC	Bidirectional Bisynchronous Vertical Channels
FT-DyXYZ	Fault-Tolerant Dynamic XYZ
LA-XYZ	Look Ahead XYZ Algorithm (LA-XYZ)
LAFT	Look Ahead Fault Tolerant
FMS	Fault Control Module
HLAFT	Hybrid-Look-Ahead-Fault-Tolerant
TAAR	Topology Aware Adaptive Routing
SBSM	Source Based Shortest Manhattan
DBSM	Destination-Based Shortest Manhattan
VL	Vertical Link
EBAR	Energy and Buffer-aware fully Adaptive Routing Algorithm
ETE	End-to-End
ACO-FAR	Ant Colony Optimization-based Fault-Aware Routing
NW	North-West
SW	South-West
EN	East-North
ES	East-South
WN	West-North
EN	East-North
SE	South-East
FSH	Fault Status Handler
FTDR	Fault-Tolerant Deflection Routing
FDWR	Force-Directed Wormhole Routing
TLM	Transaction-Level Model
RC	Route Computation
SA	Switch Allocation
ST	Switch Traversal
VCA	Virtual Channel Allocation
FSMs	Finite-State Machines

TSV	Through Silicon Vias
BIST	Built-In Self-Test
ECC	Error Correcting Codes
CA	Congestion Aware
HPU	Header Parsing Unit
AL	Arbiter Logic
CI	Congestion Index
RCA	Regional Congestion Awareness
2D	Two-dimensional
3D	Three- dimensional
ICs	Integrated Circuits
TFRF	Triggered Faulty-free Route Forwarding
PR	Provisional Relay
PD	Path Diversity
NA	Non-faulty Area
GAL	Global Average Latency

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

1.1 OVERVIEW

As per the prediction made by International Technology Roadmap for Semiconductors (ITRS) [1], the transistor feature size will follow the trend and will be smaller, and the density of the transistor will be doubled every two years. These integrated circuits will have the capability to operate under one volt and thus well suited for new generation smart devices. Another evolving trend in the semiconductor industry as we have seen in the past years from the generation of processors is the reduced cost and high performance. The smaller transistor size will also enable to have more transistors on a single die. The expanding number of transistors drives the expansion of the number of Processing Cores (PCs) that can be incorporated onto one chip. A large number of computer resources available on System on Chip (SoCs) places enormous demands on the resources for communication.

Furthermore, the shrinking feature size in the Deep Sub-Micron (DSM) era continually pushes interconnection delays and power consumption as the main factors in the optimization of modern systems. The major issues related to the evolvement of this technology are the design concepts and methods for implementation in the devices. Another challenge lie here is that direct impact of slow voltage scaling is the slow increase of operation frequency constrained by the power supply and the heat dissipation challenges. Table 1.1 [1] shows the trends of functions per chip.

Table 1.1: Transistor feature trends [1]

Year for Production	Functions per Chip at Introduction (Million Transistors)
2008	2212
2009	4424
2012	8848
2015	17696
2018	35391
2021	70782

There are some minor evolutionary progress in on-chip interconnection that has been developed from traditional bus architectures. These methods enable minor improvements over earlier approaches and are suitable for the majority of traditional SoC designs. Hence to meet the developing calculation concentrated applications in purchaser hardware and the necessities of low-control, elite frameworks, the quantity of registering assets in single-chip has tremendously expanded. Set-top boxes, remote base stations, superior quality TV, and portable handsets are only a couple of utilizations that have emerged due to multiprocessor SoCs. These sorts of utilizations need many processing assets, for example, DSP, CPU and explicit IPs for fabricating framework in SoC. Thus, the interconnection between one another turns into a testing issue [2, 3]. Many of the SoC's existing core interconnection are point to point and Shared bus based [6]. In point to point connections all the IP cores are connected with wires. A shared bus interconnection can be utilized which requires intervention rationale (arbiter) for serializing communication while deal with SoC. With the shared bus architecture, the communication has effectively been executed in for all IP cores in almost all complex SoC. The shared bus is a minimal effort arrangement and straight forward control attributes. In this approach, single IP core at any given moment can use the bus architecture. In such a situation where required transfer speed of interconnection is larger than accessible transmission capacity, the shared bus interconnection has to be neglected and other interconnections has to be considered. Using a bus - based interconnection can be difficult to meet the needs of communication for a larger number of cores with different requirements of communication [4,5]. The approaches point to point and shared bus can act as efficient infrastructure if the number of cores are lesser. This communication infrastructure suffered from lot of disadvantage like scalability, less adaptive, underutilization of resources [4-6]. Therefore, it motivated the researchers for consideration of computer network arrangement [5,6]. Therefore the researcher's began using the guideline and practices of a computer networks in SoC configuration to defeat the constraint of correspondence and a colossal wiring delay on SoC[6] . This new arrangement thus termed as Network-on-Chip (NoC)[2-6].

1.2 NETWORKS-ON-CHIP

Network-on-Chip [2-6] has been introduced as a promising method to deal with communication problems. It consists of connections of processors, memories and customized designs on a scalable and straightforward architecture platform using hop-by-hop switching packets to provide higher bandwidth and higher performance. NoC based systems is a new data communication strategy for the SoC. The performance of NoC depends on topology, the width of the data link, traffic patterns, routing mechanisms, and arbitration. Figure 1.1 shows the basic block diagram of NoC architecture.

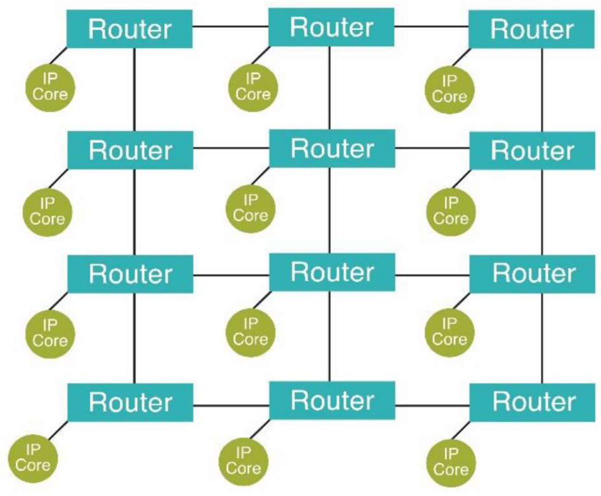


Fig. 1.1: Basic Block diagram of NoC architecture[6]

Network-like communication may be applied for shortened the required wiring by putting predefined number of routers in-between each object of communication. Subsequently, a lot of scalability and selection of complex wiring may be achieved by switch-based interconnection. NoCs replaces the SoC busses that provides high flexibility which also supports more robust circuits. The NoC likewise lessens SoC manufacturing cost, time to volume, time to advertise and configuration challenge. The NoC approach has an unmistakable preferred standpoint over conventional busses and most remarkable framework. The quality in the middle of conventional busses and NoC has been increased due to the hierarchies of crossbars or multilayered busses Moreover, performance and complexity is still lacking. In table 1.2, we have shown the difference between a traditional bus and Network-on-Chip.

Table 1.2: Difference between Bus and Network-on-Chip [4,5]

Bus	**Network-on-Chip**
A well understood simple protocol is used by the bus.	Applied Lightweight computer network protocols to the IP for communication.
All units share the limited bandwidth attached to the bus.	Total available bandwidth depend on the network size and scale accordingly.
The delay is increases with the number of masters in bus arbiter.	Delay remains constant as the process of arbitration and routing is scattered across the routers.
Arbitration latency is only present in the bus; others remains zero.	A significant amount of latency is also present in internal network contention.

The achievement of the NoC configuration relies upon the examination of the interfaces of preparing components of interconnection fabric and NoC. The built up systems has some drawbacks with respect to slow bus reaction time, power and energy restriction, adaptability issue, and data transfer capacity constraint [3,4]. Bus interconnection made out of an expansive segment in SoC's interface. It can affect the moderate interface time because of the impact of bus sharing. Additionally, the interconnection has a deformity as power utilization scores high for associating all items in the correspondence. Additionally, it is difficult to build quantity of association of IP components unendingly in light of the impediment of data transmission. As an outcome, interconnection prospective is responsible for the execution of the NoC configuration.

In spite of the fact that the network innovation in a computer network is as of now all around created, it is practically difficult to apply to a chip-level intercommunication condition with no alteration or decrease. Therefore, creating fitting network architectures for on-chip communication is a current research which numerous researchers are attempting. For effective NoC architecture, the fundamental usefulness ought to be light-weighted and direct. The actualized part of NoC architecture ought to be qualified for a basic segment building a SoC. Despite the fact that the essential usefulness ought to be straightforward, it likewise fulfills the fundamental prerequisite as a rule communication. Then again, to apply to the smaller and smart future devices, it ought to be low-powered. Therefore, numerous parameters have to be considered, for example, working voltage, clock rate and power controlling techniques.

1.3 THREE-DIMENSIONAL NoC

Two-dimensional Networks-on-Chips (2D NoCs) is introduced to be used in medium-sized multicore SoCs due to their scalability, improved throughput, and low power utilization. However, for large-sized multi-core SoCs only increasing the number of cores over 2D NoC is not a viable solution due to a large number of hops-long interconnections which increases the diameter. This has brought attention to the use of a three - dimensional, very large - scale integration (3D VLSI) design that offers tremendous benefits. Three dimensional integration is a novel design approach that reduces the size of the system and makes heterogeneous IC integration possible. Wirebonding can be used to connect several dies in 3D stacks is the simplest form of 3D integration. This type of integration has been around for a long time and reduces the size of electronic products. This wirebonding has not exhibit significant improvement in interconnect delay , Therefore for connecting the layers, TSV's are used in a cost effective manner[47]. The basic building block diagram of 3D NoC is show in fig 1.2

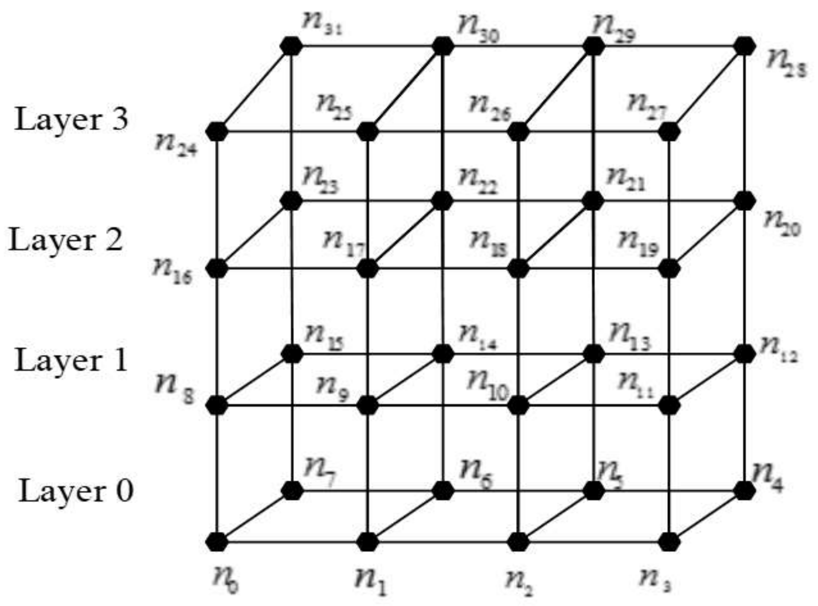


Fig. 1.2: Basic Block diagram of 3D NoC architecture[52]

In recent years, Three-Dimensional Integrated Circuits (3D-ICs) have attracted considerable attention as a possible solution to the interconnected bottleneck. The combination of the NoC structure and the advantages of 3D integration provide us 3D NoC as a new architecture. This architecture meets the scaling requirements for future SoC, taking advantage of the short vertical links between the adjacent layers that can undoubtedly improve system performance, scalability and heterogeneous integration [55].

1.4 PROBLEM STATEMENT

3D NoC architectures require many tradeoffs for meeting specific performance specifications. As its architectures produces their output and energy efficiency against 2D NoC systems, their reliability to sustain their performance growth provides the question mark [7]. Due to the more significant number of cores in 3D NoC which is more prone to congestions, faults and deadlock challenges. Thus utilizing some approaches in the form of routing algorithm which is capable of handling fault, deadlock & congestions are essential for system reliability requirements while keeping the performance parameters optimally.

1.5 OBJECTIVES OF THE WORK

The primary objectives of this work are as follows:

1. To investigate 2D topologies and routing algorithm for NoC regarding network diameter, degree, and routing efficiency.

2. To implement and evaluate 3D NoC routing algorithms concerning the performance parameters like latency, throughput & injection rate.
3. To propose a new low overhead, fault tolerant and congestion-aware routing algorithm for 3D NoC taking consideration of various parameters like throughput, average packet delay, fairness, low latency, optimum injection rate, etc.
4. To implement the newly devised algorithm using simulators like Access Noxim.
5. To evaluate the efficiency of the newly implemented algorithm by comparing it with the performance latest and existing algorithms encountering the similar issues.

1.6 ORGANIZATION OF REPORT

The complete work is presented in this report conveniently over eight chapters.

The first chapter presented the motivation and objectives of the thesis and subsequently presents the scope of work involved in meeting the research objectives.

Chapter two provides the origin of System-on-Chip and Network-on-Chip. Router architecture and switching policies are also presented in this chapter.

Chapter three presents the background on NoC topologies and routing algorithm.

Chapter four presents a literature survey is presented where various 2D NoC and 3D NoC routing algorithms are presented that dealt with deadlock, congestion and fault tolerance. 3D NoC existing architectures are also presented.

Chapter five investigated four main topologies regarding their degree and diameter. Later some common routing algorithms like XY, adaptive and odd-even algorithms are empirically investigated with the help of Booksim 2.0 simulator. The performance parameters like latency and throughput are evaluated concerning injection rate.

Chapter six investigated the various routing algorithm like XYZ, adaptive and turn model for 3D NoC. All the routing algorithms are implemented in Access Noxim simulator. Similar to chapter five, the performance parameters like latency and throughput are evaluated for 3D NoC.

Chapter seven proposed new turn model and TFRF routing algorithm. The turn model and TFRF routing algorithm are implemented with the help of Access Noxim simulator. The performance parameters are calculated and compared with existing three routing algorithms.

Chapter eight presents the conclusion and future work followed by references.

CHAPTER 2. ON-CHIP NETWORKS

2.1 OVERVIEW OF SYSTEM-ON-CHIP

Over the past decade, the significant advancement in manufacturing technologies and in semiconductor design influences the emerging technique of System-on-Chip (SoC). The consistent increase in the transistor count due to Moore's law has extended the capabilities and the levels of integration on ICs. It is expected that more than a billion transistors will be integrated on a single chip. Earlier printed circuit board is used to contain all the functionalities of an embedded device. Now the components like memory, controller, graphics & interface and many other required components can be embedded onto single IC which has obviously increased the chip complexity resulting in the SoC era. A SoC can be composed of computational Intellectual Property (IP) cores, various analog components and integrated circuits that will be implemented as a system on a single chip. Due to this shrinking feature size and current VLSI technology and to meet the growing computation-intensive applications along with the needs of low-power, high-performance systems, the number of computing resource (IP cores) is enormously increased on a single chip. These fundamental units are referred to as Processing Elements (PE) or cores. In [8], Tilera processor was introduced that has a switch with each core arranged in mesh architecture to transmit the data. Thus SoCs with hundreds of IP cores can be imagined. SoC gives an approach to diminish plan exertion and expenses by consolidating various capacities that has been previously exists. SoCs alludes some electronic framework into a solitary IC (chip) or coordinate all parts of a PC [2].

An average SoC comprises of a microprocessor, microcontroller and memory squares for example "USB, Firewire, Ethernet, USART, SPI, Analog Interfaces including ADCs and DACs, voltage controllers and power the executives circuits". Moreover, every one of these segments may incorporate diverse determinations, for example, data transmission capacity, transports, and distinctive open conventions or protocols. Embedded systems is one of its application. The communication architecture is of utmost importance to the future of SoCs. Therefore, the design of SoC can be viewed as communication-based design rather as computation based design. A shared bus and point to point interconnection in most of the System-on-Chip applications are adapted to communicate with each integrated processing unit which uses some adjudication logic to serialize bus access request. The shared bus and point to point communication architecture are presented in figure 2.1 and 2.2 respectively. The bus exhibits simple control characteristics and low-cost solution. It has been successfully exceuted in virtually all complex System-on-Chip designs. These type of interconnections has some weakness where bus requesters are large, considers more number of processing cores and also bandwidth required is more than the available. The bus itself is transmuted into a bottleneck by increasing the number of processing elements. In the above scenario, some other intercommunication methodology needs to be adopted. Another dominant reason to think for an alternative solution is that of complex long wires are being used in bus and crossbar based structures because of which the power consumption got increased on a single chip. Therefore because of the above-stated reasons, the principles and practices of already matured computer network have been considered to implement for SoCs. The thought of VLSI designers has been completely changes by Prof. Prof. William J Dally in 2001[6] that states to route the packets, not wires [6].

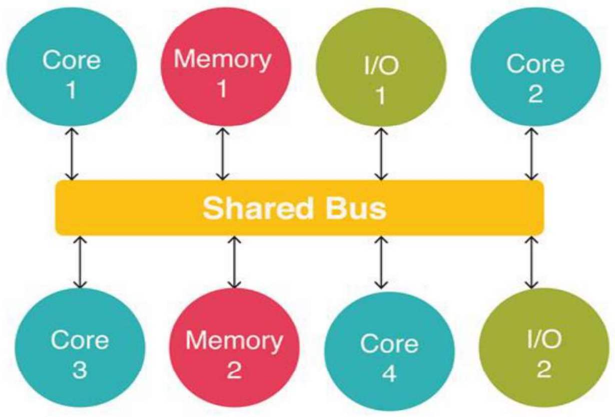


Fig. 2.1: Shared bus interconnects [4]

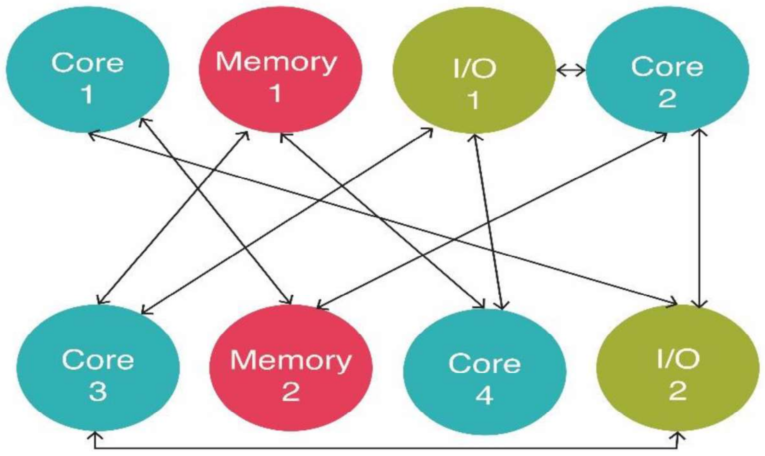


Fig. 2.2: Point-to-Point interconnects[4]

It is not practical to utilize a solitary shared bus, crossbar and point to point interconnects because of their poor versatility with framework measure for intercommunication necessities of SoCs made of many IP cores. To beat these issues of adaptability and multifaceted nature, on-chip packet switched smaller scale network of interconnects, called Network-on-Chip (NoC) architecture has been proposed [2]. It is used to design the “communication subsystem between IP cores in a SoC”. It can be constructed by multiple points to point data links, where each IP core is attached with a router and the messages can be relayed from one IP core to another over the links involved and by using the routing strategy implemented through routers. We have already presented the comparison between bus and NoC in chapter one.

2.2 NETWORK-ON-CHIP: ORIGIN

NoC research began in 1999; its main focus is to utilize the network technology to chip design which solves communication bottleneck problem and global clock problem. NoC has a huge potential, and scalable interconnect to handle the future requirements of multicore SoCs. In NoC, IP cores are interconnected on a single chip similar to the architecture of computer networks. The packet switching communication methodology is used that routes information between network components (PEs, memory, IP cores, I/O resources, etc.) Each IP block has attached router that facilitates the flow of data and routing strategy so that messages can be transmitted from source to any destination. All the IP cores and attached routers are connected through physical links. These physical links are copper wires through which the data is transmitted between IP cores. There is some input and output ports available with a router which are connected to the IP block and other routers in the NoC system. Thus these routers and physical links provide overall communication infrastructure of the NoC system. The packet switching communication approach provides high bandwidth and supports asynchronous data transfer. It transmits packets instead of words. The address of the destination is mentioned in the packet header, so there is no need of dedicated address lines. As compared to a bus-based system, the transmission of packets is evenly distributed among all the routers and can be conducted in parallel where at one time only one sender can transmit the message. Therefore NoC based system is an emerging paradigm for communication within large VLSI/Embedded Systems that facilitates scalability, reusability of IP cores and achieves parallelism [2]. NoC architectures represent the viable solution to meet performance, power and reliability requirements [3]. NoC basic architecture includes a resource node, communication node, topology, and links. The examples of innovative NoC architectures are Stanford/Bologna universities Netchip [2], Lip6 Spin [3], M.I.T. Raw [9], Nostrum [10], Philips’s Ethereal NoC [11].

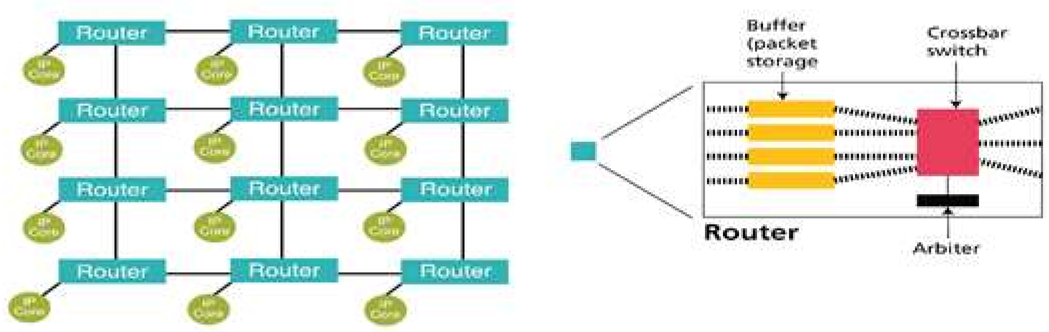


Fig. 2.3: Node model for NoC[4,6]

Figure 2.3 provides the node model of any NoC topology. Every node has associated network interface and router that connects the PE/IP (Processing Element/IP core) to the network. Routers have basic elements like buffers, a crossbar switch, and arbiter. The PE core performs as a packet source and destination (sink). The initiation of the the first (head) flit of a packet may be initiated by using the routing mechanism. Once the router of a node has processed the head flit, a switching mechanism commonly wormhole strategy is used. This mechanism forward the following packet-flits to the buffers of outgoing links of the target path to the destinations.

The main components of NoC are:

- **PE or IP core:** IP cores or Processing Element (PE) is a reusable resource which can be the intellectual property of the third party. The components like "graphics controller or general purpose processors, amplifiers, FPGAs, ADC, DSP, memory, RF unit, application specific hardware component, I/O controller, ARM bus protocols like AHB, APB, the designs like Ethernet, SPI, USB, UART core etc". can be designed as IP core and can be part of various SoC.
- **Router:** Routers are used for routing packets to the destination based on the routing protocols used. Each router can be connected to multiple cores and neighboring routers by the NoC topology.
- **Network Interface:** Network Interfaces (NI) act as the interface between the IP core and the routers. It gives a range of "communication services, such as decoding and mapping addresses, packetization and de-packetization, some prioritized services".

Generally, ICs have been structured with devoted point-to-point associations, with one wire committed to each signal. For expansive and larger outlines of chip, specifically, this has a few constraints from a physical structure perspective. The wires involve a significant part of the region of the chip, and in nanometer CMOS innovation, interconnect rule impacts both execution and dynamic power dispersal, as signal proliferation in wires over the chip requires various clock cycles. Be that as it may, the wires in the connections of the NoC are shared by numerous signals. A notable state of parallelism is accomplished on the grounds as all the connections in the NoC can work

simultaneously on various information bundles. The highlights which are appeared in circular shapes in figure 2.3 represents IP core that speak to NoC switches/routers. For straightforwardness, a mesh network is utilized as topology for interconnection among Processing Elements (PE).
The complexity of designing wires may be reduced by NoC links. The main difference between busses and NoC on "Concerning power dissipation" is that the NoC utilizes point-to-point joins while transports fan out their wires to every one of the objectives [5]. In the NoC case, appropriate floor arranging of the changes prompts a littler wire length, and related capacitance load exchanged per exchange than for the transport, bringing about lower dynamic power. NoC overcomes the problems of high manufacturing cost, scalability and system complexity prevailing in SoC and reduces time to volume, increases performance and reduces time to market and design risk.
NoCs are either broadly useful or application-explicit. The real distinction between Application-Specific Networks-on-Chips (ASNoCs) and broadly useful ones are: first, the correspondence examples of ASNoCs may be statically broke down and, in this way, they should be redone dependent on the application conduct. Second, plan targets for ASNoCs differ from those of broadly useful NoCs. Besides, structure prerequisites differ from the application space to another. Like, interactive media applications have high transmission capacity prerequisite, ongoing frameworks require an ensured postponement, and convenient gadgets ought to have low power utilization. In like manner, for ASNoCs, the style of the on-chip systems ought to be redone to consent to relevant prerequisites. However, the scope of this thesis is only general-purpose NoC's. The NoC-based SoCs have several design issues such as which topology is reasonable for the NoCs with the end goal so that the performance requirements and design obstacles can be tackled. Secondly how the interconnection mechanism between Network Interface, IP core and router will work to exchange the data. Thirdly communication strategy including switching logic, routing, flow control, etc. will be implemented. Finally, as the technology scale and the speed of switching increases, future NoCs becomes more sensitive and prone to errors and failures. Fault tolerance for on-chip communications is becoming essential.

2.3 NETWORK-ON-CHIP PROTOCOL

NoC is packet-switched networks based on the domain of parallel computing and based on set of protocols similar to the ISO/OSI model [] adopted for computer network [126]. The model is presented in figure 2.4. Similar to computer networks[129] the resource sharing among IP cores and the efficient communications is the the primary purpose of NoC. A layered-stack approach provides a set of protocol for the unambiguous on-chip inter-core communications and data transfers. As the concept of networking technology is exploited and used for the NoC, the global on-chip communication paradigm is organized into layers similar to the ISO-OSI reference model. The stacked layers describing protocol allows various services, giving the programmer an abstraction of the communication framework. Layers interact via well-defined interfaces and hide the details of low levels.

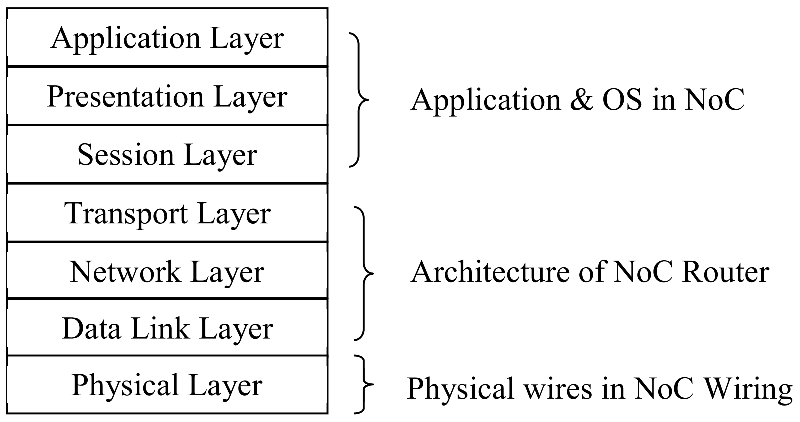


Fig. 2.4: OSI and NoC model [5]

2.3.1 PHYSICAL LAYER

The basic layer which is responsible for communications is physical layer. The physical layer is in charge of the physical dimension of the communication on wires. Its presence in Network-on-Chip alludes to such worries of electric detail of wires, flag voltages, circuits, timing, bus widths, drivers, repeaters and heartbeat state of the flag. The important elements in this layer are length of wire and directing of wires. It can dramatically affect wire length, control utilization, and inactivity. The synchronization of signs at the diverse dimension of voltage level and recurrence (distinctive timing areas) is a test at this layer.

2.3.2 DATA LINK LAYER

This is an important layer in the OSI model as well as in Network-on-Chip. This layer guarantees a dependable data exchange in spite of the physical trickiness and manages medium access control for sharing a standard channel asset, with dispute based access. This layer additionally incorporates detection of error and correction for the transmission of safe information. The layer is likewise in charge of intervention of access to a common physical medium. Examples of Medium Access Control (MAC) protocols are token ring and Time Division Multiple Access (TDMA), Dynamic Time Division Multiple Access (DTDMA). “In data link layer, the predictability, throughput and power consumption may vary significantly depending on which arbitration scheme is adopted” [12].

2.3.3 NETWORK LAYER

This layer is independent of physical network topology. This provides an end-to-end communication to the upper-level protocol layers. "In the network layer, the connections established in the network could be static, such as offered by the reconfigurable interconnect of FPGAs, or dynamic. Similarly, data routes can be persistent over multiple transactions, or each transaction can be dynamically routed. In the latter case, congestion control may be required to reduce traffic through overburdened links" [12].

2.3.4 TRANSPORT LAYER

The transport layer establish and maintain of end-to-end connections. It also performs packet segmentation, manages flow control, and ensure ordering of message ordering. This layer creates an abstraction that hides the details of network topology.

2.3.5 SESSION LAYER

This layer is responsible for synchronization of messages. Routers create these messages in the Network-on-Chip. In other words, we can say this maintains the state of the connection provided by the transport layer.

2.3.6 PRESENTATION LAYER

The presentation layer isn't straightforwardly identified with the Network-on-Chip. It might be present in the application that is executing on the highest point of preparing components. Conventions on this dimension are in charge of changing over the information into a satisfactory arrangement. For instance, two framework parts may trade messages with various byte orderings, so this layer changes over them to a similiar organization [12].

The discussion has been done above on similarity between computer network protocols and NoC. Still, it is evident that the micro-network on single chip will be different as compared to LAN's and WAN's under many different aspects.

- **Placement of IP Cores:** The distance between NoC components on same microchip is very small, and the links and uniformity of the channels provide reliable communication.
- **The Certainty of on-Chip Traffic:** NoC system does know the traffic they need to support at the design time . It is majorly dependent on the application that is currently running on NoC. Hence the traffic pattern is quite unpredictable and makes communication process a challenge .

- **Power & Energy Limitations:** NoC systems are generally made for AC or battery operated smart hand handled devices like phone or multimedia. Thus optimization of power and energy is necessary.
- **Low-Cost and Low-Complexity Requirements:** NoC systems are used in commercial-oriented electronic devices or embedded devices. In the development of these systems, the main aim is to reduce all kind of cost.
- **Reusability:** To connect heterogeneous components which execute various tasks or algorithms by providing standard network interfaces, the same network can be used for all the tasks. The standard interfaces used for NoC systems are "Open Core Protocol (OCP) [13], Virtual Component Interface (VCI) used in the Spin and Proteo [14], NoCs Advanced Extensible Interface (AXI) [15] and Device Transaction Level (DTL) [16]".

2.4 ROUTER ARCHITECTURE

The routers are the heart of NoC whose primary task is to advances information packets to their goals, as per their destination addresses. The work a router does is called steering, which is to some degree like exchanging, however a router is not the same as a switch [17]. The switch is just a gadget to interface the machine to frame a LAN. Usually, data packets are moved through routers when they transmit over a network in their journey, i.e., reaching from source machine to the destination machine. The implementation of the router is based on three communication layers of the OSI model, i.e., physical, data link and network[130].

The router works with IP packets, implying that it works at the dimension of the IP convention. Every router keeps data about its neighbors (different routers). This data incorporates the IP address and the expense with respect to time and deferral. At the point when a parcel of information lands at a router, its header data is investigated by the router. In view of the goal and source, IP locations of the bundle, the router chooses which neighbor it will advance to. It picks the route with the least expense and advances the data packet to the principal router on that route .The router's functionality is to route packets with minimal latency from source to destination. To reduce latency and to improve throughput, routers frequently use pipeline approach. The router delay within the on-chip network affects the latency in communication. This motivates the researchers to reduce the router pipeline phase delay and improve throughput [18]. A designer should consider the simplicity of a router design wisely so that these overheads like costs, space and complexity can be avoided.

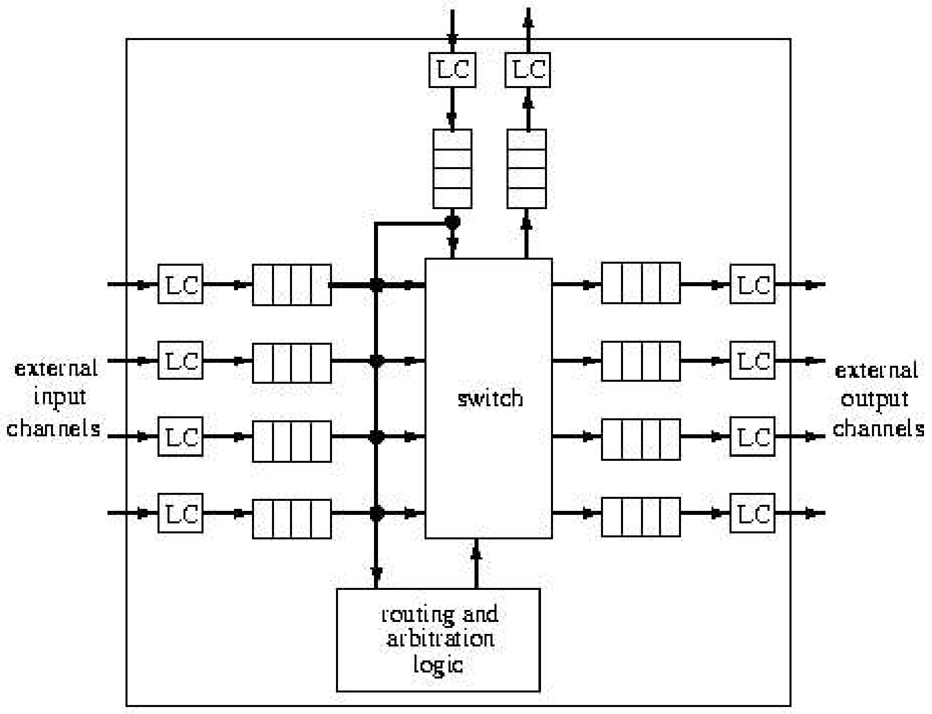


Fig. 2.5: Generic Architecture: 1-port router with input and output buffering [17]

Shown in figure 2.5, is a one port router. It has "switches, buffers and routing and arbitration unit, Virtual Channel (VC) allocator, crossbar, de-multiplexer, and FIFO or virtual channels". The input data (packet) is stored in the input buffers selected by the VC allocator. The arbiter determines which output port the input data is to be routed. The crossbar receives data from input buffers and transfers it to the output port assigned by the arbiter. The crossbar is usually implemented as a crossbar matrix. These components are part of the data path and control path [6]. The datapath is responsible for storing and moving a packet and includes an input buffer set, a switch and an output buffer set whereas the control path coordinates the movement of the packets using the data path resources. Two kinds of router models regular and irregular can be implemented for NoC system. The brief details of the components are as follows:

Routing and Arbitration Unit: It implement the directing calculation and packet stream control convention and direct the switch as needs be. For instance, settlement of clashes between synchronous solicitations for a similar yield connect.

Routing Algorithm: It find out how the packets will move to the destination. It must choose within inside of each middle of the road router which yield channel(s) are to be chosen for approaching packets.

Switch: It interconnects input supports to yield cushions (channels). The implementation can be through crossbar giving full network or less expensive alternate giving fractional availability.

Switching Mechanism: Find assignment of network assets for information transmission, i.e., to find out how and when the routing algorithm choose the info channels which are associated with the yield channel. It provides real system which expels information from information channels and provide to yield channels.

Channel: It has link controllers, buffers and communication medium (e.g., coax cable).

External Channels: It provide the interconnection between routers and characterizes the topology of an associated direct interconnection network.

Link Controller: The physical channel flow control communication protocol is implemented by the link controller between neighboring routers (such as handshaking).

Flow Control: It provides the synchronization convention among sender and recipient nodes which decides moves that has to be made if there should be an occurrence of full cushions, busy yield joins, deficiencies, stops and the sky is the limit from there. Stream control has somewhere around two dimensions:

1) **Packet Flow Control:** It deals with synchronizing sender and recipient at the dimension of packets, guaranteeing accessibility of buffer space and fruitful exchange at the collector. Stream manages the allotment of channels and supports to a packet to which direction it has to go.
2) **Physical Channel Flow Control:** Implements the multi-cycle packet stream control. It breaks the packet into bounces. Be that as it may, even bounces may take a few cycles to exchange; subsequently the most rudimentary unit of data is phit.

Buffer: FIFO memory is utilized for putting away one or a few units of communication in travel. It is useful to put away exchanged information till the following channel has been saved and may overwhelm.

The implementation of the router can be stated as follows:

1) **Input and output buffering:** single buffer is associated with input and output channels
2) **Input buffering :** Buffers are associated with input channels only
3) **Output buffering :** Buffers are associated with output channel only

2.4.1 BUFFER REQUIREMENT IN NOC

There are three fundamental conditions where buffering is vital:

1) The next phase of the network obstructs the yield port through which the packet should be routed,
2) Two packets bound for a similar yield port arrive all the while at various info ports, yet the yield port can acknowledge just a single packet at any given moment, and
3) The packet should be held while the router in the switch decides the yield port through which the packet is to be directed.

The need to store entire data packet inside a router is important factor which is creating make in developing low area, minimal and fast routers. In an execution, wormhole exchanging is utilized which is used in practically all of NoCs. In wormhole exchanging, message bundles are additionally pipelined through the system. A message packet is separated into flits as the flits is the unit of message stream control. Along these lines, the buffers in the switch are enough sufficient to store the number of flits .

Other exchanging methods are not typical in NoCs utility. For example, circuit changing system because of its low execution negates with power and execution parameters. So also, packet exchanging because of its huge buffer necessity demonstrates a similar inconsistency [4].

2.5 SWITCHING TECHNIQUES

Switching system is responsible for assignment of network resources i.e., it provides every detail of about the connection of input channel to output channel. Switching techniques are used for transmitting data across networks. A switch is a gadget that channels approaching information from any of different info ports to the particular output port that will take the information toward its proposed destination node. A switch interconnects input buffers to the output buffers (channels). Three typical switching techniques available for digital traffic; circuit switching, packet switching, and message switching.

2.5.1 Circuit Switching

Circuit exchanging is a method that specifically associates the receiver and the sender in a whole and unbreakable path. With this sort of exchanging procedure, when an association is built up, a devoted way exists between the two nodes until the association is ended. Before transmission of information, the route from the source to the destination is saved by infusing steering test. It includes controlling data and destination address. Phone exchanging gear, for instance, builds up a way that interfaces the guest's phone to the collector's phone by making a physical association. It usually needs more than one phit. This type of switching is only advantageous in the sense that communication channel (once established) is dedicated. However, it has drawbacks resulting in a possibly long wait to establish a connection during which no data can be transmitted. It is more costly than some other exchanging strategies in light of the fact that a devoted way is required for every association. It likewise has a wasteful utilization of the correspondence channel, on the grounds that the channel isn't utilized when the associated frameworks are not utilizing it.

2.6 PACKET SWITCHING

It is a convention in which messages are separated into parcels before they are sent. Every bundle is then transmitted independently and can even pursue distinctive courses to its goal. When every one of the bundles shaping a message touch base at the goal, they are recompiled into the first message. Bundle exchanging is an advanced system specialized technique that bunches every transmitted datum into appropriately estimated squares, called parcels. It very well may be viewed as an answer that attempts to join the benefits of the message and circuit changing and to limit the burdens of both. There are two strategies for parcel exchanging: datagram and virtual circuit. Bundle exchanging is financially savvy since exchanging gadgets needn't bother with a huge measure of optional stockpiling. Parcel exchanging offers enhanced defer attributes; in light of the fact that there are no long messages in the line (most extreme bundle measure is settled). Be that as it may, conventions for parcel exchanging are regularly progressively mind boggling. It can include some underlying expenses in usage. In the event that a parcel is lost, the sender needs to retransmit the information. Other hindrance is that bundle exchanged frameworks still can't convey indistinguishable quality from committed circuits in requests needful insignificant deferral - like voice discussions or moving pictures.

2.6.1 WORMHOLE SWITCHING

In wormhole switching, unlike packet switching, the large network packet is broken into small pieces called flits (flow control digits). Since the packet is communicated flit by flit, it might inhabit numerous flit buffers along its path, making a worm-like an image. Instead of whole packet, the router has buffers which has been provided to one or a few flits. The advantages of wormhole switching are that it provides simplicity, low cost, increased throughput and distance insensitivity, reduces latency [19]. In this bandwidth and channel, allocations are decoupled, and it makes efficient use of buffers. Communication units are shown in figure 2.6.

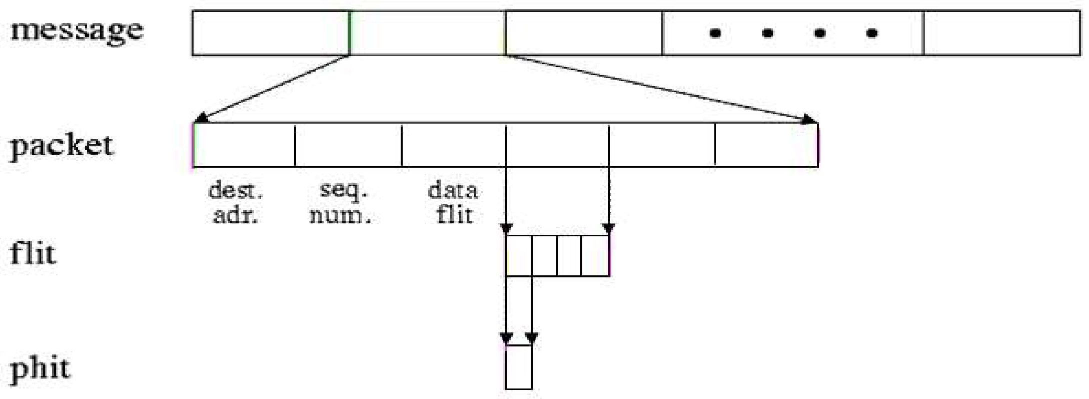


Fig. 2.6: Communication unit [17]

Message: The unit of correspondence from the software engineer's point of view. Its size is restricted just by the client memory space. The message may be broken into one or more packets for transmission.

Packet: Fixed-size littlest unit of a correspondence containing directing data (e.g., a goal address) and sequencing data in its header. Its size is of request hundreds or thousands of bytes or words. It comprises of header bounces and information flutters.

Flit: Small unit of data at the connection layer, or the measure of one or a few words. Flutters can be of a few kinds, and dance trade convention normally requires a few cycles.

Phit: The littlest physical unit of data at the physical layer, which is exchanged crosswise over one physical connection in one cycle. The main dance, called the header flutter, holds the data about this current bundle's course (goal address) and sets up the steering conduct for every single ensuing dance. The head bounce is trailed by at least zero body flutter containing the real payload of information. The last flutter, called the tail dance, plays out some accounting to close the association between the two nodes.

As we stated, in this exchanging, every packet is partitioned into equivalent littler areas named as bounce [17]. Dances are simultaneously moved in the network. In this manner if assume, 16-bit dances are prepared to be exchanged, 32-signals between two routers are considered to exchange the flutters, 16-signals for sending and 16-signals for accepting. Along these lines, flutters are moved in parallel.

As we stated, in this exchanging, every bundle is separated into equivalent littler segments named as flutter [17]. Flits are simultaneously moved in the system. In this manner if assume, 16-bit dances are prepared to be exchanged, 32-signals between two switches are considered to exchange the dances, 16-signals for sending and 16-signals for getting. Thusly, flutters are moved in parallel. In wormhole switching, the address is so short that it can be translated before the message itself arrives. Wormhole switching does not dictate the route to the destination but decides when the packet moves forward from a router. This type of switching allocates buffers and channel bandwidth on the flit level instead of packet level. Thus it reduces the amount of buffering required on each node making it possible to build fast, inexpensive routers.

2.6.2 DEADLOCK PROBLEM IN WORMHOLE SWITCHING

The wormhole switching has many advantages, but it is very deadlock prone. One disadvantage of this straightforward wormhole exchanging is the powerlessness of interleaving or multiplexing clear messages over a physical channel. Be that as it may, by applying virtual channels, such channel use can be expanded [9,20]. The header bounce fabricates a way in the system, which alternate dances follow in the pipeline. The arrangement of supports and connections involved by bounces of a given parcel shapes the wormhole. Nonetheless, the length of the way here is corresponding to the quantity of dances in the parcel. Commonly, it traverses the entire way between the source and goal. On the off chance that the header can't continue because of occupied yield channels, the entire chain of bounces gets slowed down, possessing dance cradles in switches on the way built up until now and blocking other conceivable correspondences. Blocking assets in the event of slowed down pipelines is the primary downside of this procedure. We can't blend bounces of various bundles into one cushion (one physical channel) since flutters convey no character banners. With including some control rationale, we can part one physical channel into a few virtual channels. They will have their cradles yet will share one single physical channel.

2.7 NETWORK-ON-CHIP: OPERATION

The operation of a network is divided into cycles, and the same events occur during each cycle. Flit is allowed to move through routers each cycle if it is not blocked. This allows an entire packet to be transferred quickly if there is no contention for ports. Each cycle has three major events: packet generation, routing, and wire transfer. At the beginning of each cycle, the routers check for new packets generated by an IP core. If a new packet is in the queue of the component, the first several packets are moved to the input queue of the router. A router will use the routing algorithm and route the packets after checking for newly generated packets. First, the priority of the flits residing in the router's input port is checked by the arbiter for routing. Afterward, the correct output port is selected for forwarding the flit using routing algorithm. In the case of adaptive algorithms, the network state is also considered before selecting the appropriate output port. It may take more than one cycle to determine the output port. The first flit of the packet is head flit containing destination addresses; all other flits follow head flit only. The wire transfer forwards flits from the output queue of one router to the input queue of a neighboring router. Finally, when tail flit reaches the destination, it can be assumed that packet reaches the destination.

2.8 QUALITY OF SERVICE: NETWORK-ON-CHIP

The Network-on-Chip (NoC) methodology is gaining attention as a promising alternative approach over traditional bus-based SoCs which may not be able to meet the scalability, reliability, and high throughput requirements for complex System-on-Chip of the future. A NoC breaks the communication bottleneck by applying packet-based switching which is widely used in computer systems and networks. Limiting the system cost (or amplifying system utility) while keeping up the required Quality of Service (QoS) is one of the noteworthy factors in NoC engineering structure. In such manner, we will portray NOC on QoS parameters like Guaranteed Service (GS) and Best Effort (BE) [19].

The Guaranteed Service (GS) is association situated, giving certain limits in inertness and data transmission. An association is a unidirectional virtual circuit set up from a source NI to a goal NI by means of the system. Ensured Service requires reservation of assets reses, for example, supports and connection transfer speed for associations. The NIs oversees associations in regards to the foundation, arrangement and tear down to guarantee information uprightness, lossless and requested information conveyance. GS makes a genuine certification on execution, notwithstanding major assurances of rightness and finishing for correspondence.

The Best-Effort administration (BE) is affiliation less, passing on packages in a best-effort structure. It has no establishment stage, and sources send groups without the cognizance of states in objectives. Best-Effort organization does not require any booking of advantages and no insistence are expected to be given. BE organizations are definitely not hard to use, while GS organizations require vigilant programming to spare the required resources in the framework. Here accuracy and consummation of correspondence are ensured.

CHAPTER 3. TOPOLOGIES AND ROUTING

3.1 EVOLUTION OF ON-CHIP ARCHITECTURES

The most important way to understand the design space of on-chip architectures or computer networks is to understand through topology adopted[128]. Initially for on-chip architectures shared bus were used where all IP cores are interconnected through links. A simple arbiter is used to manage the bus request. The constraints like latency and bandwidth and the compatibility requirement of IP reuse, the necessary bus evolution like a hierarchical bus, bus matrix & crossbar were adopted. These structures have improved that latency and bandwidth requirements but long wires generated severe implications for power requirements. Due to issues above NoC has started displacing these bus-based architectures and will exhibit the economic advantages like reducing SoC manufacturing cost, increasing SoC performance, reducing SoC time to market, reducing SoC time to volume and reducing SoC design time. The evolution of on-chip architectures is shown in figure 3.1.

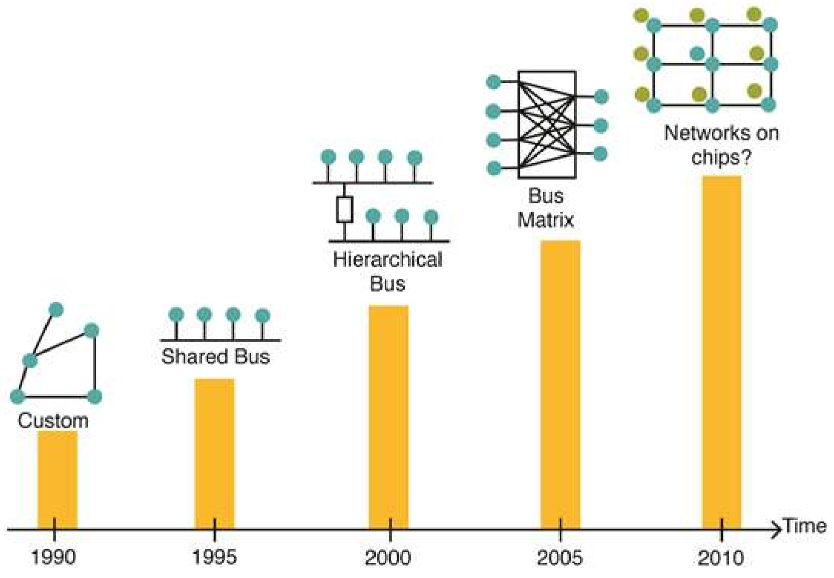


Fig. 3.1: Evolution of on-chip architectures[8]

3.2 TOPOLOGIES

The system topology is characterized as the how different components (Processing Elements, IP's, DSP's, and Processors, and so forth.) are associated physically and intelligently. Physical topology suggests the physical structure of a framework. Astute topology suggests how truly data is being moved in a framework. The NoC is a correspondence driven interconnection approach which gives a versatile structure to interconnect different IPs and sub-systems in a SoC [21]. NoC's can make

SoC's undeniably composed, reusable and can in like manner improve their execution. Since the correspondence between the distinctive taking care of focuses will be the fundamental factor for the execution of such structures, thusly we need to focus on making this correspondence speedier similarly as progressively trustworthy.

Most of interconnected structures utilized for NoC began in the parallel figuring field. In any case, the SoC structure worldview forces certain limitations on these models, as there is a huge distinction between on-chip and off-chip applications (installed). Topology is one of the fundamental parameters for the execution of a NoC.

Following are the factors which should be considered while choosing topology:

- **Performance Requirements:** The NoC architecture should provide important performance regarding throughput, low latency, low power consumption, and low area requirements and complexity of implementation.
- **Scalability:** A scalable architecture means that the I/O bandwidth and network bandwidth should increase proportionally as more functional units are added.
- **Simplicity:** Simple designs often lead to higher frequencies of the clock and increase performance.
- **Reliability and Fault Tolerance:** In the event of a limited number of defects, an interconnection network should be able to provide information reliably and be designed for continuous operation.

3.2.1 CHARACTERISTICS OF NETWORK TOPOLOGY

Topology significantly affects framework execution and dependability. It for the most part impacts network distance across (the length of the greatest briefest way between any two nodes), format and wiring. Before we dive further into the broadly utilized topologies, the fundamental attributes of a network topology which are depicted as pursues ought to be comprehended first:

- **Bisection Width**: A Bisection of a network is a cut that partitions the entire network nearly in half. Bisection width is used to divide the topology into two networks of approximately the same size, i.e., the number of links needed to be removed. The full bisection width provides more paths between two sub-networks, improving overall network performance. It is one of the best parameters for parallel architecture evaluation and comparison of interconnection networks.
- **Throughput:** The throughput of a system is the information rate in bits every second that the system acknowledges per input port. The perfect throughput is characterized as the

throughput expecting an ideal directing and streams control, the i.e., load is adjusted over substitute ways and no inactive cycles on bottleneck channels.

- **Latency or End-to-End Delay:** The latency or end-to-end deferral of the system is the time required for a message to cross a system, i.e., a period is taken for a bundle, flits or message to reach from source to goal. It additionally incorporates the time taken for figuring mediation rationale just as routing path calculation and different deferrals. The average delay is calculated by taking the mean of the end-to-end delay of each successfully sent packet. The lost packets are not taken into account in the average end-to-end time. The average delay from end-to-end reflects how quickly packets can be delivered to their destinations. The smaller this value, the better the network will be.
- **Diameter:** The Diameter (D) of a network is the maximum internodes distance. It is the maximum shortest possible path between all pairs of nodes. If there is no direct link between two nodes, the message must travel through specific intermediate nodes that introduce multiple hop delays. As the message delay is proportional to the hop number, the length of the shortest path becomes an important metric. Low network diameter can also help with low and predictable latency, routing paths, traffic flow. The smaller the network diameter, the less time it takes for a message to be sent to the farthest node from one node.
- **Node Degree:** The Node Degree (ND) is defined as the number of physical channels emanating from a node. This attribute shows the node's I/O complexity. A network is called regular if all nodes are of the same degree, otherwise irregular. For example, ring topology has degree 2, hypercube having $\log_2 N$.
- **Link Complexity:** Link Complexity increases with increasing number of IP core. The addition of additional links to a particular network can reduce its diameter and improve communication between nodes. Increased connection complexity can lead to increased hardware complexity and overhead area
- **Hop Count:** Hop Count can be defined as the number of intermediate nodes a message takes to reach from source to destination. It is one of the essential properties of topology as it has a direct impact on the latency of the network. Even if there is no congestion, every node causes some delay.

We can comprehensively arrange the network topology as pursues:

- Shared medium: bus, token ring
- Direct topologies (interface each change specifically to a node)
- Indirect topologies (probably a portion of the changes interface with different switches)
- Hybrid topologies

We additionally present a portion of the broadly utilized topologies in Network-on-Chip as pursues.

3.2.2 NoC TOPOLOGIES

3.2.2.1 2D MESH TOPOLOGY FAMILY

The switches are orchestrated into a 2D cross section. The communication is permitted just between neighboring switches. Variations permit fold over associations between switches on the edge of mesh (known as TORUS). The figure 3.2 and 3.3 presents mesh and torus topology individually. The assessment parameters of 2D Meshes are as per the following:

- Diameter: Θ(n^(1/2))
- Bisection width: Θ(n^(1/2))
- Number of edges per switch: 4

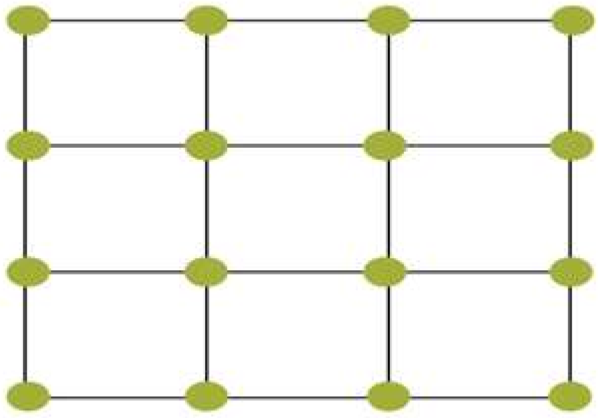


Fig. 3.2: Mesh topology[6]

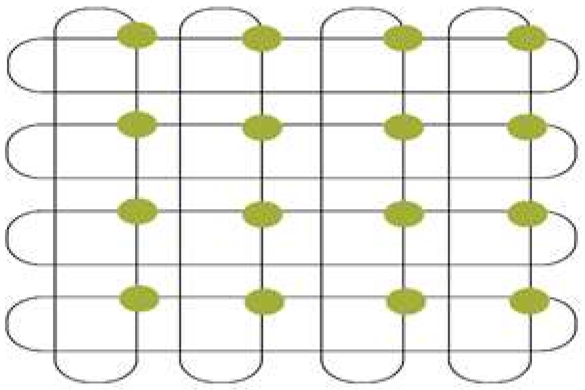


Fig. 3.3: Torus topology[8]

In [22] author proposed m × n mesh interconnecting computational resources (IPs) placed along with the switches, as shown in figure 3.4 called CLICHÉ (Chip-Level Integration of Communicating Heterogeneous Elements). Each switch is connected to four adjacent switches and one IP block, except those on the edges. The number of switches in mesh-based architecture is the same as the number of IPs. IPs and switches are connected via two unidirectional links.

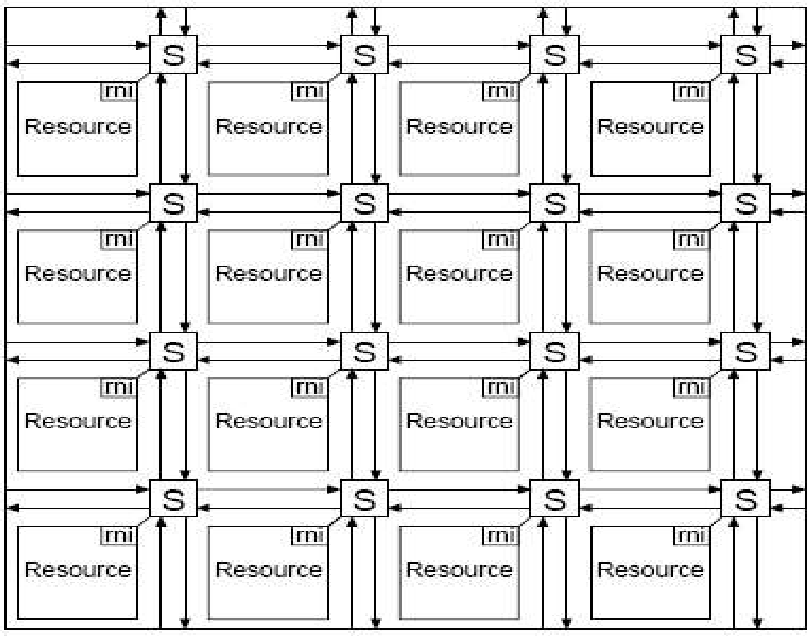


Fig. 3.4: CLICHÉ [22]

3.2.2.2 HYPERCUBE: 2 X2 X… X2 MESH

The amount of hubs n in the hypercube is a force of 2(2^k for k-dimensional structure). For example in a 3D hypercube n = 2^3 = 8. The center point will in general keep running from 0, 1… .(2^k)- 1 for k-dimensional structure. In a 3D hypercube, center point addresses are 0, 1, 2,… .,7. The center point is related with hubs whose addresses shift from it in exactly one piece position. The evaluation parameter of the hypercube organize is according to the following:

- Edges per node: log n
- Bisection width: n/ 2
- Diameter: log n

3.2.2.3 BINARY TREE NETWORK

In a regular network, the processors are associated in a round about way through different routers. In such case the absolute number of nodes, n = 2^d processor nodes where "d" is the tallness of the tree. For instance in a double tree of tallness 3, n = 2^3 = 8 as appeared in figure 3.5. The most extreme number of middle of the road changes so as to impart starting with one hub then onto the next hub will be n-1.

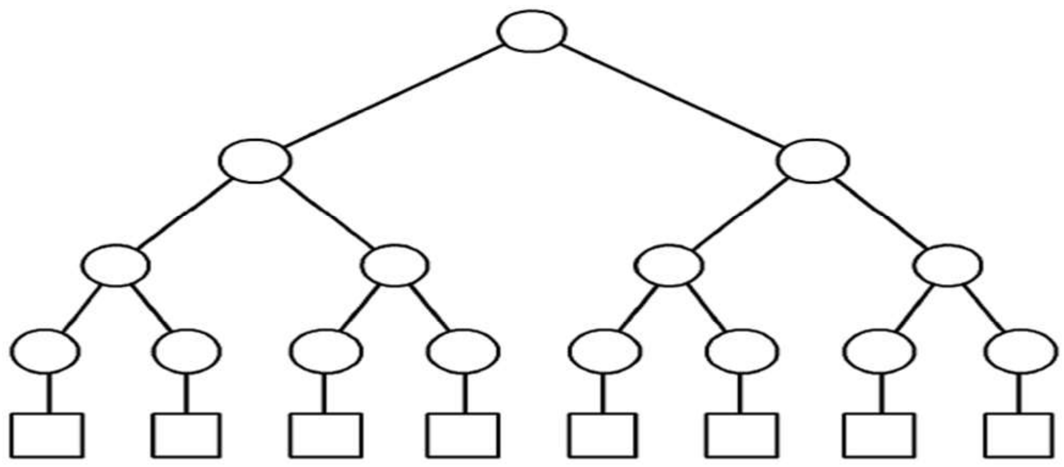


Fig. 3.5: Binary tree network[8]

The evaluating parameters of binary tree network are as follows:

- Bisection width: 1
- Edges / node: 3
- Diameter: 2 $\log_2$ n

3.2.2.4 SPIN ARCHITECTURE

In [23], the author proposed tree based SPIN (Scalable, Programmable, and Integrated Network) architecture. The leaves are IP blocks, and the rest of the layer consists of routers. The network is non-blocking because there are many parents for the leave nodes. It means that the connection between an idle pair of input and output ports that do not affect existing connections can always be established. It is shown in figure 3.6.

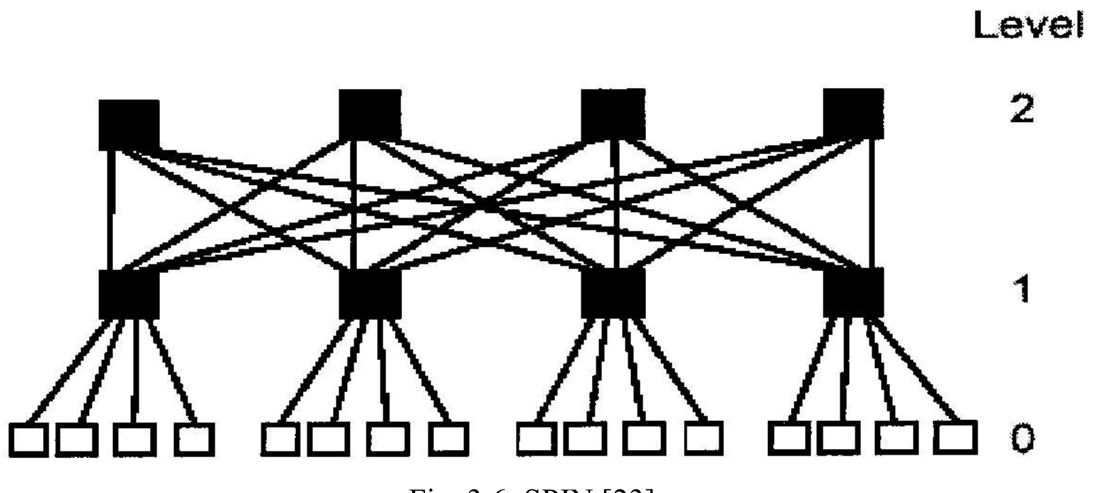


Fig. 3.6: SPIN [23]

3.2.2.5 HYPER-TREE NETWORK

The hyper-tree arrange shares low breadth of the twofold tree. They extraordinarily improve separation width. from "front" resembles a k-ary tree of stature d. From "side" looks like topsy turvy twofold tree of tallness d. We can assess 4-are hyper-tree as follows:

- Edges per node: 6
- Bisection width: n/ 2
- Diameter: log n

3.2.2.6 BUTTERFLY NETWORK

The butterfly network is another indirect topology. The total number of nodes *n*= 2^d processor nodes connected by *n* (log *n*+ 1) switching nodes as shown in figure 3.7.

We can evaluate the butterfly network as follows:

- Edges per node: 4
- Bisection width: n/ 2
- Diameter: log n

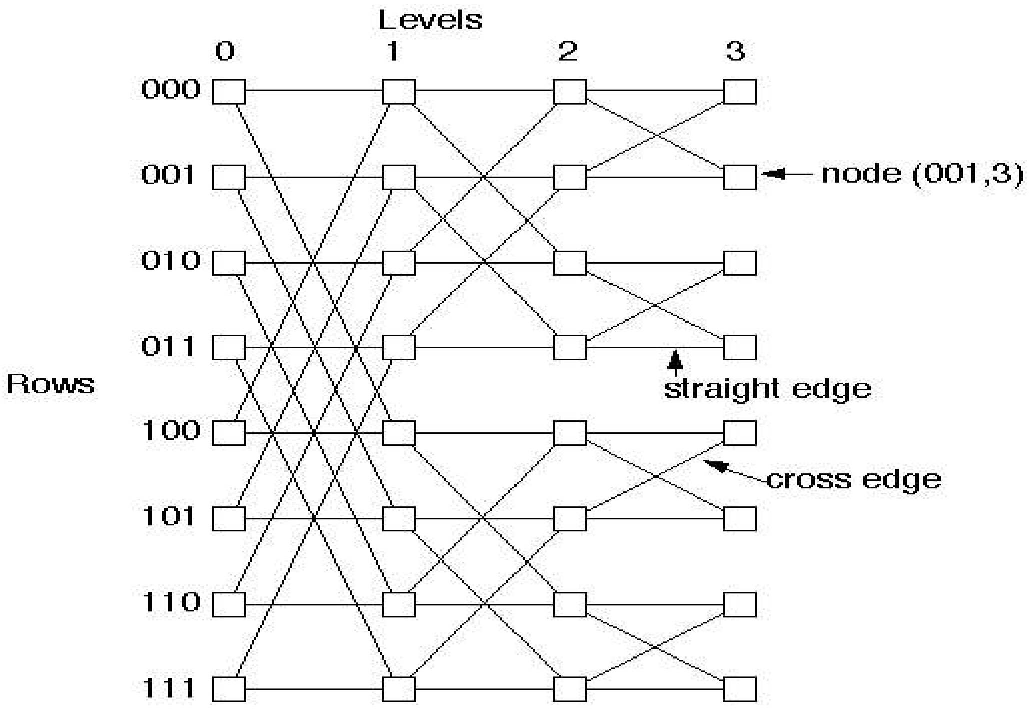


Fig. 3.7: Butterfly network

3.2.2.7 OCTAGON

There are eight nodes and twelve bidirectional links in octagon topology. Each pair of hubs has a most extreme two-jump method for correspondence. On the off chance that a framework including in excess of eight hubs, at that point octagon topology can be reached out to multidimensional space with expanded wiring unpredictability. The design for the most part utilizes a straightforward short way calculation. The throughput is a lot higher than the transport and crossbar. The octagon is shown in figure 3.8.

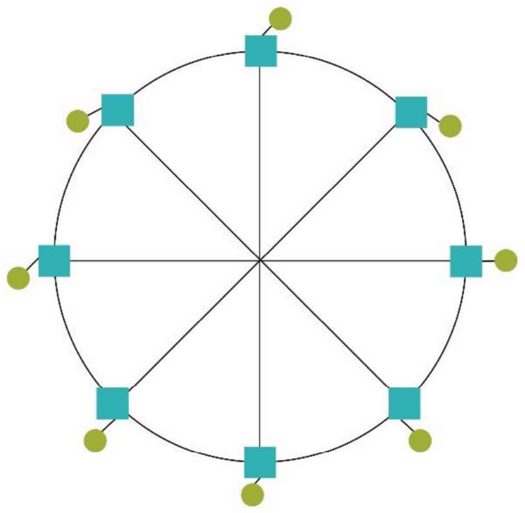


Fig. 3.8: Octagon [24]

3.2.2.8 Star Topology

A star arrange comprises of a huge switch put amidst the star and other PC assets or sub-organizes in the star's spikes. As the traffic between all spikes goes through the focal switch, in this manner the limit of the focal switch is very vast, and there are conceivable odds of clog in the center. The star topology is shown in figure 3.9.

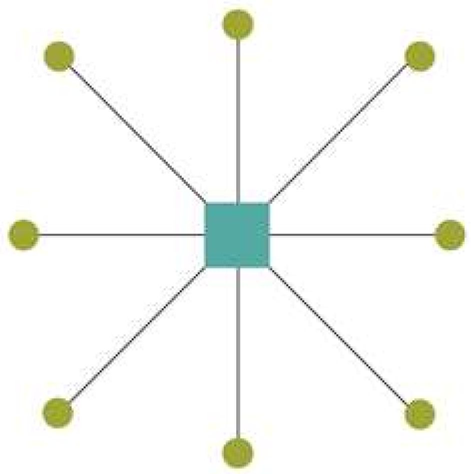


Fig. 3.9: Star topology [24]

3.3 BACKGROUND OF ROUTING ALGORITHM

We should initially decide the topology for the on-chip network and after that an appropriate routing algorithm is selected. Routing algorithms are in charge of effectively and proficiently routing packets from the source to the goal. The essential assignment of a routing algorithm is to disseminate traffic evenly over the network from various nodes. The decision of a routing algorithm relies upon exchange offs between a few possibly clashing measurements, for example, limiting the power required for routing, routing algorithm complexity and size of routing tables on a chip , expanding execution by decreasing deferral and boosting traffic use of the network, and enhancing robustness to more readily adjust to changing traffic needs. A specific routing algorithm influences the multifaceted nature of the router , the onchip area necessities, energy and power utilization and throughput of the network to meet all execution prerequisites. There are specific routing algorithm properties as follows that are essential for interconnecting networks:

- **Connectivity:** Ability to route packets to a destination node from any source node.
- **Adaptivity:** Ability to route packets via alternative paths in the presence of congestion and faults.
- **Deadlock/Livelock:** Ability to ensure that packets never block or wander through the network.
- **Fault Tolerance:** Ability to route packets in the presence of faults.

Many routing algorithms have been proposed in the last decades to avoid these challenges and improve performance and to develop scalability in NoC. There are numerous approaches to characterize steering in on-chip systems. There are diverse sorts of directing calculations that contrast contingent upon their key highlights. The directing calculations are grouped into two sorts, unicast and multicast steering, contingent upon the quantity of goals. By way directing choice, unicast steering can additionally be arranged as source directing, circulated steering, and brought together directing and multiphase directing. In source directing, the steering way is determined by some directing methodology when an information bundle is created at the hub and put away in the parcel header. In dispersing the choice is made at each hub when a bundle or bounces move in the system. Multi-stage directing is the blend of the source and dispersed steering. For brought together directing, a unified controller does all the steering for all hubs.

With respect to, the directing calculations depend on two kinds, i.e., the query table and the FSM. In the query table strategy, a query table is available in each switch and is executed at the product level. The routing algorithm updates the entries in the search table if these entries change, the entire routing algorithm can be changed. Finite state-based routing algorithms are machine based and can be used in both software and hardware. Also, according to the adaptability, distributed routing is also classified into either deterministic (non-adaptive), partial or full-adaptive routing algorithms.

Another routing algorithm includes backtracking routing algorithm which are fault tolerant, progressive routing where the communication channel is reserved for forwarding the flits. Profitable routing always does routing towards the path which is close to the destination node; misrouting algorithm is the reverse of the profitable routing they forward the flit towards the path which is away from the destination and may cause packet loss also.

3.3.1 UNICAST AND MULTICAST ROUTING

Unicast routing algorithms can be operated as one-to-one, in which communication between a single pair of source and destination is established. However, multicast NoC routing algorithms can connect one source to many destinations. Between the unicast and multicast routing policies available, unicast routing is considered as a good methodology for the on-chip network due to its point-to-point communication links between different nodes on a chip system.

3.3.2 STATIC AND DYNAMIC ROUTING

Directing choices in a NoC switch can be either static (likewise called deterministic or neglectful) or dynamic (additionally called versatile). In static routing, settled ways are utilized to exchange information between a specific source and goal. This routing plan does not consider the present condition of the system and is ignorant of the traffic load on the switches and connections while settling on directing choices. One of the numerous preferences of static routing is that it requires minimal logic to implement and almost no extra switch rationale or router configuration is required. Static directing additionally allows data packets to be divided into flits and allows to take different ways between a source and destination, in a foreordained way. In the event that just a solitary way is utilized, static routing ensures all together delivery of information packets. This disposes of the requirement for adding bits to packets at the NI, so as to effectively distinguish and reorder them at the destination. In dynamic routing, directing choices are made by the present condition of the system, considering elements, for example, accessibility and traffic burden on links and connections. All things considered it is conceivable that the way between the source and goal changes after some time, as traffic conditions and necessities of the application change. This versatile conduct, in any case, comes at the expense of extra assets that constantly screen the condition of the system and progressively change directing ways. In any case, dynamic routing can all the more likely appropriate traffic in a system and can use exchange ways when certain headings end up clogged. This likewise permits support for more traffic on a similar NoC topology. Regularly, static routing is utilized for situations where traffic prerequisites are enduring and known early, while dynamic directing is progressively alluring when traffic conditions are increasingly sporadic and eccentric. Instances of static routing algorithm incorporates Dimension Order Routing (DOR), XY[29], pseudo versatile XY[29], surrounding XY[32], Valliant's arbitrary[34], topology versatile, coordinated flood, and irregular walk directing. Instances of dynamic (or versatile) directing calculations incorporate negligible versatile, completely versatile, clog look-ahead[34], 2TURN, odd– even[36,37], slack time mindful, and hot potato steering[34].

3.3.3 DETERMINISTIC AND ADAPTIVE

Distributed routing algorithm can be categorized as deterministic (non-adaptive) or adaptive routing algorithms. Deterministic routing algorithms route a packet via only one route based on its destination information without taking into account the network conditions. In adaptive routing algorithms, on the other hand, the packet can be routed through any path based on network conditions and the routing algorithm used. The routing paths in adaptive routing algorithms are therefore not only determined by the source and destination information, but also by the network condition when the packet is routed. Adaptive routing is capable of preventing congested and faulty paths in the network. The drawbacks of this type algorithm are its complexity, cost, and higher energy consumption. Therefore it is must to consider the right QoS metrics before its implementation. The deterministic routing, however, cannot prevent or change the routing path, even if congested.

3.3.3.1 XY ROUTING

XY routing[27] is one of the most common non-adaptive routing algorithms in NoC. The data packet will first move towards X axis in the same row of the source node and then to Y axis when it reaches the column of the destination.

Figure 3.10 shows how the packet can be routed using the XY routing algorithm between the source node (4,2) and the destination node (1,4) in 5*5 2D mesh NoC.

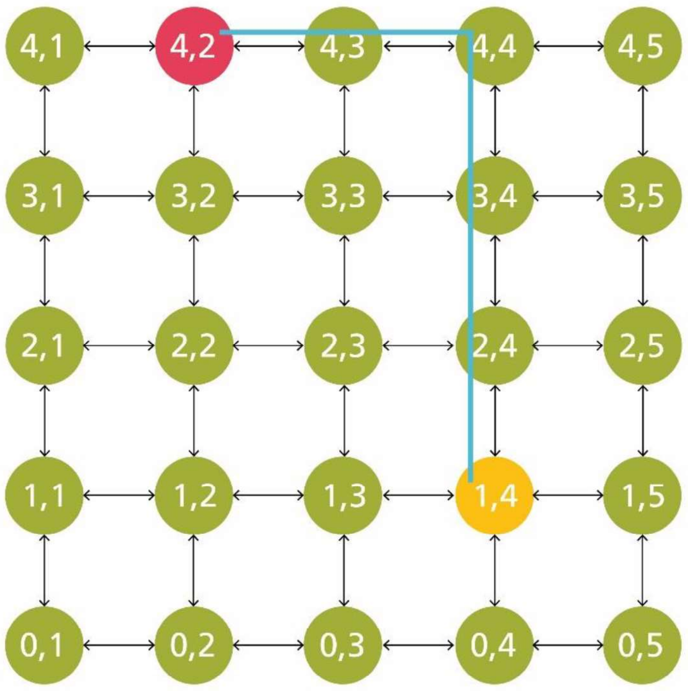


Fig. 3.10: Routing a packet using the XY routing algorithm[27]

First, the packet traverses from the source node (4,2) horizontally into the node (4,3) toward the destination. Then, the routing algorithm will check if the y-coordinate of the current source node (4,3) is the same as the y-coordinate of the destination node (1,4). Because the y-coordinates of both nodes are not the same, the packet will continue moving horizontally to the next node (4,4). Then, the routing will again check the y-coordinates of the new current source node (4,4) and the destination node (1,4). Because the y-coordinates of the current source node and the destination node are identical, the packet will change its movement to moving vertically (without changing its path) until reaching the destination node (1,4). If a node (3,4) or node (2,4) are congested or the link between (4,4) and (1,4) fails, changing the route will not be permitted in this routing algorithm. As a result, the packet will not reach the destination and will be considered as a lost packet which will severely affect the resultant throughput in NoC.

XY algorithm is deadlock-free as it does not allow all possible turns in mesh based network. Figure 3.11 illustrates all possible turns in a 2D mesh network while figure 3.12 presents the more limited set of permissible turns allowed by DOR XY routing.

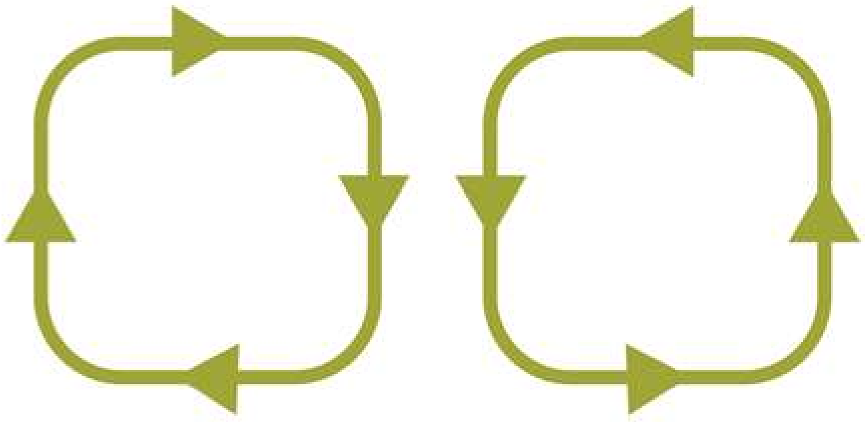


Fig. 3.11: All possible turns

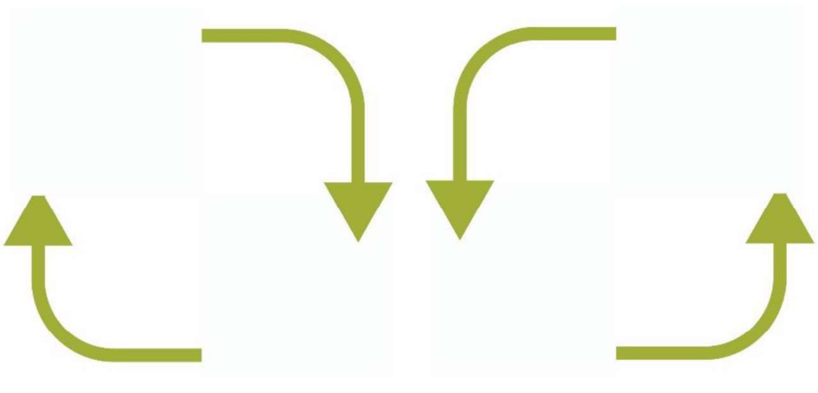


Fig. 3.12: XY turns[27]

If all turns are allowed in the network, then it may form deadlock due to cyclic resource dependencies. Thus to avoid these resource dependencies, few turns should be prohibited. As we can see in figure 3.12, there are no cycles. Specifically, a message voyaging east or west may turn north or south, yet messages are voyaging north, and south are not permitted to turn. As we are not permitting two of the four turns, hence a cycle can't be framed. The XY routing is both straightforward and gridlock free, yet wipes out the assorted variety of ways in a mesh network and along these lines diminishes throughput.

3.3.3.2 OBLIVIOUS ALGORITHM

In oblivious routing algorithm routing ways are resolved without giving any thought to the condition of the network. These routing algorithms can be kept straightforward by not utilizing data on the status of the network. Valiant's randomized routing algorithm is one case of a careless routing algorithm. A middle of the road goal k is haphazardly chosen for routing a packet from sources to goal d utilizing the Valiant algorithm. The packet is first routed from s to k and after that from k to d. By first routing to a haphazardly chosen middle of the road goal before achieving the last goal, the Valiant algorithm can stack traffic through the network. This randomization prompts uniform irregular traffic designs. The quantity of jumps is expanded because of this irregular determination of the middle of the road hub, which thus builds the normal idleness of the packet and the normal vitality devoured by the packet in the network. The Valiant routing can likewise be created to help insignificant courses by confining the decision of routing to the most brief courses so as to safeguard the area.

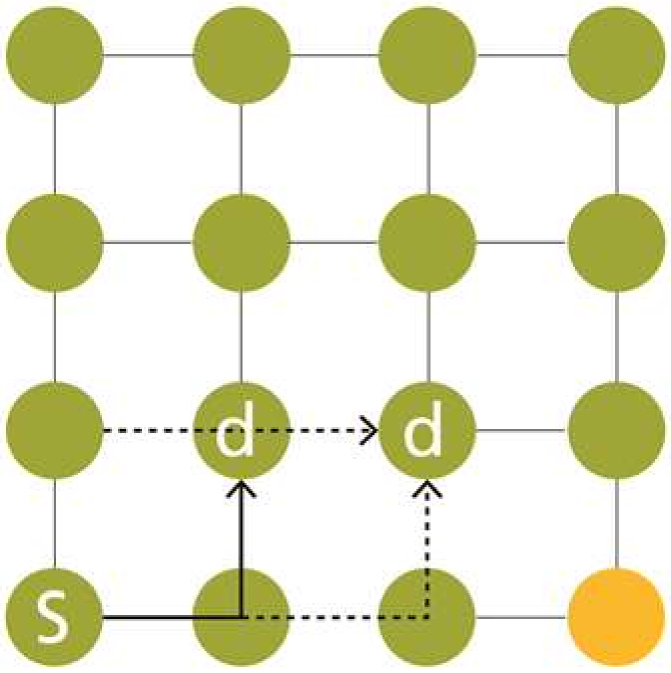


Fig. 3.13: Valiant's routing algorithm

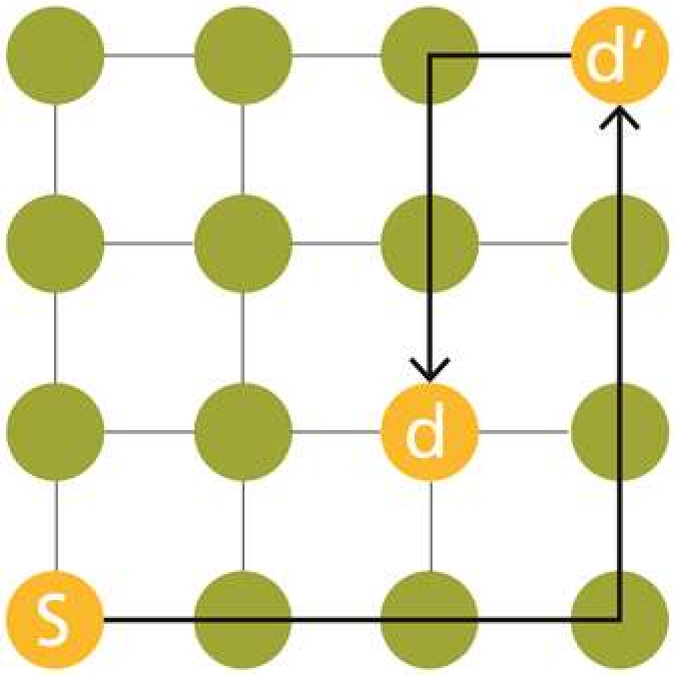


Fig. 3.14: Minimal oblivious routing

Figure 3.13 and 3.14 shows a routing way chosen utilizing Valiant's algorithm and negligible neglectful routing. The algorithm of Valiant arbitrarily picks the transitional goal K. The irregular determination can wreck the area and increment the quantity of bounces altogether; jumps are expanded from three jumps to nine bounces. To protect the area, insignificant unmindful routing can be utilized as appeared in figure 3.13. Presently k must be picked to be inside the base quadrant framed by s and d, saving the bounce consider three. Valiant's routing algorithm and negligible unmindful routing are without gridlock if XY routing will be utilized. It will not be deadlock free if the algorithm chooses randomly between XY and YX routing.

3.3.3.3 ADAPTIVE ALGORITHM

Versatile directing calculations don't constrain a message to a solitary way from the source to the goal. The current network conditions are considered when making a routing decision. Due to this the routing process becomes more flexible and minimizes unnecessary waiting times and improves the tolerance of faults. There are two processes involved in adaptive routing: Directing and Selection. The steering procedure gives a lot of yield channels dependent on the present hub and the objective hub. Determination process chooses the most fitting yield channel from that set. Versatile calculations likewise utilize a backtracking strategy, which enables the bundle header to backtrack, discharge recently held channels and look for the fitting way deliberately. For instance, a message can be going from (1,1) to (2,5) and if there is congestion at (1,3) node may choose to take (1,1) → (1,2) → (1,3) → (2,3) → (2,4) → (2,5) instead of choosing path through (1,3). Adaptive routing algorithms usually use local router information to get the congestion information, buffer occupancy, links availability/failure. The process of adaptive algorithm is shown in figure 3.15.

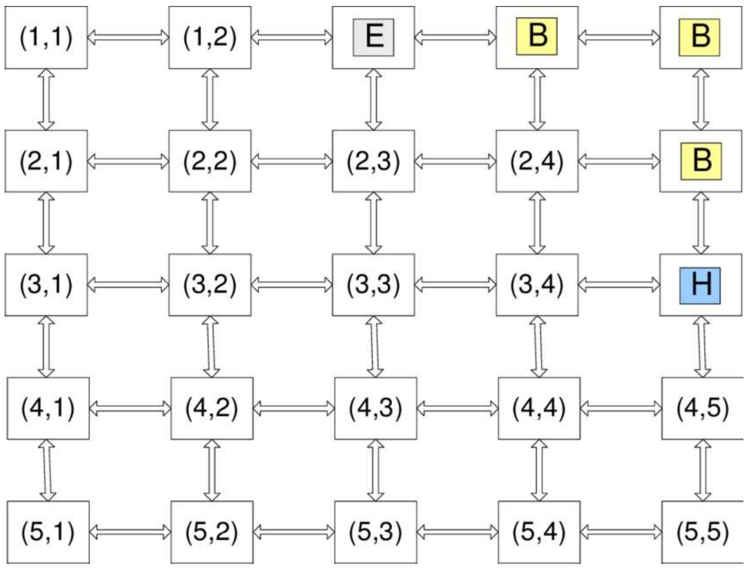


Fig. 3.15: Adaptive algorithm

The adaptive algorithm can further be categorized as Minimal Adaptive algorithm and Non-Minimal Adaptive. Minimum routing algorithms always use the shortest route to reach the target. In contrast to the minimal adaptive routing algorithm, where a packet only searches for the shortest path, non-minimal adaptive routing enables the packet to take a longer path if no shortest path is available.

3.3.4 DISTRIBUTED AND SOURCE ROUTING

Both static and dynamic directing plans can be additionally ordered relying upon where the routing data is put away, and where routing choices are made. In appropriated directing, every data bundle conveys the destination address (e.g., XY facilitates or number recognizing the goal hub or switch), and routing choices are made in every switch by looking into the goal addresses in a steering table or by executing an equipment work. In this way every system switch can be considered to actualize a capacity that takes the goal address of a data parcel as an information and creates a directing choice to reach to the output node. At the point when a data packet lands at the info port of a switch, the routing table is counseled (or directing rationale is executed) to decide the bundle's yield port dependent on its goal address. In source routing, pre-processed directing tables are put away at a hub's (or PE's) NI. At the point when a source hub transmits an information bundle, the directing data is gazed toward the source switch (or NI) in light of the destination address, and this data is added to the header of the parcel. Every data packet therefore conveys the directing decisions in its header for each node in its way. At the point when a data packet arrives at a switch, the routing data is removed from the directing field in the parcel header. Dissimilar to the source routing algorithm , source directing does not require a goal address in a parcel, any middle of the road directing tables, or capacities expected to discover the path to destination. Be that as it may, source directing requires extra routing data in a bundle header, and the quantity of bits increments for longer ways. Extra directing tables may likewise be required with explicit passages for each source.

3.3.5 MINIMAL AND NON-MINIMAL ROUTING

Another way to deal with perceive coordinating plans is to arrange them as insignificant and non negligible directing . A steering is negligible directing if the length of the guiding route from the source to the objective is the briefest possible length between the two center points. For instance, in a NoC topology if the source center is at (0, 0) and the objective center point is at (I, j), by then the insignificant way length is |i| + |j|. In negligible directing , the source does not start sending a bundle if insignificant way isn't open. Curiously, a non-negligible coordinating arrangement does not have such necessities and can use longer ways if an insignificant way isn't available. By allowing non-insignificant ways, the amount of elective ways is extended, which can be significant for keeping up a key separation in blockage circumstances.

CHAPTER 4. LITERATURE SURVEY

4.1 BACKGROUND

Network-on-Chips (NoC) is the utmost preferred interconnection paradigm made to increase the parameters: (i) extensibility of bandwidth related to bus architectures (ii) energy competence (iii) consistency (iv) reusability (v) low power consumption and (vi) scalability considerations.

NoC's are primarily responsible for ensuring communication between IPs. The reliability of the network depends on good communication architecture. As the connectivity is incomplete, it is required to route the packet among a destination and source nodes. A Routing algorithm calculates the complete route to reach the data packets from source to destination. The algorithm should be able to distribute the traffic load across the network evenly. It should be capable of optimizing the performance parameters, i.e., network latency and throughput. We have already presented taxonomy on the routing algorithm in chapter two. There are various challenges that may encounter during the routing process. The problems like deadlock, livelock, congestion, and faults are very common and need to be addressed during routing. The challenges are discussed as follows:

Deadlock: In NOC protocols for routing, this problem consider as the hardest. Deadlock generally follows when the data packets are not able to make progress as they may be waiting for each other to release the resources (links, buffers, etc.) holding by them. As the number of resources is finite, Deadlock freedom is a must for on-chip networks. Three strategies deal with deadlock: preventing deadlock, preventing deadlock and recovering from deadlock. An obvious solution is to use the deterministic routing in which all data packets sent between pair of each source and target using a fixed path. Deterministic algorithms, however, do not provide alternative routes. In [25], the author presented necessary and sufficient conditions for the routing algorithm to be deadlock free. In [26], the author has removed the deadlock by eliminating cycles. A cycle has been removed by adding virtual channels or new vertices in the path and then rerouting data packets on new channels. The main reason of deadlock is the cyclic acquisition of channels in the network, usually detecting and breaking deadlocks are expensive; therefore it is better to adopt deadlock free algorithm.

Livelock: When the data packet retains wandering about its end without ever accomplishment it, this situation is referred to as livelock. Non-Minimal algorithms usually suffer from livelock situation. Time to live (TTL) and packet's age are most commonly used ways that are used to avoid livelocks. TTL uses a hop counter to count the traveling time of a packet in the network and the packet has been removed from the network, as soon as the predefined threshold value occurs. Data packets can also be given priority based on their age. The oldest packet always gets the highest priority and will be routed forward.

Faults: Another challenge encountered in modern NoC's is faults due to the design complexity and physical defects in the deep submicron domain. The need of developing fault-tolerant approaches through designing routing algorithms in such a way so that there must be a path between all source-destination pairs in the presence of faults while ensuring freedom from deadlocks and livelocks is primary solution for handling faults .

4.2 2D NOC ROUTING ALGORITHMS

There are several routing algorithms that have been implemented and used for 2D NoC. The algorithms may or may not use virtual channels. One of the simplest routing algorithms used for 2D NoC is XY algorithm [27]. It is also considered as shortest path algorithms; the packet first traverses through X direction until the destination's X coordinate has reached and then move towards destination's Y coordinate. Because of the simplicity of the XY algorithm, other variants of XY are presented in [28,29]. In [28], Patooghy et al. presented XYX algorithm that is deadlock and fault tolerant also. 60% fault tolerance is achieved by sending a copy of data packets through alternative routes and also adopt an error detection technique.In [29], the author has presented DyXY algorithm where the network state (lowest traffic route) has also taken into consideration while routing packets towards the destination. Another derivative is presented in [29] as adaptive XY where deterministic and adaptive both approaches are utilized while making a routing decision. Whenever there is traffic or congestion, adaptive XY is followed otherwise simple deterministic algorithm will work. The 2-bit identifier is used to depict each port congestion status. Another similar work carried by Hu, J. in [30], the author proposed DyAD as a smart algorithm. It exhibits the combined benefits of deterministic and adaptive approaches. In a less congested network, the deterministic algorithm has the less latency advantage, whereas adaptive approaches work well in a congested network and find an alternative path for sending data packets from source to destination. Each router keeps on checking the congestion status of its output port and accordingly operates between deterministic and adaptive routing. This algorithm is benefited as less latency in less traffic, high throughput in congestion and it distributes the traffic load as well. Similar work has been evaluated by Dehyadgari et al. in [31], an author named it as pseudo adaptive XY algorithm. As the load is much higher in the center the distribution of load will be done. It uses XY routing (deterministic) in low congestion and adaptive routing in high congestion. Bobda et al. [32] has introduced Surrounding XY (S-XY) routing that operates in three routing modes N-XY, SH-XY, SV-XY. The algorithm will adopt N-XY when there is no congestion. SH-XY will be used when the node's left or right neighbor goes down. If upper or lower neighbor is not working then, SV-XY will be used. Other variants of XY algorithm “XY-YX” and “IX/Y” are presented in [33,34]. These algorithms use two virtual channels and are deadlock and livelock free.

In NoC, data packets usually pass through many switches/routers before reaching the destination, but there may be many circumstances like deadlock, livelock, and starvation which do not allow the packet to reach to the destination. Among these challenges deadlock is one of the significant issues. Deadlock can be defined as a situation in the network wherein each packet’s header has not reached its destination and requires more cushions while keeping the cradles at present putting away the parcel, in such a circumstance stop may happen. A deadlock happens when certain bundles can't progress to their goal in light of the fact that the asked for cushions are not accessible. Another explanation behind the stop to happen is the level of adaptiveness. The degree of Adaptiveness (DoA) is one of the important parameters for evaluating routing algorithm which determines the total number of path available between the source and the destination. The more the DoA, more chances are there for the successful delivery of the data packet to the destinations. Sometimes to achieve higher DoA, the number of possible path increase and which in turn increases the chances of deadlock. Turn model [36] algorithms introduced by Glass et al. are widely adopted routing algorithm for avoiding deadlock and livelock situations. Deadlocks and livelocks situations can be detected, broken and prevented. As avoiding deadlocks and livelock situations are much easier and less expensive rather than detecting and breaking them. Therefore turn model algorithms are adopted for NoC mesh network. The packet can traverse in four directions north, south, east, and west in mesh-based topologies. There are four clockwise turns and four anticlockwise turns. A cyclic condition can be formed if a packet can move in all four directions. Therefore in Turn model, we prohibited few turns and packets are not allowed to move on that turn. In figure 4.1, the various variants XY, North-Last, West-First, Negative-First, Odd-Even of Turn model is shown.

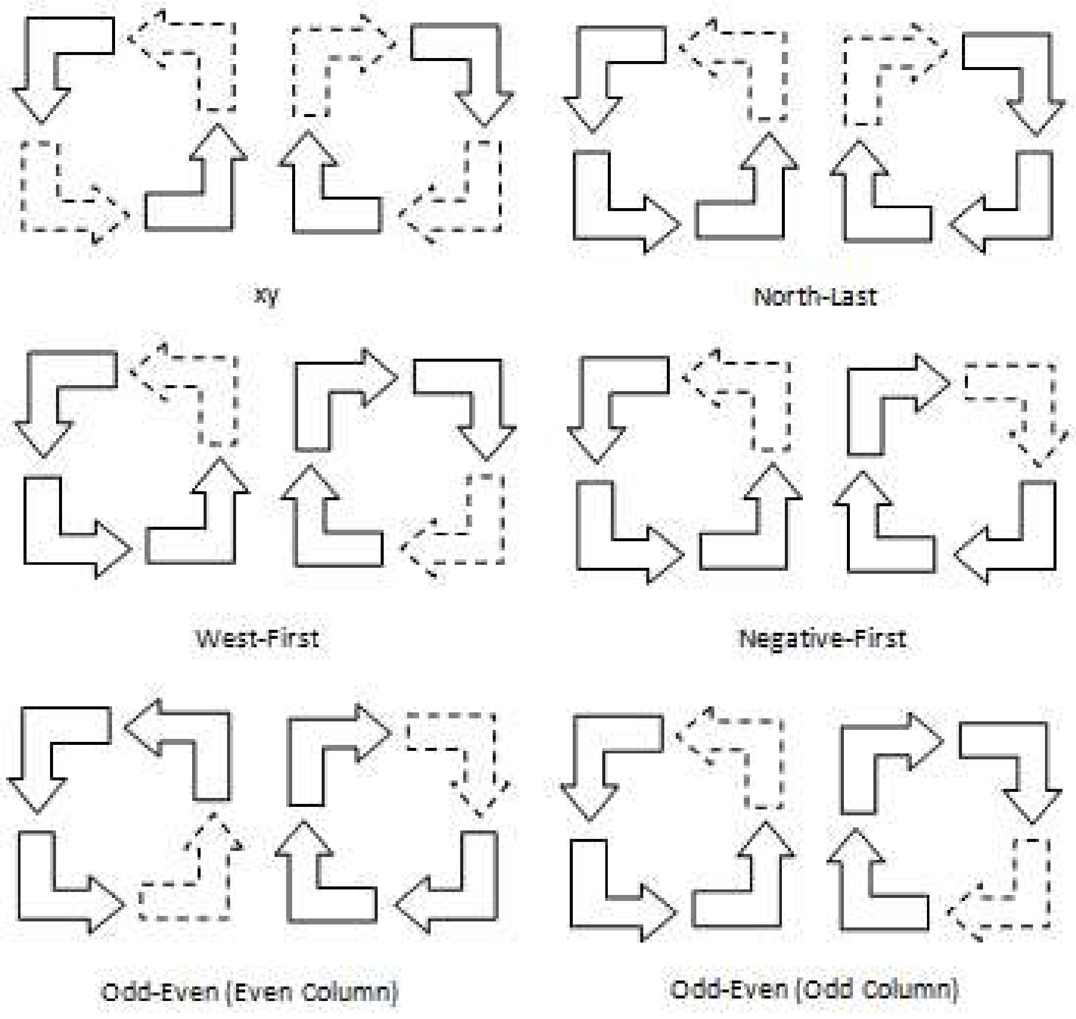


Fig. 4.1: Turn model [36]

In North-Last algorithm, the directions North-West and North-East are eliminated, and ones the packet is in north directions, no further turns are allowed in any directions. Hence the turn to north has to be made last. In West-First algorithm, the packet will move to west direction first before traveling to the different direction to reach to the destination. In Negative-First algorithm, the packet can take only negative directions, i.e., packet can travel from east to south, west to south, and north to west. Turn model [37] proposed by Chiu et al., the columns are represented as odd and even according to x coordinate. In 2D-mesh all node is identified by its coordinates (x,y). If x coordinate is odd - column is considered as odd and if x coordinate is even - column is considered as even. Thus as per the algorithm model, if the packet has to move from even column, the packet cannot make a turn to north or south from the west. If the packet is an the odd column, it cannot turn east. In [38], author proposed "Abacus Turn Model" that includes two algorithm "Tug-War" and "Arm-Wrestling" that guarantees deadlock and livelock freeness by applying indistinguishable guideline from the odd steering calculation, which dependent on the way that a system is free of deadlock if right-hand segments are expelled from the system in both the clockwise and the counter-clockwise headings. The two calculations accomplish ideal execution in uniform rush hour gridlock and lessen bundle idleness for all applications. In [39], the author introduced "NARCO" based on odd-even turn model. The algorithm uses the concept of Odd-Even (OE) routing with two virtual channels. The algorithm is also faulted tolerant by making use of virtual channels and duplication of packets while routing. Another algorithm presented in [40] is also deadlock free and fault tolerant. In this method, each node sends "I am an alive message" to other nodes to detect failures.

In [41], Tsai et al. presented another variant of odd-even i.e. Non-Minimal Odd-Even (NMOE). The algorithm uses one of the three sets directions (0^o, 90^o, 180^o) to choose an output port. The algorithm selects a set and directions by looking at the input queue size of the router. If the algorithm uses directions from set 0^o, the algorithm will work like a minimal odd-even algorithm. If all input ports are full in set 0^o, the algorithm will consider directions of set $90^{o,}$ and further, if input ports of set 90^o are also full, the set 180^o directions are selected. Liu et al.[42] proposed Efficient Dynamic Adaptive Routing (EDAR), where dynamic routing takes place at the time of congestion. Each port will be assigned value based on busy, congested and faulty conditions. The busy and faulty condition will be assigned a minimum and maximum values. The port with the minimum value will be considered for forwarding the packets towards the destination.

In [43], the author has investigated probabilistic flooding, directed flooding and redundant random walk algorithms to address the issues of fault tolerance, deadlock, and livelock. These algorithms suffer from major drawbacks of storage, more power consumption, and communication overhead. In [44], Kim et al. proposed arbitration look ahead scheme where the arbitration stages at each router are skipped. As the routing path is already determined, each router is supposed to re-establish the path to be followed by the packet towards the destination and avoids input queuing and other stages. This the advantage is a considerable reduction in latency, but this algorithm only works with deterministic algorithms and zero congestion networks.

In [45], the author proposed distance vector routing and link state routing for 2D NoC's. In distance vector routing, each router required to maintain a table about the destinations and the respective links where it can send the data packets, and share this information to its neighbor's only. As distance vector routing do suffer from "counting to infinity" challenge, the link state routing can be considered where each router is supposed to broadcast the information to all the routers. In this way, all routers have a similar map of the entire network.

In [46], Chen et al. proposed PDA-FTR algorithm which considers fault path diversity and buffers occupancy to calculate Effective Buffer Length (EBL) for deciding the routing directions. Initially, odd-even turn model is used to determining the route if there exist two or more paths. EBL parameter will be calculated for each path. The higher value of EBL implies that this direction will be less congested and have better traffic flow.

In [48], the author presented “SoE” where a focal deadlock support is utilized to avoid deadlock circumstance rather than virtual channels. This gives a superior act as far as normal postponement, immersion throughput and adaptiveness than an odd-even calculation.

Ebrahimi et al. in [49] proposed “HARAQ” a Q-learning algorithm that uses a virtual channel in the Y dimension to avoid congestion in chip networks. Through this Q-learning model, idleness from each yield channel to the target can be estimated, and congested output channel is avoided. This method is very scalable and adaptable.

In [50], the author presented DuQAR algorithm which means to lessen blockage in the system by sending bundles through the most limited Q-esteem way. The unique component of this calculation is that the Q-esteem isn't much of the time refreshed to evade worldwide blockage, however just if the normal number of free cushions in the switches surpasses a base edge.

In [123], author extended turn model and introduced “PR-WF” for ensuring deadlock and fault tolerance. This technique utilizes three progressive cradles inside every hub: a switch port cushion, a system interface Pseudo-Receiver (PR) support, and a neighborhood IP reserve. This technique anticipates disallowed swings because of the West-First calculation and builds the quantity of associations in the system.

In [124], the author presented a completely versatile calculation called "Select y directing" that utilized virtual channels. The base number of virtual channels for the nD-mesh network is set to 4n-2. This algorithm uses at least six virtual channels for a 2D mesh at different turns. “Opt-y routing” is minimal and free of livelocks.

4.3 3D NOC ARCHITECTURES

This section presents distinctive models that were proposed with respect to 3D NoCs. One of the remarkable 3D NoC structures is the 3D-Mesh as showed up in figure 1.2.

In this 3D NoC structure each IP focus is connected with seven port switch appeared in figure 4.2 where one port is joined to the IP square, two ports to the switches above and underneath, and other four toward each way for instance north, south, east, and west. The inconvenience to this essential designing is exorbitantly number of pads at each port despite when it isn't required while experiencing upward and plummeting associations since latency between levels is too much low as levels are stacked in all respects eagerly [51]. The amount of ports depends upon the circumstance of the switch in the structure since we have to take out any unused associations that have no relationship with various changes to lessen control use.

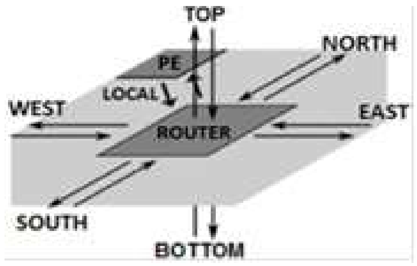


Fig. 4.2: Seven port 3D router [52]

3D NoC transport hybrid plan [53] was proposed as a creamer of package traded NoC's and shared transport building where various 2D-work layers are freely executed with group traded advancement, and vertically the layers give through shared transport designing. A switch in this blend building has, at most, six ports, i.e., one port is related with the IP, another to the vehicle, and rest four to the orientation (N, S, E, W). This tale structure was executed in 3D NUCA L2 store for CMPs [54]. It encounters lower information transmission while passing on interlayer on account of shred transport.

Ciliated 3D-work [55] was proposed as a variety of 3D fundamental work based design where each switch/switch obliges no less than two IP square. In a ciliated 3D-work organize, each switch/switch involve at most 5+k ports, one for each course (N, S, E, W), one port either for up or down layers for 2Layer 3D-work and K ports are related with each K IP squares. This structure encounters lower transmission limit as switch/switch is shared by different IP focus, anyway control scattering is low.

In [56], the Vertical Partial Mesh-based 3D NoC (VPM3NoC) structure is depicted. This plan contains work based stacked layers where each layer is isolated with respect to advancement, application, and size. In each layer, a couple of centers have vertical associates with all over layers. This structure uses varieties of switches for instance 2D switch, a 3D switch with an upward association, a 3D switch with a plummeting association, a 3D switch with up and downlink.

In [57], the makers have displayed two 3D topologies for instance 3D star topology and 3D recursive framework topology and pondered them seeing the parameters as seeks after: inaction, essentialness dissipating, and framework width. In both the topologies, get-together of four centers are confined which is named an a pack with one center is doled out as Cluster Head (CH) which can go about as CH similarly as a center. A layer has four gatherings, and the total number of centers in a layer is sixteen. TSV's are used to relate the three layers vertically. The Performance of 3D RNT is evaluated and found better performing in relationship with 3D ST concerning inaction and imperativeness scattering parameters.

In [58], the makers have proposed 3D 2-layer and 3-layer structures. In both the strategies, the goal is to update the floor-space of the chip. In 3D 2-Layer approach, the layer one is focused on putting heterogeneous IP focuses while layer two is used to place switches in a work setup, relatively, in 3-layer plans, layers one and three are used to put heterogeneous IP focuses, and layer two executes work structures of allotted switches. The 3-layers are related with each other Through Silicon Vias (TSV).

POC topology [59] was proposed as another variety of 3D-work topology which contains two unmistakable sorts of switches for instance 7-port switch and a 6-port switch. 7-port changes are used to interface with IP where the 6-port switch was used for correspondence. The qualification between 3D-work and POC topology is that the vertical associations are simply executed at the 6-port switch. The advantage of this topology is less vertical associations which lead to low imperativeness spreads.

Another 3D-work based NoC designing called Lasio [60] is moreover displayed where skips between layer (z-center point) and bounces inside the layer (X, Y-turn) have a comparable cost. The most basic part of this designing is that all switch ports are bidirectional. It executed XYZ coordinating and used credit-based stream control.

Another class of 3D NoC topologies called Xbar-related Network-on-Tiers (XNoTs) [61] proposed by Dally, w , including different layers and each layer is changed with different topology as demonstrated by the application .The layers are solidly related through crossbar switches which diminishes the ordinary hop check and low power use.

3D honeycomb NoC topology [62] proposed by Yin et al. is confined by stacking 2D honeycomb topology which is made out of hexagons. This topology encounters the high use cost on account of endless correspondence joins. This designing has a framework dimension of 5 which prompts higher framework cost.

In [64], 3D tree based SPIN is proposed, and from its detailed outcomes, it very well may be presumed that when the 2D BFT system is mapped to a 3D SoC multi-layer, wire steering winds up less complex and the longest exchange wire length is decreased. This will prompt lower vitality scattering and decreased region overhead.

In [65], Multi-layered on-chip Interconnect Router Architecture (MIRA) is proposed. MIRA is a 3D stacked NoC switch engineering having various layers and advanced to diminish the general vitality utilization and region prerequisites.

In [66], De-Bruijn based topology is proposed which is intended for parallel handling. The advantages of De-Bruijn chart organize topology are little distance across, high network, and high dependability. The level of this 3D design does not change with an expansion in system measure.

The De-Bruijn design offers better throughput contrasted with the NoC 3D-work because of its littler width.

In [67], the creator has proposed an effective 3D NoC design for burden adjusting for example On the off chance that one layer is blocked and different layers are light; the framework isn't productive as far as power and execution. In this manner 3D design for vertical correspondence with burden adjusting thought has been proposed. The layer-multiplexed engineering all the more correctly replaces the one-layer per-bounce directing in a customary 3D work with less difficult vertical de-multiplexing and multiplexing stages.

In [68], creator proposed 3D engineering Bidirectional Bisynchronous Vertical Channels (BBVC) - based NoC for advancing TSV region impression. In 3D NoCs, the quantity of centers in each layer increments to help the expanding intricacy of the application. The correspondence between layers is additionally expected to increment, and the quantity of interconnected TSVs will subsequently increment. Since each TSV requires a cushion to tie to a wafer layer, the impression of the TSVs is never again immaterial in each layer. Along these lines due to this reason the creator has investigated a component to lessen the impression of the TSV region. 3D IC costs, routability, warm proficiency, and power utilization are in this way improved. The fundamental thought of the proposed 3D NoC framework is to utilize a two-path channel for interlayer correspondence at a higher recurrence than intra-layer correspondence. It can powerfully alter the course of the direct between switches in neighboring layers dependent on the ongoing data transfer capacity necessity.

4.4 3D NOC ROUTING ALGORITHMS

One of the important designs challenges that should be focused upon conceptualizing 3D NoC is the design and implementation of an effective routing algorithm. As of now most of the routing algorithm developed so far is focused on 2D NoC. Algorithms like Dimension Ordered Routing (DOR) [68], Valiant [69], ROMM [70], O1TURN are extended to the third dimension.

4.4.1 CONGESTION, DEADLOCK AND FAULT TOLERANCE

Fault-Tolerant Dynamic XYZ (FT-DyXYZ) [54] proposed by Jouybari et al. is capable of handling runtime and design time faults on the vertical and horizontal links both. This algorithm used congestion parameter according to occupancy in input buffers while deciding path between source and destination. This algorithm utilizes the concept of misrouting when there is no output port left towards a destination or if there is heavy congestion on the preferred path or due to the removal of faulty links.

In [69] presented a fully adaptive fault-tolerant routing algorithm for Network-on-Chip architectures. They proposed a novel routing algorithm called Force-Directed Wormhole Routing

(FDWR) was presented in this paper for handling fault-tolerant mechanism. Even when the links are a fault, the traffic across the whole network is balanced by the proposed algorithm. To achieve fault tolerance, virtual channels were not used in FDWR. In our introduced system, the wormhole routing principle was used. There are two advantages to this principle. One was the data will be transferred in a pipelined manner, and the other was, it has only a limited buffer space. According to this principle, packets are divided into flits to transfer it from the sender to the receiver one by one. The header flit was the first flit of the packet where the information about the destination address was contained. To examine the buffer status of its neighboring switches, the header flit was used as a look-ahead flit. Based on the routing table of a switch and the buffer status of its neighboring switches, the routing decision was made. In the previous routing algorithms, with an unequal number of hops, different packets were routed to the destination on different paths. This results in out of order delivery of packets. But in our approach, the packets from higher protocol levels can be reordered because sequence numbers were used by the packets. Using system C, this algorithm was implemented in the switches of a Transaction-Level Model (TLM) packet switching NoC. Topologies for NoCs can be automatically generated based on the switches, mesh, torus, and hypercube. To bypass overload in links, the traffic was distributed regularly across the whole network by using forces. Simulation results describe that even in the case of faulty links, packets can be routed.

In [70] Pasricha et al. had described a reconfigurable and adaptive routing method for fault-tolerant mesh-based Networks-on-Chip. During the design of on-chip networks, the main goal was to reduce an unwanted effect which leads to unreliability. In mesh-based Networks-on-Chip to tolerate faulty links and routers, a very low-cost fault-tolerant routing method was presented known as fault-tolerant routing method. To support irregular topologies caused by faulty components, this algorithm can be dynamically reconfigured. Here for providing adaptiveness and fault-tolerance, only two virtual channels were utilized. With additional hardware costs, the proposed routing method has the ability to tolerate more faulty components in more complicated faulty conditions because it has a multi-level fault-tolerance capability. Addition to this, congested traffic was also issued generated by faulty components by integrating different congestion-aware approaches. In this method, there are two types of variants. Single faulty links are supported by the basic variant of this method, but faulty links and many faulty regions were supported by the main variant. Both of them do not need any extra virtual channel. Via appropriate simulations and syntheses, the performance of the network, the ability of fault-tolerance and hardware overhead can be calculated. Although this proposed algorithm contains a small hardware overhead, the experimental results clearly show that the overall reliability was increased.

In [71] Ahmed et al. has proposed to Look Ahead XYZ Algorithm (LA-XYZ) to handle the inefficient pipeline stage of conventional XYZ and executed the routing calculation stage and switch arbitration stage together and pre-computes the next port towards the destination. This algorithm reduces latency but has not taken consideration of fault information. In [72] author proposed to Look Ahead Fault Tolerant (LAFT) routing which employs Fault Control Module (FSM) and takes advantage of LA-XYZ. FSM uses the 6-bit signal for providing the status (faulty or valid) for each link (N, S, E, W, Up and Down) with each router. Thus by employing FSM, overall system performance has improved, but LAFT may not select the best route.

Further, in [73], author has proposed an improved version of LAFT as Hybrid-Look-Ahead-Fault-Tolerant (HLAFT) that always searches the minimum distance route towards the destination .It takes the benefits of LAFT and uses a technique called random access buffer for deadlock recovery. In [74] Somasundaram et al. has proposed multi-dimensional flow algorithm which initially searches all possible augmenting paths from source to destination. For each path, residual value (capacity of link-flow of the link in both directions) is calculated for each link. Finally, the path having maximum residual value and minimal path length is selected. The topology taken here is Hamiltonian which ensures that there is always a path between any pair of nodes.

In [75] Dubois et al. has put forward elevator first - deadlock-free routing for partially connected 3D NoC. In this algorithm, if the destination node is on another tier, the algorithm adds the address of nearest vertical alignment or elevator to the packet header. Therefore the packet uses the nearest elevator to reach to the destination tier. After reaching to the desired tier, elevator information has been removed from the packet header, and conventional XY algorithm is used to reach to the destination.

In [76] Liu et al. has presented path optimized multicast routing algorithm, where each intermediate router participates in deciding the path towards the destination. This algorithm achieved lesser communication energy. In [77] Ying et al. proposed uniform solo minimal routing which provides better throughput. Author has modified the dimension order routing and devised ZXYXZ, XYZYX, and YZXZY routing. The logic behind the algorithm is to randomize the traffic load in two dimensions and route minimally in the third direction. The approach also uses three virtual channel over one physical channel to avoid deadlocks.

In [78] author developed a quadrant based XYZ dimension order routing and uses torus topology for their work. The algorithm partitioned the topology into quadrants and used the nearest wrap around edge for reaching to the destination. This algorithm has o(n) complexity and always finds a minimal path.

Topology Aware Adaptive Routing (TAAR)[79] for irregular mesh topology is presented by Chen et al.. It uses the inter-layer and intra-layer routing for avoiding the throttled (faulty) nodes. The algorithm provides larger path diversity which leads to more balanced traffic at the time of congestion. In [80] author has proposed fully adaptive routing algorithm for 3D-mesh with a limited number of TSV's. The algorithm operates in two modes Interlayer routing and intra-layer routing. Initially, the algorithm operates in Interlayer mode where nearest vertical node to the source is selected to send the packet to the destination layer. Then after reaching to the destination layer, the algorithm operates in intra-layer mode. The routing is fully adaptive here as while selecting the route to the destination, path diversity, traffic conditions, and network state is also considered.

In [81] Ebrahimi et al. has designed algorithm for the 3D NoC's where few nodes have a pillar or vertical links for delivering the packet to the upper layer or downward layer. Thus for delivering the packet to the destination layer, the nearest pillar is used. Afterwards, the conventional XY routing is used for sending the packets to the destination. In addition to the above logic, two virtual channels are also used for traffic management. For deadlock avoidance, the network is divided into two disjoint set of nodes, wherein the packets in each set are allowed to use only one virtual channel according to the predefined directions. No u-turn is allowed in the algorithm.

A similar type of work has been done by Ying et al. in [82] where author has proposed two routing algorithms as Source Based Shortest Manhattan (SBSM) distance and Destination-Based Shortest Manhattan (DBSM) distance where XY algorithm is followed for interlayer communication and for interlayer communication proper Vertical Link (VL) is chosen. The two constraints imposed here is that the distance between two adjacent VLs node cannot be more than two. The other constraints are to select the VL with shortest manhattans distance to the destination. The algorithm is deadlock free. In [83] Naghibijouybari et al. presented FT-Z-OE algorithm. In the absence of faults, the packet is routed towards z-direction from the source node. After reaching to the destination layer, OE [36] routing algorithm is used for delivering the packet to the correct destination. In the presence of faults, i.e., if the vertical link of the source is faulty, the packet will be navigated towards the image of the destination node on the same layer. Afterwards, the vertical link of the image will be used to sending the packet to the destination. If the vertical link of image and source both are faulty, then misrouting will be done. In [84] Viswanathan et al. has presented three 3D topologies mesh, star and recursive with three layers. Each layer has four clusters. The clusters are formed by grouping four nodes and one of the nodes in the group designated as Cluster Head (CH). The idea here is to navigate or communicate only through CH for all inter-layer, intra-layer, and inter-cluster communication. The algorithm is deadlock free, but congestion conditions and fault tolerance are not considered in the scope of work. In [85] author proposed routing algorithm for avoiding deadlocks and faulty links. The proposed routing algorithm calculates the cost of feasible routes between any pair of source and destination. The cost is measured in terms of distance, i.e., minimizing the number of hops. The algorithm uses the tagging method for fault reporting in the network. Thus when a fault is encountered, the nodes connected to faulty switch/link sends one flit long error report to their neighbors. Further neighbors send the fault report to their neighbors with the relative position of faulty link/switch. Therefore when there is no fault, the minimum cost route is selected between source and destination. If there is fault detected on the minimum cost route, the second minimum cost route is selected.

Lin, et al. [86] has given an answer for clog, deadlock and blame mindfulness information in channel evaluation ability to evade the hotspot around the imperfect switch in NOC using Ant Colony Optimization-based Fault-Aware Routing (ACO-FAR) figuring for weight changing in broken frameworks. The lead of an underground creepy crawly territory while going up against a deterrent (dissatisfaction in NoC) can be depicted in three phases: 1) Encounter; 2) Search, and 3) Select. They executed the relating frameworks as 1) Notification of accuse information; 2) Path looking instrument, and 3) Path picking segment. With proposed ACO-FAR, the switch can evaluate the available ways and sidestep packages through a less-blocked accuse free way. The amusement results show that this paper has a higher throughput than related works by 29.1% – 66.5%. Likewise, ACO-FAR can decrease the undelivered bundle extent to 0.5% – 0.02% and balance the movement of traffic stream in the broken framework.

In [87] Wang et al. proposed an Energy and Buffer-aware fully Adaptive Routing Algorithm (EBAR) is proposed to adjust the warm dissemination and fulfill the execution prerequisites. A system state work display, which takes both verifiable and current conditions of the system into thought, is developed to adjust the dissemination of warm and ease the system blockage. In the then, another warm administration conspire is proposed to bring down the temperature and fulfill the execution prerequisites of high need bundles. The reproduction results demonstrate that EBAR can give an enhancement in warm dissemination without execution corruption. The upgrades of start to finish postponement and throughput additionally exhibit the focal points that can be given by EBAR. The proposed EBAR calculation is an effective method to give a superior execution of warm dispersion under various traffic designs with better End-to-End (ETE) postpone execution.

In [88] have proposed a novel QoS aware and congestion aware Network-on-Chip architecture that not just empowers quality-situated system transmission and keeps up a practical usage cost yet in addition well parity traffic load inside the system to improve generally throughput. By separating application traffic into various administration classes, transfer speed allotment is overseen in like manner to satisfy QoS necessities. Consolidating with blockage control plot which comprises of dynamic mediation and versatile steering way choice, high need traffic is coordinated to less clogged regions and is offered inclination to accessible assets. Reproduction results demonstrate that the normal inertness of high need and by and large traffic is enhanced significantly for different traffic designs. Cost assessment results additionally demonstrate that the proposed switch engineering requires unimportant cost overhead yet gives better execution to both propelled work NoC stages.

In [89] author had described a low-overhead fault-aware deflection routing algorithm for 3D Network-on-Chip. In 3D NOC to make the routing decision efficient and to avoid both TSV and horizontal link faults, a low-overhead fault-tolerant deflection routing algorithm was introduced. The proposed algorithm makes use of both layer routing table and two TSV state vectors. In hardware with 65nm TSMC technology, the proposed switch was executed, and it can produce up to 250MHz. Our proposed system consumes less power and less area when compared with reinforcement-learning-based fault-tolerant deflection switch which uses global routing table.

In [90] author had developed a deadlock-free and plane-balanced adaptive routing for 3D Networks-on-Chip. They utilized the odd-even turn model for deadlock-freeness. Here based on the concept of dynamic programming networks, performance was improved. To avoid waiting cycles and also to ensure deadlock freeness there were some rules. In the first rule, packets were not allowed to take North-West (NW) or South-West (SW) turns when they were in odd column. In the second rule, packets were not allowed to take East-North (EN), or East-South (ES) turns when they were in even column. In the third rule, the packets cannot enter into the even xy-plane, which were traveling upward and the packet cannot enter into the odd xy-plane, which were traveling downward. The rules 1, 2 and 3 can be applied for 3D odd-even routing and was called conventional OE. Whereas for Plane balanced 3D odd-even routing, these rules were modified. In the fourth rule, packets are not allowed to take West-North (WN), or East-North (EN) turns when they were in an odd row. In the fifth rule, packets were not allowed to take South-West (SW), or South-East (SE) turns when they were in even row. These modified rules were used to ensure deadlock-freeness.

In [91] author had proposed a highly adaptive and deadlock-free routing for 3D Network-on-Chip. In this method, a turn model for adaptive routing algorithms from 2D to 3D NoCs was extended. For different layers, various rules were applied resulting in a different restriction on traffic flow. A higher degree of adaptiveness was achieved using 3D plane-balanced approach. The main aim of 3D odd-even with DPN routing algorithm was to free the deadlock and to analyze the path diversity.

In [92] author had introduced a without deadlock directing calculation requiring no virtual channel on 3D NoCs with incomplete vertical associations. From the lift initially directing calculation, a strategy known as Redelf was altered on a 3D work topology. To guarantee deadlock-opportunity, this strategy does not utilize any virtual channel. In Elf, when the goal and source were in various planes, the calculation can pick any lift, however in the proposed methodology, there were a few principles. The primary standard was, they need to take the closest lift among the south-or-due-east ones of the source when the goal was contained in the distinctive layer. The second guideline was, take the rotate lift, if there was no south-or-due-east lift of the source in the ideal bearing. The third standard was, we need to expect that, a down lift was not at the self-hub, and it was picked by rule1, and furthermore we need to accept that it was situated toward the south or due east of the up rotate lift, at that point bring the down turn lift on a similar plane as opposed to bringing the down lift. Whenever here and there heading was exchanged equivalent guideline was used.

In [93] author had implemented fully adaptive routing algorithms and region-based approaches for two-dimensional and three-dimensional networks-on-chip. If the congestion level was measured for a group of routers and broadcasted over the network, then the performance increases. A distributed approach was proposed here, to gather and spread the congestion information of different clusters. To deliver packets via less congested regions, the routing units used the collected information. Here they have proposed a fully adaptive routing algorithm for 3D NoCs. A region-based approach was introduced on top of each fully adaptive routing algorithm. By balancing the traffic load over the network, congestion can be skipped. To do so the information about the traffic conditions in different regions should be kept updated and the packets should be routed through less-congested regions. Based on regional traffic condition, if routing decisions were made then the performance increases. Therefore without any routing restriction, packets have to be routed using all the shortest paths. Only minimum number of virtual channels is needed for this algorithm. Here the network can be divided into two parts. In one side, to remain free from deadlock and to reduce the number of virtual channels, the two main subnetworks were separated. On the other side, each main sub-network contains a subnetwork in which different sets of virtual channels were used. Therefore to maintain the network as deadlock-free, the network was divided into eight sub-networks.

In [94] Ebrahimi et al. had built up a completely versatile directing calculation for 3D NoCs. To transmit parcels from source to goal hub without influencing the execution, a completely versatile steering calculation known as DyXYZ was intended for 3D NoCs. To choose data among the yield channels the clog data at the information cradle of the neighboring switches was utilized as a blockage metric. 4, 4, and 2 virtual channels along the X, Y, and Z measurements were utilized by this calculation to give without deadlock. Bundles were conveyed through three sent to a middle of the road hub a less blocked way utilizing this calculation. Until the bearing achieves, zero the bundles were conveyed. To give sans deadlock character here the system was partitioned into eight sub-systems, for example, ENU, END, ESU, ESD, WNU, WND, WSU, and WSD where N, S, E, W, U, and D represent North, South, East, West, Up, and Down headings individually.

In [95] the creator defined a savvy directing for Networks-on-Chip. Thus joining the ideas of the low dormancy of the deterministic directing and the high throughput of the versatile steering dependent on the system's clog conditions, another idea called DyAD is actualized. It might prompt deadlock when odd-even and XY are incorporated to shape a DyAD switch. For this situation, we actualize another idea for DyAD known as DyAD-OE under the idea of odd-notwithstanding directing for expelling the odd even's adaptiveness. A parcel will dependably be steered to p1 in oe-fixed if a bundle with a given source and goal can be directed to both yield p1 and p2. The info bundles are cradled by the FIFO before conveying them to the yield ports. The location decoder forms the flit and sends the goal address to the port controller when another header flit is gotten. This decides to which yield port the parcel ought to be conveyed. To course bundles, if there is more than one yield bearing in the odd-even mode, at that point the heading is chosen by the port controller dependent on the more vacant spaces in its information FIFO. So as to set up a way to the comparing yield port, the port controller sends the association demand to the crossbar referee once the switch has settled on its choice to course improper heading. FIFO occupation proportion is checked by each info port. An esteem 1 will be appointed if the proportion achieves the preset blockage limit else esteem 0 will be doled out. To decide if to concede association consent to the port controller, the status of the present crossbar association is kept up by the crossbar judge. The crossbar judge utilizes the FCFS approach to choose which input port for conceding the entrance to which input port when there exist numerous information port controller's solicitations for the equivalent accessible yield port. Along these lines, starvation is maintained a strategic distance from.
Hoda Naghibi and Karim Mohammadi [96] author had presented a low overhead, fault tolerant and congestion-aware routing algorithm for 3D-mesh based NoCs. Most of the errors occur due to runtime and manufacture faults. Here to tolerate faulty links we proposed an adaptive fault-tolerant routing known as FT-DyXYZ. The performance can be increased when the number of virtual channels was increased. To reduce runtime faults, every node must be aware of its own faulty links. After determination of preferred output ports, if there were permanent faulty links, based on congestion parameter one output port is chosen after the removal of the faulty links from the preferred link set. But in the case when the output port is not left to route packet on the minimal path after the faulty links were removed then misrouting should be done that is anyone intermediate node should be selected randomly then routing was done at two phases. When the phase bit was 0, the packet was routed from the current node to intermediate node, and when the phase bit was 1, the packet was routed from intermediate node to destination node. To avoid deadlock situations and to tolerate faults, at least two virtual channels are needed in a fully adaptive routing algorithm.

In [97] author had developed a fault-tolerant Network-on-Chip based on Fault-aware Flits and deflection routing and also fault-tolerant buffer less NoC was designed. Here both deflection routing and permutation network techniques were combined. In order to achieve the energy-efficient design, it avoided the routing table and also the exchange of fault information between the routers. If there was failure occurred in two-links, then all the four permutor blocks contained in the central unit was used. Fault Status Handler (FSH) and fault aware flits were introduced so that complex fault situations can be avoided. If the packets were carefully routed away from its destination, then the faulty links can be avoided. To avoid fault links, the flit-structure was extended by three additional bits and to forward this flit in the necessary direction, this bit signals the nearby routers. By getting an acknowledgment of received flits and by counting the number of failed transmissions, faults can be detected.

In [98] described a reconfigurable fault-tolerant deflection routing algorithm based on reinforcement learning for Network-on-Chip. Based on reinforcement learning, a reconfigurable Fault-Tolerant Deflection Routing (FTDR) algorithm was proposed here to tolerate faults. In view of a sort of fortification Q-learning, the directing table was reconfigured utilizing 2-jump blame data. A various leveled Q-learning based deflection steering calculation was likewise presented, to diminish the span of the directing table. To assemble the directing table Qx in each switch, the quantity of jumps to the goal was utilized for deflection steering. In the steering table, there are n×m sections, where n is the quantity of switches in the system and m is the quantity of neighbor switches. To send a parcel to a goal, the course with the most modest number of bounces was chosen by the switch. The one with the littlest pressure esteem was picked by the switch if there are a few headings with an equivalent number of jumps to the goal was available. At the point when a parcel to d was sent to x through y, the base number of jumps to d back to x was returned by y. The relating section with 1 bounce in addition to the base number of jumps to d from the steering table refresh work was refreshed by x. The directing table can't be refreshed, if there was no blame in the system. Every single table section comparing to this heading are set to ∞ in the event that one connection was broken in the switch. The base number of jumps from each port to every goal was signified by the fixed esteem. To decrease the normal bounce checks, 2-jump blame data was used. In view of the 2-jump blame data, on the off chance that just a single connection was not blamed, at that point all passages in the table set to ∞ with the exception of that interface. The table passages from d to all goals along j were refreshed with the past Q-esteem in addition to 2 if there are a 2-jump interface with blame. When achieving a switch in the meantime, there are four parcels. From the most elevated need parcel to the least, directing choice was done by the switch. At first, the goal ID of the bundle was determined by the switch, and after that the directing table was watched that if the

parcel has achieved the goal or not. The switch gazes upward the profitable direction(s) with the base number of jumps to the goal from the steering table, and after that a free beneficial port with the littlest pressure esteem was picked to course the parcel if the bundle has not achieved the goal.
In [99] creator had exhibited a blame tolerant work for 3D Network-on-Chip. In 3D Mesh NoC, to maintain a strategic distance from deadlock states, we utilized the k most brief ways calculation dependent on the guideline of a most brief way issue. Given a coordinated diagram G with nonnegative edge loads, a positive whole number k, and two vertices s and t, the issue requests the k most brief ways from s to t in expanding request of length. With no circle or cycle (without deadlock), the k ways should be straightforward. The arrangement of k most limited ways was determined and embedded into the steering table. To course consequently, the way was set apart by the switch, and the following reinforcement way was chosen if there was a disappointment in the edge or hub of a way. Accordingly for the particular adaptation to internal failure rate, enough number of substitution ways was found effectively by setting the edge esteem utilizing the K-briefest ways calculation.
In [100] the author formulated a novel fully adaptive fault-tolerant routing algorithm for 3D NOCs. When a fault occurred, the proposed algorithm applied a fault detection scheme instead of rerouting the packets because our proposed system can get the fault information one hop away in advance. Because in the above method when a fault occurs, we have to reroute the packets around fault regions. But in our proposed system when doing the path computation itself the fault information was combined. In the 3D NoC architecture, this algorithm can deal with multi faults. When compared to other routing algorithms, our proposed system proved that it could achieve lower latency, energy consumption and higher packet arrival rate based on the simulation results.
In [101] author had proposed a fault-aware dynamic routing algorithm for on-chip networks. The proposed system has the capacity to handle with both static and dynamic permanent failures. Addition to this the stress factors were considered to issue the load over the entire network. In our dynamic routing algorithm, the packets were pushed close to the destination according to the relative positions of the current and the destination nodes. The address of the packet in the destination was checked by the router to decide the appropriate route for the packet in which the route was close to the destination and also to avoid faulty links. To reroute the packets around faulty links and to issue the network load to reduce the probability of congestion both the stress factor and the acknowledgment signal was integrated into our algorithm. The addresses of the current and the destination routers was assumed as (XD, YD) and (XR, YR) respectively. When the destination node was vertically or horizontally along the current router, then the number of preferred ports will be “1”. In two cases a link was masked. One was permanent faults, and the other was temporary

congestion. The average latency of the network was decreased by masking permanent faulty links because it saves the time for sending and receiving the packets on the links and also it avoids packet loss. Masking congested links were the same as masking permanent links also in addition by redirecting the network traffic from the hot spots; the total average latency was reduced.

In [102] author had developed a low overhead tolerant routing scheme for 3D Networks-on-Chip. In this paper, a novel fault-tolerant routing scheme (4NP-First) for 3D NoCs was designed. To create a robust hybrid routing scheme, the 4N-First and 4P-First turn models were integrated here. The advantage of the proposed system was the low implementation overhead. In the 4NP-First model, the dual virtual channel router architecture was described. A header flit and a destination ID is contained in all the packets. There are two input FIFO buffers dedicated to 4N-First routed packets and 4P-First routed packets respectively. The input channel was connected to an appropriate output channel using the switch when the flit was ready. Route compute control unit, virtual channel allocator and switch allocator are the three main control modules to control the data path. To determine the next hop direction, the next virtual channel and when a switch is available for each flit, these control modules were utilized. Route Computation (RC), Switch Allocation (SA), Switch Traversal (ST) and Virtual Channel Allocation (VCA) are the four phases in routing operation. To determine the next hop for the packet, using router the flit was stored in the buffer when the first flit of the packet was received at the input channel. If the packet was received successfully at the destination, then ACK signals must be sent from destination to source. In other cases due to faults if the packet was dropped, then NACK signal must be sent to the source through the intermediate node. In the proposed approach, after the NACK signal was received, the source can send the packet maximum two times. In this case, faults can be avoided.

In [103] author had implemented a fault-tolerant vertical link for 3D Networks-on-Chip. A scalable 3D NoC architecture was implemented. Here an accurate path was provided Through Silicon Vias (TSVs) to support vertical communication among different layers of a vertically stacked chip. In order to improve the global yield of 3D stacked chips, semi-automated design flow for 3D NoCs including a defect-tolerance scheme was presented in this paper. For vertical interconnections, a circuit-level model was extracted first. Then to calculate the design implications of extending switch architectures with ports in the vertical direction, this extracted circuit-level model was utilized. Then through effective use of redundancy, a defect-tolerance technique was presented for TSV-based multi-bit links. At last, a design flow allowing for post-layout simulation of NoCs with links in all three physical dimensions was presented. Based on post-manufacturing study and reconfiguration of the electrical resources, to leverage a small amount of on-chip spares a novel fault-tolerant dynamic routing approach was proposed here.

In [104] author had developed a reliable routing architecture and algorithm for NoCs. To avoid system failure due to permanent faults in NoC, a fault-tolerant architecture and companion routing protocol known as Vicis was proposed here. The proposed method was very robust (i.e., in case of permanent transistor failures, it allows communication to take place). There are two levels of approach. One was by controlling architectural components; it works around errors within the router. Next was a link's connectivity which was disabled due to faults; here the faulty node was rerouted by Vicis. To locate the number of hard faults, each router contains Built-In Self-Test (BIST) units. To protect against crossbar failures, it has a crossbar-bypass bus. To protect data path elements, it contains Error Correcting Codes (ECC) and to reduce internal faults, reconfigures the FIFO buffers. To run a distributed in-hardware network reconfiguration algorithm, the routers work together and thus bypass the broken links and routers.
In [105] Wang et al. described a scalable load balancing congestion-aware Network-on-Chip router architecture. A novel scalable load balancing congestion-aware NoC architecture was introduced here. In this architecture, the overall network throughput was improved, and also feasible implementation cost was maintained. To balance global traffic load distribution, a combination of dynamic input arbitration and adaptive routing path selection was contained in this congestion control scheme. To prevent packets from routing through defected areas, faulty links are transmitted by existing congestion management control signals. Here the input ports of a router were observed, so that network congestion can be identified in our proposed Congestion Aware (CA) router design. The number of blocked input ports (not routed by the router) was used as the congestion index. Two congestion indices CE and CW are needed because here each router was divided into two sub-routers. Through dedicated signal wires, CE and CW were sent to the neighboring routers. To process destination information from the header flit, Header Parsing Unit (HPU) was used. To decide routing path, perform arbitration and to manage the crossbar switch, Arbiter Logic (AL) was used. The transmission of packets from input FIFOs to output links was done by the Switching unit. At last, to calculate the number of non-serviced flits, Congestion Index (CI) was used.
Paul Gratzet et al. [106] had proposed regional congestion awareness for load balance in NoCs. The limitations of conventional adaptive routers can be defeated by introducing a method called Regional Congestion Awareness (RCA). Congestion information from different points in the network was integrated into the port selection process using the RCA technique. The metrics of congestion throughout the network were collected and broadcasted by a light-weight monitoring network. This results in a better picture of network hotspots for each router. For communication, centralized tables in RCA were not needed. Instead to spread congestion information among adjacent routers a low-bandwidth monitoring network was used. Local congestion estimate was

aggregated with the neighboring nodes by the router at each network hop. For port pre-selection, the new congestion estimate was utilized, and it was broadcasted over the upstream. Based on the distance from the current node, the contention information was weighted by the aggregation step to reduce the negative effects of staleness and to avoid interference from non-minimal paths.

In [120] author used Hamiltonian path strategy for deadlock and livelock handling as well as fault-tolerant mechanism. This algorithm guarantees minimal path due to the hamiltonian path and data packet reaches to the IP core in ascending or descending order. HamPA does not require any blame conveyance, extra blame data, or any virtual channels. They adjusted the essential type of the Hamiltonian way so as to probably switch among high and low channel subnetworks. In this manner, HamPA is without deadlock since no unique cycle will be shaped. Both the flat and vertical connection issues are considered. The constraint of this calculation is that the broken connections ought to have a place with the equivalent subnetwork so as to endure numerous vertical or even one-flawed connections. Deficiencies at some exceptional positions are not upheld by HamFA. When these deficiencies happen, the entire NoCs can't work ordinarily.

In [121] creator proposed AFRA: A minimal effort elite dependable directing for 3D-work NoCs. This calculation courses bundles through ZXY without connection issues. At the point when shortcomings are identified, the calculation changes the steering to XZXY: The dances are sent to a departure hub through X, and afterward they proceed to their goal through ZXY for what it's worth in the no-blame case. The creators introduced straightforwardness, great execution, and power as the principle highlights of this calculation. The real constraint of the calculation is that it endures blames just in vertical connections and level connections are not mulled over. Furthermore, the system blockage status isn't mulled over while steering. A blame conveyance component is required to think about the blame data on every single vertical connection along each column.

In [122] author proposed HARS: A high-performance reliable routing scheme for 3D NoCs. In the algorithm, a mid-node at each layer is used to transmit the data between two nodes. Author has adopted partly adaptive routing DyAD. It has an advantage of Dimension order routing and adaptive routing with odd-even turn model.

In [125] the author has given a blame tolerant ability to Networks-on-Chip (NoC) by means of an effective steering way choice system utilizing a novel versatile directing calculation, i.e., Efficient Dynamic Adaptive Routing (EDAR) is proposed. It depends on a weighted way determination methodology, which abuses the status of continuous NoC traffic made accessible through screen modules. The key execution objective is to keep up throughput under blocked and flawed conditions by means of compelling steering way choices. In the proposed EDAR, port loads are determined continuously as indicated by the channel status - Idle/Busy/Congested/Faulty, and the port with the most minimal weighting is positioned as the close ideal course to advance parcels. This system empowers the switch to sidestep blocked ports and endure broken ports. To survey the inactivity and throughput of the proposed steering calculation, a few traffic designs for both blame free and broken NoCs were assessed. Results demonstrate that EDAR can accomplish higher throughput contrasted with another best in class steering calculations under different traffic examples and dimensions of infused flaws. Likewise, the equipment territory overhead for EDAR is shown to have a sensibly minimal effort which keeps up versatility for substantial NoC executions is avoided

4.5 CONCLUSION

In this chapter, some background work already done for 2D NoC routing algorithms, 3D architectures, and 3D NoC routing algorithm is presented. The fundamental test with NoC steering is to expand the system unwavering quality while guaranteeing extensive execution. The steering calculations gave have significantly managed criteria like deadlock, livelock, clog and adaptation to non-critical failure. The explanation behind choosing these arrangement criteria is on the grounds that the arrangements proposed to accomplish these objectives are straightforwardly connected to organize execution. The primary execution parameters assessed in practically all the work introduced are low inactivity, high throughput, and low power utilization. The examination of this calculation demonstrates that each work evaluates the measurements specifically identified with the goal considered. The deadlock techniques are in this way assessed by their normal traffic throughput in different rush hour gridlock designs while inertness is the most assessed parameter for clog and adaptation to internal failure. We likewise noticed that the vast majority of the deadlock and livelock calculations are either the improvement or alteration of the two prevalent XY and Turn calculations. This is on the grounds that they are anything but difficult to execute and keep up high unwavering quality. Every one of them, obviously, proposes another thought, yet they share a couple of properties, for example, the utilization of wormhole exchanging procedures, to decrease support space. The quantity of calculations displayed in this section demonstrates that the NoC directing is broadly examined in view of the extraordinary significance of information transmission. All through this examination it is seen that the deadlock and livelock challenge is the most considered, the greater part of the calculations concern the issue of the deadlock and livelock because of the unsafe impacts it causes on the NoC. The fault and congestion challenges are second important issue concerned in the papers. The literature presents a number of solutions to avoid or eliminate these routing problems. This chapter discussed the important routing issues in NoCs and presents many routing algorithms for 2D NoC and 3D NoC . The algorithms presented are mostly implemented on

2D-mesh and 3D-mesh based network and used a wormhole switching technique. This solution for routing issues are summarized in table 4.1.

Table 4.1: Summary of solutions discussed in the literature survey

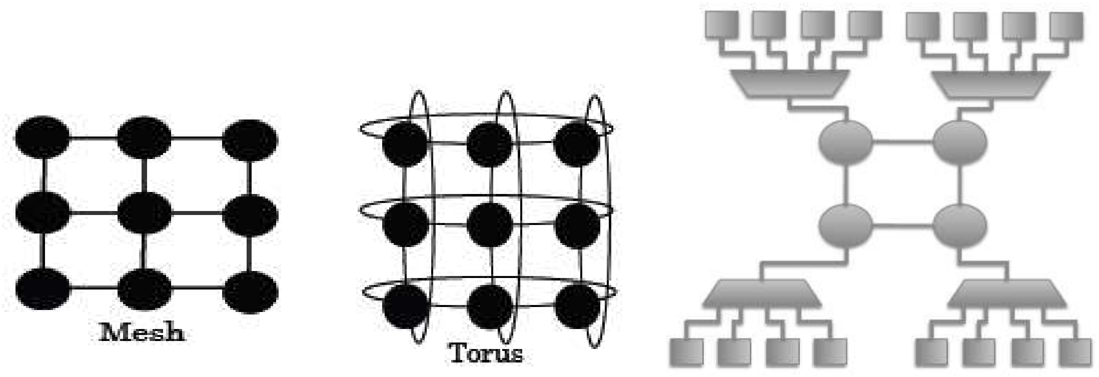
Proposed Solution	**Routing Challenges**	**Limitation**
Deterministic algorithm[71,75,78,121]	Deadlock, Livelock	They cause congestion, a the path is fixed
Cyclic dependency graph[120]	Deadlock	Graph-based solution needs proof
Virtual channels[77,81,102,121]	Deadlock, livelock	Buffer space is needed more
Prohibited turns[90,91]	Deadlock, livelock, Congestion	Few links lead to the lower path diversity
Minimal path[97,99]	Congestion	Alternative path is not available
Routing tables[101,103,104,105,122]	Congestion, Failures	Buffer space is required and suffers from the non-scalability issue
Q-learning[87,89,98]	Congestion	Extra buffer space is required
ACO[86]	Congestion	Energy consumption is much higher
Adaptive routing[79,93,94,100]	Congestion, deadlock, failures, livelock	Causes high latency
Packet duplication	failure	Buffer space requirement & causes congestion

The solutions presented in table 4.1 are appropriate for one issue than other. In order to prevent deadlock, most research uses virtual channels, deterministic strategies while it is recommended to use minimal paths, deterministic routing or limit the number of hops to avoid livelocks. Next for handling congestion adaptive routing, turn models, prohibited turns are adopted. Finally, the solutions proposed to prevent failures are adaptive strategies, routing tables, reuse/alternative buffers, re-routing in defective regions and the use of ACO in certain papers.”

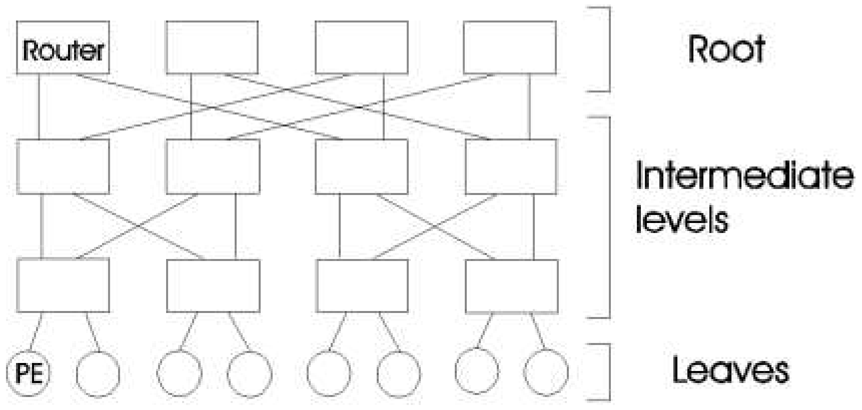
CHAPTER 5. ANALYSIS OF 2D TOPOLOGIES AND ALGORITHMS

5.1 BACKGROUND

Networks on the chip are the current edge technology for on-chip communication architectures which were based on shared bus and pointed to point interconnections. NoC’s are the shift from computation centric to communication centric and provides optimum performance in terms of bandwidth, throughput, and scalability as compared to traditional System-on-Chip. In this chapter, we have evaluated the four basic topologies mesh, torus, fattree, and cmesh for 2D NoC as shown in figure 5.1.



Cmesh (4 cores attached to 1 router)



Fat-tree

Fig. 5.1: Basic topologies for Network-on-Chip

Many topologies have been proposed for NoCs so far, such as Mesh [6], Torus [78], CMesh and Fat-tree. Mesh has grid based (M*N) M columns and N rows layout and has gained more consideration by designers due to its simple design. The IP cores and associated routers can easily be identified as (x,y) coordinates. It is also known as Manhattan street address.

Torus is an improved version of mesh as routers on the edges are connected to the routers on the opposite edges through wrap around edges. The diameter of the mesh topology is more thus communication latency also got increased, whereas torus topology has the same design as mesh, but due to wrap around edges the path diversity is more. The routes identified are also minimal as compare to Mesh.

In both of the topologies, PE/IP cores are places in an n-dimensional layout, and the packet moves in one dimension at a time to reach to the destination. These layout and design are the ones that prove out to be the best tradeoff between cost and performance parameters, and also achieve good scalability.

Concentrated Mesh (CMesh): This is the novel and efficient topology for optimizing the performance of NoC. CMesh is topology in which multiple nodes share the same router. In topologies like mesh and torus, each IP core along with associated router consumes more area whereas in CMesh it consumes lesser space. In the given table 5.1, each router is providing services to the four IP cores. Therefore neighbor IP cores communicate data with a single router. Due to the less number of routers, we can have routers with larger buffer space and also less wiring is required for interconnections. The advantage of this architecture is the distance across the network is considerably low and permits lower hop count thus reducing latency.

Table 5.1: Topology parameters for mesh and torus

Topology Type	**No of Nodes**	**Node Degree**	**Diameter**	**Routers Required**	**Network Cost**
Mesh (M*N)	M*N	Boundary =3, Centre =4	M+N-2	M*N	O(N)
Torus (M*N)	M*N	Boundary =4, Centre =5	M/2 +N/2	M*N	O(N)
Fat Tree	N = kn k = no of children's n = level of tree	Boundary & leaves = 2 Intermediate nodes = 4	2n	$K^{n-1}*$ n	O(Nn)

Fat-tree Topology: In Fat-tree topology routers and IP cores are arranged in a hierarchy where the upper nodes and intermediate nodes are routers and leaves are IP cores. The number of routers required is dependent on the number of IP cores. As compare to meshes and torus, Fat-tree is non-cyclic thus do not suffer from deadlock situations.

5.2 SIMULATOR USED: BOOKSIM 2.0

In our work, Booksim 2.0 simulator shown in figure 5.2 is used for modeling the NoC architectures. Booksim 2.0 simulator is a detailed cycle-accurate simulator for 2D NoC environment. It is implemented in C++ and has been specifically tested with the GNU G++ compiler.

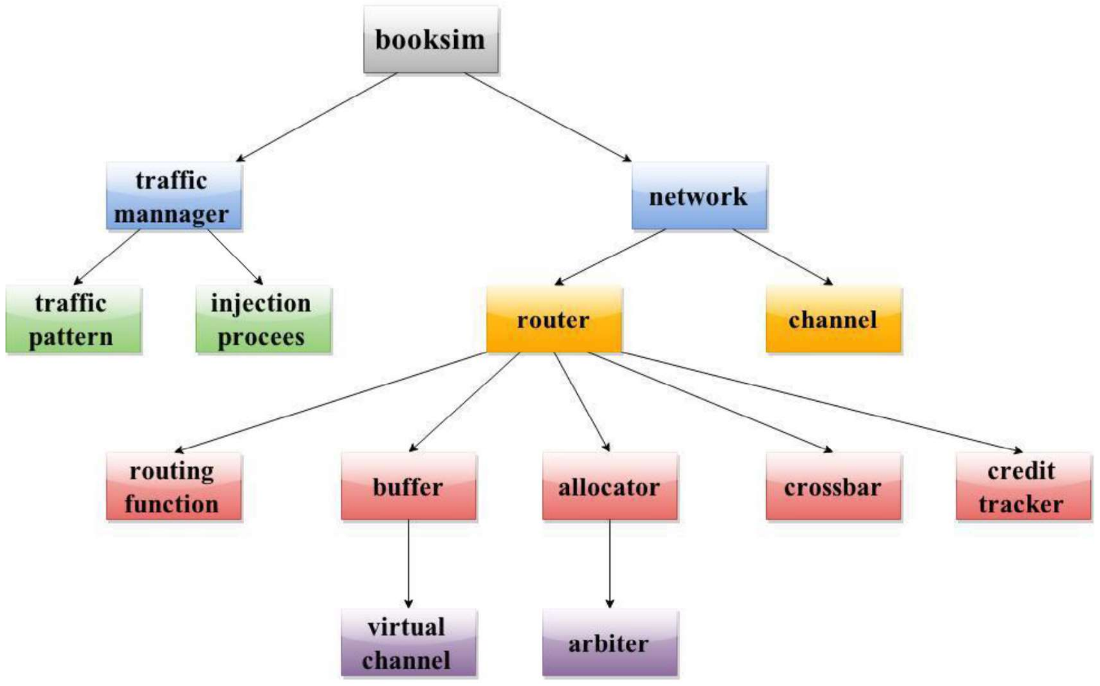


Fig. 5.2: Booksim 2.0 Simulator

It can be used to simulate all the prime aspects including topologies, routing, and flow control, router architecture and evaluate the performance parameters. Booksim 2.0 supports the following type of traffic:

- Uniform traffic
- Random
- Hotspot
- Bit-reversed traffic
- Bit-complement traffic
- Tornado traffic
- Neighbor traffic

5.3 ANALYSIS OF 2D MESH, TORUS, CMESH AND FATTREE TOPOLOGIES

We have analyzed the impact of injection rate on the parameters latency and throughput for the four basic topologies i.e. Torus, Mesh, Fat-tree, and CMesh. The various simulation parameters used are shown in table 5.2.

Table 5.2: Parameters used for Booksim 2.0 simulation

Name of Parameter	Value of Parameter
Number of Nodes	64
Routing Algorithm	DOR Routing (XY)
Packet Size	20 Bytes

Flit size	32 bit
Traffic Pattern	Uniform
Virtual Channels	8
Number of Buffers	8
Simulation type	Latency and throughput against injection rate

With varying injection rate, the latency and throughput are simulated for all four topologies. The algorithm used for this analysis is dimension order routing. The results obtained are the average values obtained after running of 30 cycles of Booksim 2.0 which are shown in table 5.3 and figure 5.3.

Table 5.3: Latency comparison of topologies (XY algorithm, 2D NoC)

Injection Rate	**Latency (Mesh)**	**Latency (Torus)**	**Latency (CMesh)**	**Latency (Fat-tree)**
0.1	8.16688	6.09203	11.1967	23.22
0.2	8.36785	6.12798	11.3399	23.2502
0.25	8.57567	6.22626	11.5578	23.4255
0.3	8.68573	6.29864	11.921	23.5276
0.35	8.8553	6.36982	12.612	23.7092
0.4	9.12075	6.48916	14.2959	23.8873
0.45	9.39171	6.58926	35.6894	24.0946
0.5	9.72269	6.75767		24.3976
0.55	10.1946	7.00264		24.7219
0.6	10.8985	7.26118		25.2352
0.65	11.6745	7.66861		25.8901
0.7	13.0366	8.21282		26.9072
0.75	15.243	9.11454		28.2196
0.8	18.7931	10.6792		31.537
0.85		14.5583		39.1369

Dimension Order Routing: In [111] author has proposed a very popular XY routing algorithm for NoC. Packets have header flit, tail flit, and data flit. Header Flit will contain the destination address. Each IP core can be identified through (X,Y) coordinates. The source IP will have (Xsrc, Ysrc) coordinates, current IP will be identified as (Xcurr, Ycurr) coordinates and destination IP will be identified as (Xdest, Ydest).

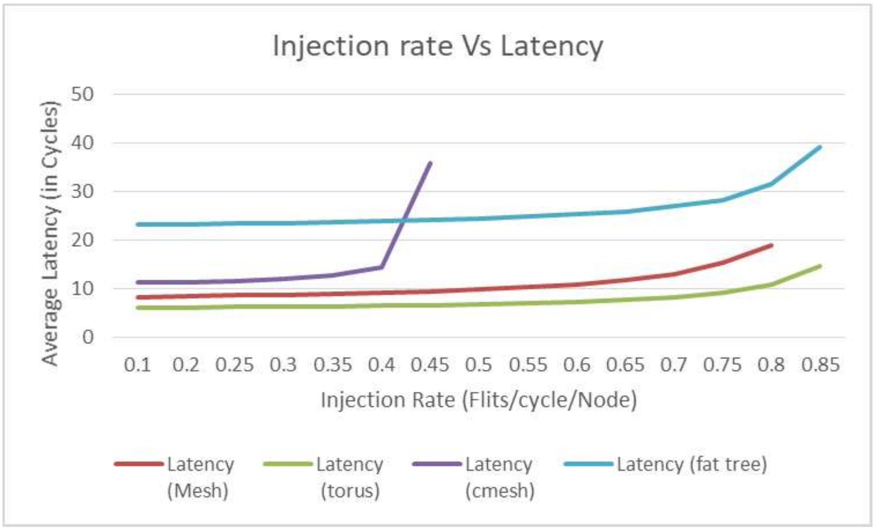


Fig. 5.3: Injection rate vs. latency (XY algorithm, 2D NoC)

The detailed algorithm is presented below and implemented in Booksim 2.0 for evaluating the Mesh, Torus, Fat-tree, and CMesh topology.

XY Algorithm

```
Source IP:  (Xsrc, Ysrc);
Destination IP: (Xdest,Ydest);
Current IP: ( Xcurr, Ycurr)
```

```
Begin
    If ( Xdest > Xcurr) then Moveto(East );
    Elseif(Xdest < Xcurr) then Moveto(West);
            Else (Xdest = Xcurr) Return (Xcurr);
    /* Current column and destination column is same check Y coordinate */
    If ( Ydest > Ycurr) then Moveto(North);
    Elseif(Ydest < Ycurr) then Moveto(south);
            Elseif(Ydest = Ycurr) then Return(Ycurr) ;
            /* Reached to the destination IP core */
End;
```

5.3.1 LATENCY COMPARISON

Latency can be defined as the time in the clock cycles or nanoseconds for transferring a packet from the source node to the destination node. Latency does not only include traversing time but buffering, virtual channel allocation and handling, routing decisions switching, arbitration and congestion delays also adds to latency.

5.3.2 THROUGHPUT COMPARISON

Table 5.4: Throughput comparison of topologies (XY algorithm, 2D NoC)

Injection rate	throughput (Mesh)	throughput (torus)	Throughput (c mesh)	Throughput (fat tree)
0.1	0.009941	0.101265	0.0993333	0.0994167
0.2	0.201435	0.201753	0.199437	0.199687
0.25	0.252303	0.252425	0.25	0.25
0.3	0.302149	0.302415	0.299667	0.3
0.35	0.352145	0.353373	0.348479	0.348971
0.4	0.402233	0.40228	0.398188	0.398417
0.45	0.452269	0.452514	0.447271	0.45
0.5	0.501804	0.502299		0.499063
0.55	0.552312	0.552439		0.548708
0.6	0.601104	0.601128		0.598896
0.65	0.650265	0.651037		0.64875
0.7	0.699069	0.700533		0.699042
0.75	0.749401	0.750474		0.749708
0.8	0.79271	0.800443		0.799687
0.85	0.47117	0.849584		0.851771

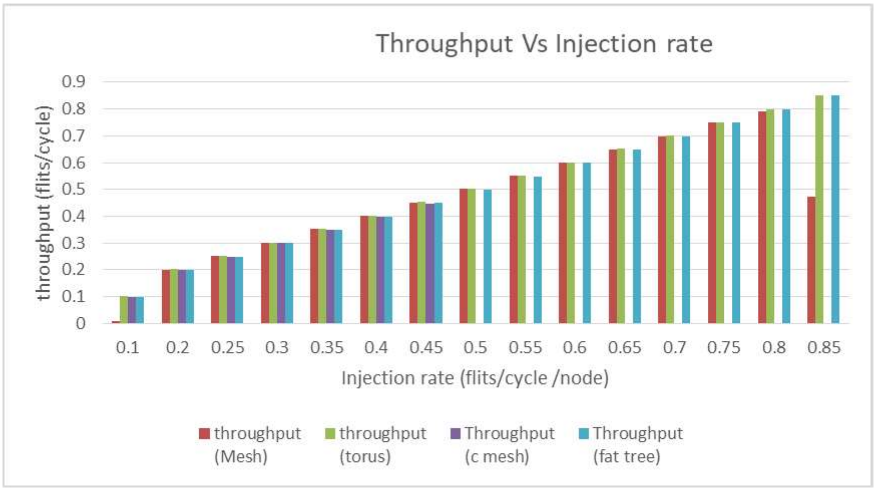


Fig. 5.4: Injection rate vs. throughput (XY algorithm, 2D NoC)

5.3.3 NUMBER OF HOPS COMPARISON

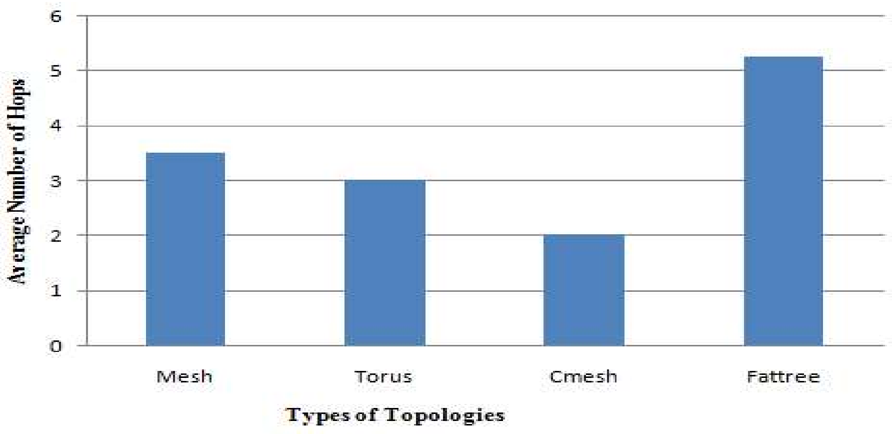


Fig 5.5: Hops comparison for 2D NoC

5.3.4 VARIATION OF THROUGHPUT WITH NUMBER OF VC UNDER UNIFORM TRAFFIC

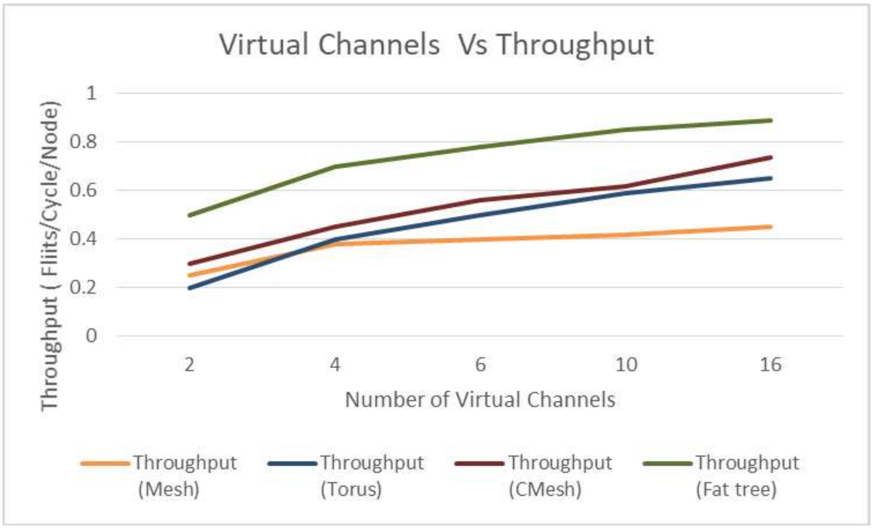


Fig. 5.6: Virtual channels and throughput for 2D NoC

5.3.5 RESULT ANALYSIS OF XY ALGORITHM

We have investigated the four NoC topologies (Mesh, Torus, CMesh and Fat-tree) support for communication under dimension order routing strategies which is the commonly adopted algorithm for NoC where the flits from the source node traverses towards the X (horizontal link) nodes up to the column of the destination nodes, then along the Y (vertical link) nodes up to the destination node. The first set of data shown in table 5.3 and graphically represented in figure 5.3 shows simulation based latency values of all the four topologies with respect to injection rates. Latency range increases with the increasing values of injection rate. There is the difference in the simulation values of the topologies due to stochastic variability of the simulation environment. The presented values and graphs confirm that Torus performs better among all the evaluated topologies in terms of latency and it is minimum followed by mesh topology. The latency values of CMesh and Fat-tree are very high.

Table 5.4 and figure 5.4 shows the throughput values of the four NoC architectures with respect to the varying range of injection rate. The values presented to depict the highest throughput for torus followed by mesh architecture. Infect there is a marginal difference in the throughput values of mesh and torus. CMesh and Fat-tree present poor performance in terms of both latency and throughput.

There are no values shown both in latency table (table 5.3) and throughput table (table 5.4) for CMesh architecture after particular injection rate value. CMesh topology does not support high injection rate range. The reason for this behavior is saturating injection rate of CMesh architecture. The bottleneck here is the limited buffering space of CMesh architecture.

The graph presented in figure 5.5 shows the average number of hops for all the topologies. The average number of hops is almost equal for mesh and torus. CMesh outperforms all other topologies and requires minimum hops for sending the packet from source to destination.

Figure 5.6 presents the impact of virtual channels on throughput. Usually, wormhole technique is used as a flow control mechanism for on-chip communication, but as the network load increases, it leads to blocking that can be solved by virtual channel. As we increase the number of virtual channels, throughput values get increased.

After this comparative study, it can be analyzed that torus and mesh can be the good choice for NoC architecture as both perform almost the same. Both topologies have scalable and symmetric architecture. The torus will have additional wiring complexity and more power consumption due to the wrap around edges.

5.4 ANALYSIS OF ODD EVEN ALGORITHM AND ADAPTIVE ALGORITHM

The odd-even routing algorithm [37] denotes columns as odd or even as per the column number. The first column number is assumed as zero hence it is even, then column number 1 is odd, column number 2 is even and so on. The algorithm prevents some turns depending on the column in which the packet is in. If the packet is an even column, it cannot turn north or south from the west. If the packet is an the odd column, the packet cannot move towards east. The algorithm does not suffer from challenges like deadlock and livelock. Adaptive algorithm [30] combines both approaches of XY and odd-even. The advantages of adaptive routing algorithms are to avoid congested path in the network and to have high throughput. The detailed algorithms are present below and implemented through Booksim 2.0 simulator. Similar to the XY algorithm, latency and throughput values are evaluated against the injection rate for Mesh-based NoC.

Odd Even Algorithm:

```
Input:
    Source Node = X_Src, Y_Src
    Destination Node = X_dest, Y_dest
    Current Node =  X_curr, Y_curr
    Temp_X = X_dest - X_Src
    Temp_Y = Y_dest –Y_Src
    New_Directions – It is an array containing all routing destination
```

```
Begin
    New_Directions = Null;
    If (Temp_X and Temp_Y = 0)
        choose local node for routing
    If (Temp_X = 0), It means packets are in destination column
        compute the direction North or South
        New_Directions = New_Directions U North
        or New_Directions = New_Directions U South
    If (Temp_X > 0)
        check Temp_Y = 0
        add New_directions = New_directions U East
    Else
        X_curr is odd or X_curr = X_dest
    If the above condition is satisfied then
        check Temp_Y, accordingly
        compute the direction North or South
        New_Directions = New_Directions U North
        or New _Directions = New_Directions U South
    If X_dest is odd or Temp_X ! = 1
        New_directions = New_Directions U East
    If (Temp_X <= 0)
        New_directions = New_Directions U West
    If X_Curr is even and Temp_Y < 0
        New_directions = New_Directions U North
        select the dimension from New_directions to forward the packet.
End
```

Adaptive Algorithm:

```
Begin
    If (CONGESTION = 0)
        Deterministic routing algorithm (XY Routing)
    Else
        Adaptive routing algorithm (OE Routing)
End
```

5.4.1 LATENCY ANALYSIS OF ODD-EVEN AND ADAPTIVE ALGORITHM

Table 5.5: Latency values for odd-even and adaptive algorithm (2D mesh topology)

Injection Rate	Latency (Odd-Even)	Latency (Adaptive)
0.1	7.16688	6.15588
0.2	7.36785	6.37685
0.25	7.57567	6.4657
0.3	7.68573	6.68573
0.35	7.8553	6.75531
0.4	8.12075	7.42075
0.45	8.39171	7.50171
0.5	8.72269	7.72269
0.55	9.1946	8.1946
0.6	9.8985	8.8985
0.65	10.6745	9.6745

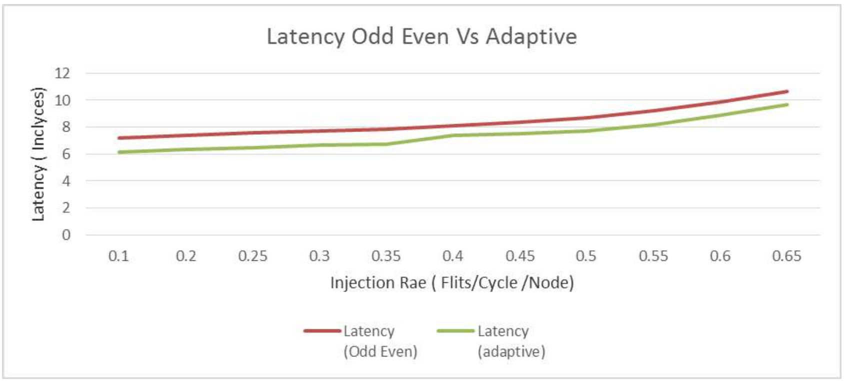


Fig. 5.7: Latency values and Injection rate for odd-even and adaptive algorithm (2D mesh topology)

5.4.2 THROUGHPUT ANALYSIS OF ODD-EVEN AND ADAPTIVE ALGORITHM

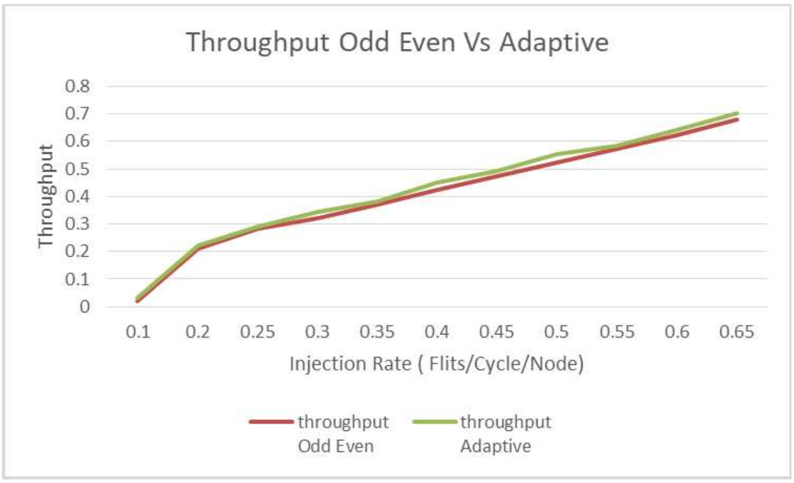


Fig. 5.8: Throughput values and injection rate for odd-even and adaptive algorithm (2D mesh topology)

Table 5.6: Throughput values for odd-even and adaptive algorithm (2D mesh topology)

Injection rate	**Throughput Odd-Even**	**Throughput Adaptive**
0.1	0.019941	0.029941
0.2	0.211435	0.221435
0.25	0.282303	0.292303
0.3	0.322149	0.342149
0.35	0.372145	0.382145
0.4	0.422233	0.452233
0.45	0.472269	0.492269
0.5	0.521804	0.551804
0.55	0.572312	0.582312
0.6	0.621104	0.641104
0.65	0.680265	0.700265

5.4.3 RESULT ANALYSIS OF XY, ODD-EVEN, AND ADAPTIVE

The latency and throughput values of the odd-even and adaptive algorithm are presented in table 5.5 & 5.6 and graph 5.7 & 5.8. Performance metrics (P) are the ratio of average latency to average throughput. It is one of the important parameters for evaluating routing algorithm performance.

Performance Metrics (P) = Average Throughput /Average Latency

Table 5.7: Performance metrics of XY, odd-even and adaptive algorithm for 2D NoC

Algorithm	**Average Throughput**	**Average Latency**	**P**
XY	0.38890	9.42310	0.0412709
Odd Even	0.408905	8.42310	0.0485456
Adaptive	0.426178	7.44111	0.0572734

The OE routing algorithm has idleness not exactly the XY routing algorithm for all infusion rates. From tables 5.3, 5.4, 5.5 and 5.6 and diagram 5.3, 5.4, 5.7 and 5.8, the total strength of OE over the XY routing algorithm can be seen. Be that as it may, the DyAD routing algorithm dependably surpasses XY and OE routing if there should arise an occurrence of inertness and throughput. From Performance measurements table 5.7, the DyAD routing algorithm is more proficient than both XY and OE routing. The execution measurements demonstrated the OE routing algorithm is superior to anything the XY routing algorithm and that the DyAD routing algorithm is superior to the XY and OE routing algorithms as far as execution, i.e., throughput.

CHAPTER 6. ANALYSIS OF 3D NOC ROUTING ALGORITHMS

6.1 BACKGROUND

NoC was introduced as a simple and scalable communication architecture that connects processors, IP cores and other custom designs together using packet switching methodology that transmits data on a hop-by-hop basis and provides higher bandwidth and higher performance. At the same moment, future applications are getting increasingly mind-bogglers and complex, that quest for an adaptable design to guarantee the data transmission efficiency on the small chip. This has made regular 2D NoC not reasonable enough for future extensive large scale systems. Since the processing industry moves quickly from a single core with high-frequency designs to multi-core chips. The integration of three-dimensional (3D) rather than two-dimensional (2D) is another trend. 3D technologies have created significant opportunities and challenges in designing interconnection networks with low latency, low power, and high bandwidth.

This Chapter will discuss the reasons behind the adoption of 3D NoC, the 3D topologies, and advantages of 3D NoC. This chapter will present the analytical results of XYZ, adaptive and negative first routing algorithms.

6.2 TRANSITION TO 3D NOC FROM 2D NOC

The items, for example, cell phones, note pads, and individual handheld sets are enhanced practically as well as turns out to be quicker, littler in-measure, bigger in-limit, lighter-in-weight, lower-in-control utilization and less expensive. Because of this, 2D NoC interconnect are not a reasonable contender for future extensive scale multicore SoCs which will oblige several centers. Specifically, the impediment of the 2D NoC worldview is because of the high distance across of the regular 2D NoC. The width of the system is the quantity of jumps a bounce goes to reach to the goal from the source. In 2D NoC, if a specific bundle goes through countless to its goal, the correspondence time (inactivity) is long, and the throughput is thusly low. At the end of the day, the expanded system width negatively affects the dormancy of the directing framework. Thus the quest to optimize 2D NoC based architecture is becoming increasingly necessary, and a lot of research has been carried out to achieve this goal by using different approaches such as implementing fast routers [112], modifying the topologies [20] or optimizing the routing approaches. Thus some advance architecture is required to ensure sufficient bandwidth for any transactions and communication between different IP cores on the same chip.

This has led the semiconductor industry to consider other ways of increasing integration on the chip. One of the solutions proposed was to move the 2D NoC architecture to the third dimension

[113,114]. 3D integration quickly emerges as a viable option to continue the exponential trend [115]. Today's Integrated Circuits (ICs) are “2D” and place all devices in one layer – in contrast, a 3D IC stacks multiple layers of active devices. These stacks are vertically interconnected via silicon vias (TSVs), emerge as a promising technology for SoCs. Compared to 2D designs, 3D integrated circuits allow for reduced latency for critical interconnecting structures, resulting in higher system throughput, performance, and power, and enable heterogeneous integration as well [116]. Three-dimensional Network-on-Chips (3D NoCs) combines the benefits of short vertical interconnects of 3D ICs and the scalability of NoCs. 3D NoCs support each horizontal and vertical links.

Till now, the research in this area has shown that 3D ICs can achieve higher packaging density by adding a third dimension to the conventional two-dimensional design [117]. The high performance is achieved due to the reduced interconnect wire length [118]. The power consumption can be lower down because of this shorten wiring. 3D ICs also make circuitry more noise resistant and allow the mixed technology to be implemented. The NoC structure and 3D integration offer a promising 3D NoC architecture. This combination gives NoC designs a new horizon to meet the high demands of future large-scale applications. In [53] author has presented and proved that 3D NoC has the capability of reducing latency and the energy per packet by decreasing the number of hopes by 40% which is one of the important characteristics to evaluate the system performance.

6.3 3D NOC TOPOLOGY

The most common 3D mesh based interconnection structure is presented in figure 6.1. We have presented an extensive literature survey on 3D NoC architectures in our chapter three. We have also presented an analysis of 2D Mesh topology in chapter four. Based on the literature survey and empirical analysis of mesh topology it can be concluded that mesh-based architectures are most preferred topology for implementing NoC structures. Each node is connected to 2N of its neighboring node in the N-dimensional mesh network. The degree of a non-boundary node in the N-dimensional mesh is therefore 2N. The number of physical connections per node is fixed in a mesh network even if the network size increases. The diameter of the 3D mesh can be defined as D=d(k-1) where d is the dimension and k is the number of nodes on a single plane. The link is estimated by using the formula

$$link = n_1 n_2 (n_3 - 1) + n_1 n_3 (n_2 - 1) + n_2 n_3 (n_1 - 1)$$

Where $n_1, n_2\ and\ n_3$ are the respective x, y and z-axis. The link is estimated in order to determine the faulty nodes previously which are of two categories where some faults are permanent, and the rest are transient.

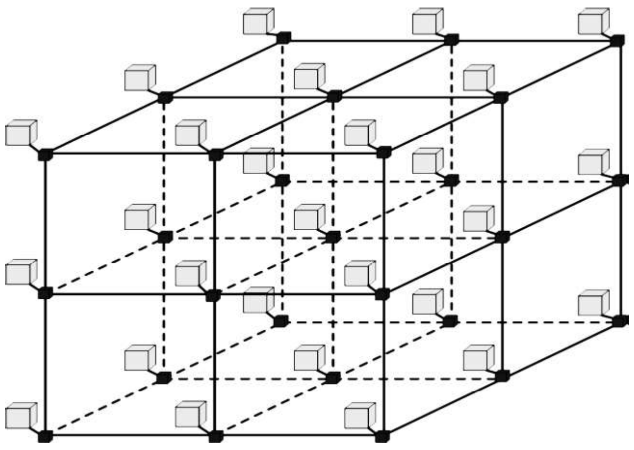


Fig 6.1: 3D mesh based topology

6.4 EMPIRICAL INVESTIGATION OF 3D MESH BASED NOC STRUCTURE

In this section, we will investigate 3D mesh NoC structure and evaluate its performance. In order to analyze the 3D mesh NoC, Access Noxim [13] is used. The performance of the Network-on-Chip is analyzed by its performance parameters. In this chapter, we have compared the performance of some of the existing routing algorithms in terms of throughput, average latency. The usage of a powerful routing algorithm while there is countless routing algorithms are one of the vital strides in structuring 3D NoC. Wormhole exchanging is utilized as an information transport instrument where the information packets are isolated into settled length units known as flutters. We have performed reproductions of 144-node and 256-node 3D mesh architectures and thought about their execution. For a scope of infusion rates inside the reproduction time frame, the normal idleness esteems and throughput are determined. The default parameters set as follows and shown in table 6.1 for conducting simulation with Access Noxim.

Table 6.1: Default parameters for Access Noxim

Parameter Name	Values Taken
Dim X, Dim Y, Dim Z (Mesh Size)	6*6*4 and 8*8*4
Buffer size (in flits)	Default 16
Simulation run	10000 cycles
Packet size	2-10 flits
Traffic Pattern	Random
Routing Algorithm	XYZ ,Adaptive algorithm
Parameters	Latency , throughput & Injection rate
Topology supported	3D Mesh
Traffic Pattern	Random

6.4.1 INVESTIGATION OF XYZ ALGORITHM

The Dimension Order Routing (DOR) XYZ algorithm is one of the well-known and well used routing designs in 3D NoCs [93]. XYZ is an easy to implement and deadlock free algorithm. The values for throughput and latency against injection rate are presented in table 6.2 and 6.3. The analytical graphs are shown in figure 6.2 and 6.3. The pseudocode of the algorithm as follows:

XYZ Algorithm:

Input:

```
    Source IP: X-src, Y-src, Z-src;
    Destination IP: X-dest, Y-dest, Z-dest;
    Current IP: X-curr, Y-curr, Z-curr;
    Initialize: X-curr = X-src; Y-curr = Y-src; Z-Curr = Z-src;
```

Start

```
        if (X-curr = X-dest && Y-curr = Y-dest && Z-curr = Z-dest)
                deliver() the packet to local node;
                exit()
        /*X routing */
        if (X-Curr < X-Dest )
                do
                        moveto(X-curr +1, Y-curr, Z-curr)
                while (X-curr = X-dest );
        else
                do
                        moveto (X-curr -1, Y-curr, Z-curr)
                while (X-curr = X-dest);
        if (Y-Curr < Y-Dest )
                do
                        moveto(X-curr, Y-curr +1, Z-curr)
                while (Y-curr = Y-dest);
        else
                do
                        moveto (X-curr, Y-curr -1, Z-curr)
                while (Y-curr = Y-dest)
        if (Z-Curr < Z-Dest)
                do
                        moveto(X-Curr, Y-curr , Z-curr +1)
                while (Z-curr = Z-dest);
        else
                do
                        moveto (X-curr, Y-curr, Z-curr-1)
                while (Z-curr =Z-dest);
End;
```

Table 6.2: Injection rate vs. throughput (XYZ algorithm)

Injection Rate	Throughput (6*6*4)	Throughput (8*8*4)
0.01	0.0798705	0.0794013
0.02	0.157087	0.158455
0.03	0.235071	0.234774
0.04	0.309366	0.0193707
0.05	0.0178685	0.00817582
0.06	0.0127124	0.00608811
0.07	0.00861753	0.00552243
0.08	0.00783967	0.00471543
0.09	0.00870087	0.00426605
0.1	0.00712293	0.00379332

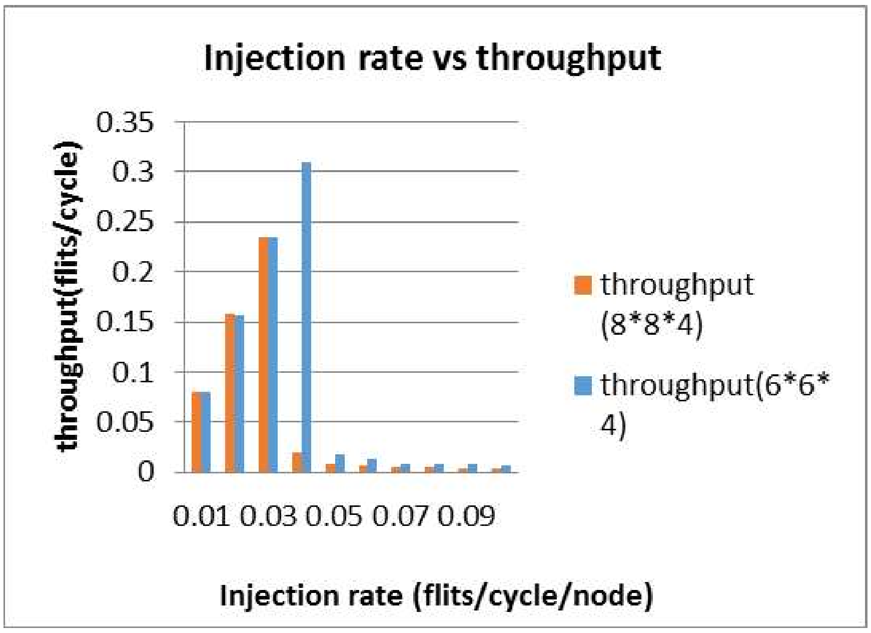


Fig. 6.2: Injection rate vs. throughput (XYZ algorithm)

Table 6.3: Injection rate vs. latency (XYZ algorithm)

Injection Rate	**Latency 6*6*4**	**Latency 8*8*4**
0.01	48	52
0.02	72	110
0.03	118	193
0.04	170	370
0.05	256	270
0.06	216	220
0.07	201	212
0.08	244	202
0.09	254	245
0.1	184	219

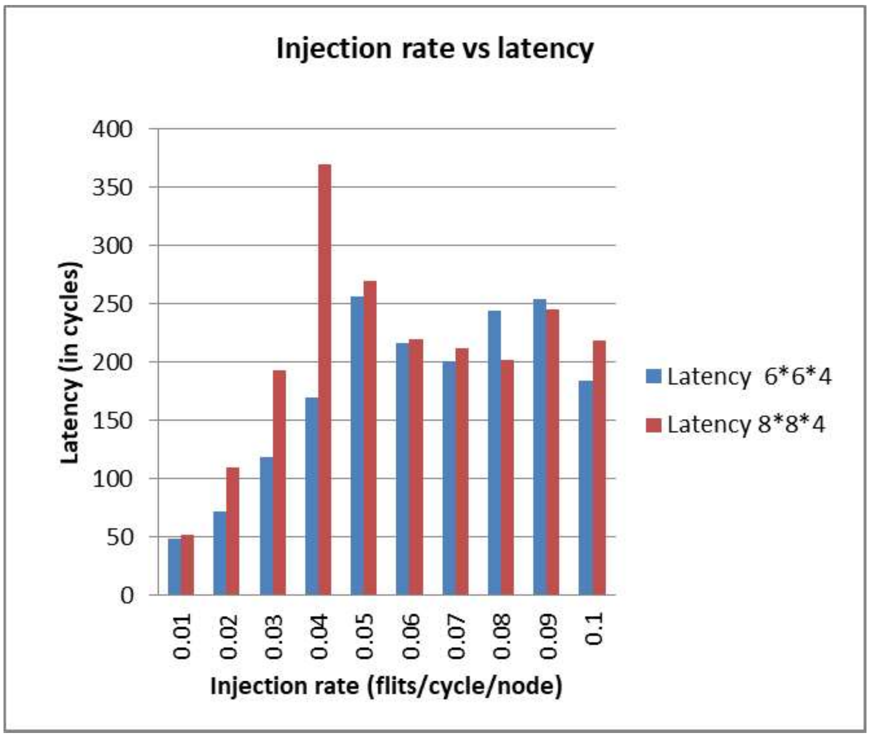


Fig. 6.3: Injection rate vs. latency (XYZ algorithm)

6.5 INVESTIGATION OF DY-XYZ (ADAPTIVE) ALGORITHM [72]

In [72] author had developed a fully adaptive routing algorithm for 3D NoCs. To transmit packets from source to destination node without affecting the performance, a fully adaptive routing algorithm known as DyXYZ was designed for 3D NoCs. To select information among the output channels the congestion information at the input buffer of the neighboring routers was used as a congestion metric. 4, 4, and 2 virtual channels along the X, Y, and Z dimensions were used by this algorithm to provide deadlock-free. Packets were delivered through three sent to an intermediate node in a less congested direction using this algorithm. Until the direction reaches, zero the packets were delivered. Table 6.4 shows the comparison between the injection rate and throughput. Furthermore, figure 6.4 shows the comparison in graphical format.

6.5.1 COMPARATIVE ANALYSIS OF XYZ AND DyXYZ (ADAPTIVE)

To assess the efficiency of the XYZ and DyXYZ (adaptive algorithm, we have used a cycle-accurate Access Noxim simulator and implemented both the algorithms. The values obtained for the throughput and latency of individual algorithms are already presented in table 6.2, 6.3 & 6.4, and their respective analytical graphs are shown in figure 6.1, 6.2, 6.3 & 6.4. For better understanding, we made a comparison between XYZ and DyXYZ algorithm. For this analysis, we have used 6*6*4 mesh topology. It can be visualized from the figure 6.5 & 6.6, and adaptive algorithm outperforms XYZ algorithm.

Table 6.4: Injection rate vs. throughput (DyXYZ adaptive algorithm)

Injection Rate	**Throughput (6*6*4)**	**Throughput (8*8*4)**
0.01	0.0798705	0.0794013
0.02	0.157087	0.158455
0.03	0.235071	0.234774
0.04	0.309366	0.0193707
0.05	0.0178685	0.00817582
0.06	0.0127124	0.00608811
0.07	0.00861753	0.00552243
0.08	0.00783967	0.00471543
0.09	0.00870087	0.00426605
0.1	0.00712293	0.00379332

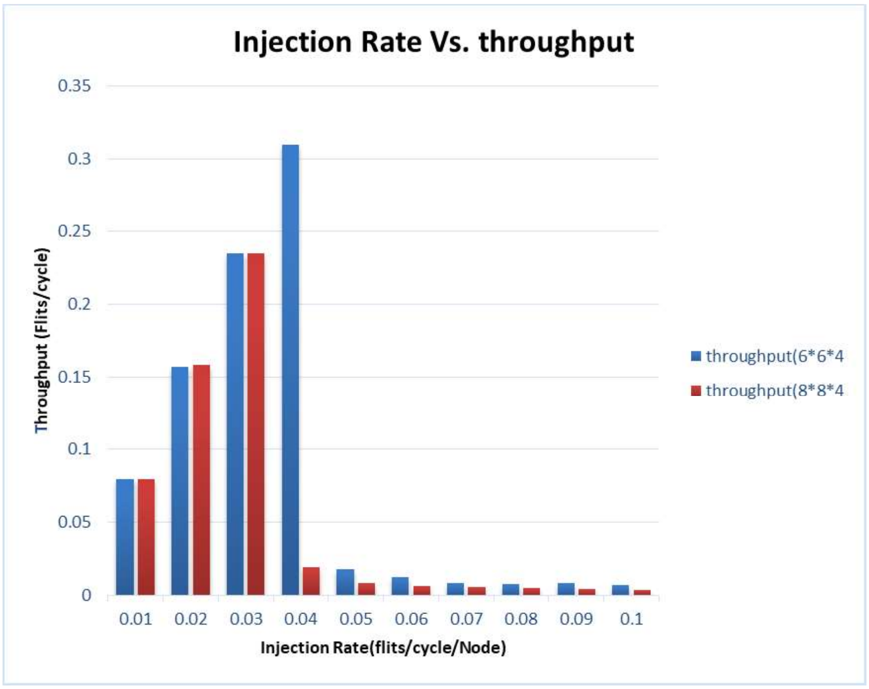


Fig. 6.4: Injection rate vs. latency (DyXYZ adaptive algorithm)

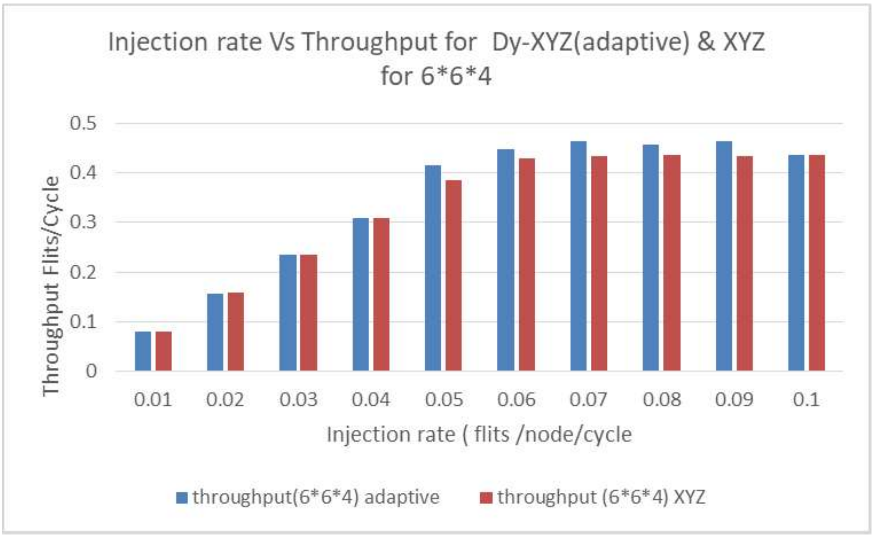


Fig. 6.5: Throughput comparison XYZ and DyXYZ (3D mesh based NoC)

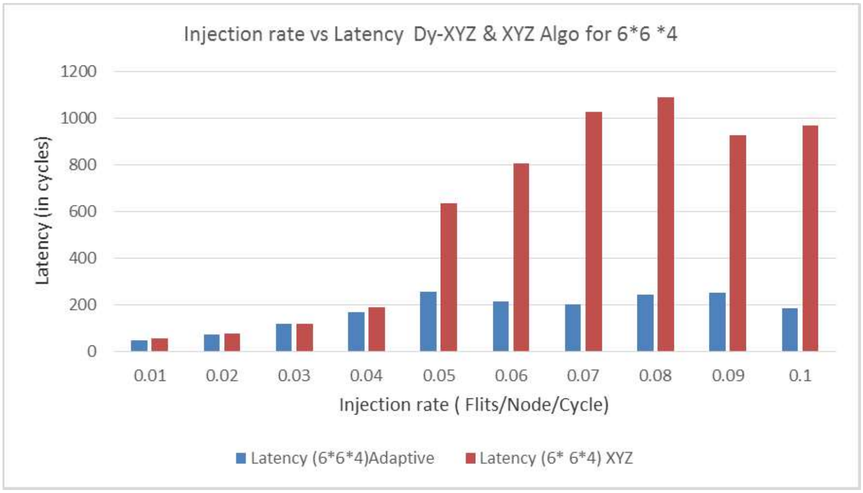


Fig. 6.6: Latency comparison XYZ and DyXYZ (3D mesh based NoC)

6.5.2 ANALYSIS OF TURN MODEL ROUTING ALGORITHM

In [121,122], the author introduced the turn model concept for partially adaptive which are deadlock and livelock free without the addition of virtual channels. We already investigated various turn model algorithms for 2D mesh and 3D mesh in the literature survey. Due to the extensive work already done in turn models routing algorithm, it is important to investigate in more detail for 3D mesh based architecture.

In this section Negative-First routing is implemented using Access Noxim tool. The simulation parameters are same as XYZ and DyXYZ. Latency and throughput are two evaluation criteria used for evaluating mesh topology of size 6*6*4 and 8*8*4. Due to its simplicity and scalability, the most commonly used turn model is the "Negative-First" routing algorithm. The three planes in the 3D mesh are (xy), (xz) and (yz).

To easily understand the algorithm: (x-axis is east, -x-axis is west), (y-axis is north, -y-axis is south), (z-axis is up, -z-axis is down). The routing algorithm for the Negative-First 3D turn model consists of routing a packet first adaptively to -x, -z and -y and then adaptively to + y, + x and+ z. The turns which are not allowed are north-west, east-south, up-west, up-south, down-west and down-south. The turns which are not allowed in each plane are shown in figure 6.7.

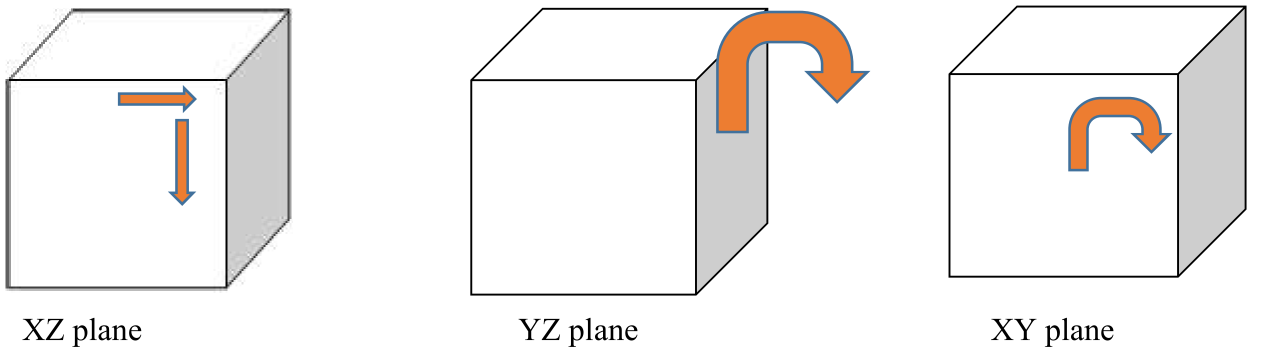


Fig 6.7: Turns which are not allowed in each plane

Negative-First Routing Algorithm:

Input:

```
    Source Node = X-src, Y-src, Z-src
    Destination Node = X-dest, Y-dest, Z –dest
    Current Node  = X-curr, Y-curr, Z-curr
```

```
Begin:
    if X-curr != x-dest && Y-curr != Y-dest
        if X-curr > X-dest
            New_direction = New_direction U west;
        if Y-Curr > Y-dest
            New_direction = New_direction U south;
    else
        if X-curr < X-dest
            New_direction = New_direction U east;
        if Y-curr < Y-dest
            New_direction = New_direction U north;
    if X-curr = X-dest  then
        if Y-curr > Y-dest
            New_direction = New_direction U south;
    else
        if Y-curr < Y-dest
            New_direction = New_direction U north;
        if Z-curr > Z-dest
            New_direction = New_direction U down;
        if Z-curr < Z-dest
            New_direction = New_direction U up;
    if Y-curr = Y-dest
        if X-curr > X-dest
```

```
                    New_direction = New_direction U west;
        else
                if X-curr < X-dest
                    New_direction = New_direction U east;
                if Z-curr > Z-dest
                    New_direction = New_direction U Down;
                if Z-curr < Z-dest
                    New_direction = New_direction U Up;
        select one direction from New_directions
                move forward()
End;
```

The latency and throughput values evaluated for Negative-First algorithm are presented in table 6.5 & 6.6 and figure 6.8 & 6.9.

Table 6.5: Latency values for Negative-First algorithm (3D NoC)

Injection Rate	**Latency (6*6*4)**	**Latency (8*8*4)**
0.01	55	68
0.02	93	122
0.03	127	2374
0.04	269	5366
0.05	2352	5277
0.06	2336	9670
0.07	2570	6541
0.08	1996	5412
0.09	2934	5643
0.1	3148	5635

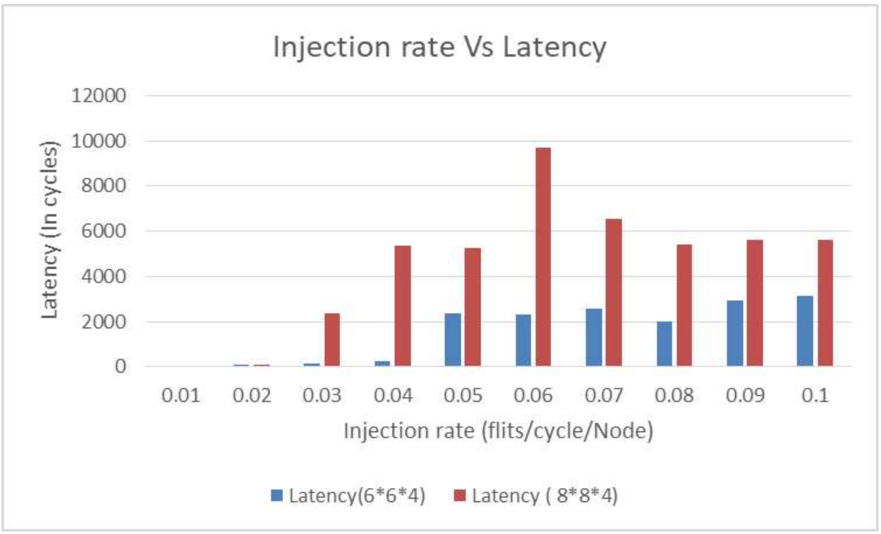


Fig. 6.8: Injection rate and latency for Negative-First algorithm (3D NoC)

Table 6.6: Throughput values for Negative-First algorithm (3D NoC)

Injection Rate	Throughput (6*6*4)	Throughput (8*8*4)
0.01	0.0800052	0.0791497
0.02	0.160618	0.158372
0.03	0.233259	0.23076
0.04	0.310682	0.211056
0.05	0.328582	0.195295
0.06	0.317333	0.18378
0.07	0.288834	0.172543
0.08	0.303165	0.169919
0.09	0.299441	0.180981
0.1	0.293404	0.185249

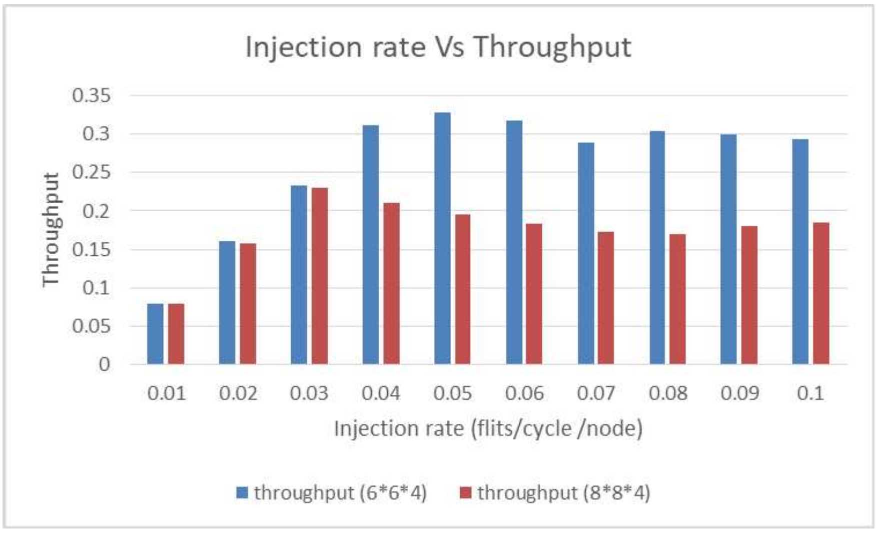


Fig. 6.9: Injection rate and throughput for Negative-First algorithm (3D NoC)

6.6 CONCLUSION

Networks on Chip are becoming increasingly popular as a solution that can accommodate a large number of IP cores, offering an efficient and scalable network of interconnections. As compare to 2D NoC, 3D NoC take advantage of integration and packaging technology. In this chapter, the performance of the 3D Network-on-Chip is analyzed through its performance parameters. We have evaluated XYZ, DyXYZ and Negative-First routing algorithm for 6*6*4 and 8*8*4 mesh topology. All the algorithms are implemented using Access Noxim simulator. The latency and throughput values are presented with respect to the injection rate. In this work, the packet injection rate from .01 to .1 has been taken for the simulation for different routing algorithms. If the injection rate of the packet is 0.04, it means that one node sends 4 packets per 100 cycles to other nodes. The comparison between XYZ and adaptive algorithm is also presented and through graphs and its values it can be concluded that the adaptive algorithm outperforms XYZ algorithm. Turn model "Negative-First" is also presented in the work because these algorithms are widely accepted algorithm for avoiding deadlock and livelock situation. The fundamental idea is to prohibit sufficient turns to break the possible waiting cycles and use the remaining turns to route packets. Thus empirical evaluation of turn model is also important for thesis contribution.

CHAPTER 7. PROPOSED WORK

7.1 BACKGROUND

The Network-on-Chip (NoC) worldview has developed as a dynamic philosophy for coordinating an extremely high number of protected innovation (IP) hinders in a solitary pass on. Up to now, NoC structures were constrained to two measurements. Right now developing 3D mix innovation exhibits two noteworthy favorable circumstances, to be specific higher execution, and littler vitality utilization. It is essential to guarantee ideal throughput and lower idleness to amplify the communication execution. The routing models that have been proposed as of recently, typically have the high overhead of routing tables, information repetition, and worldwide course or blame data endure defective connections. These overheads increment territory and power utilization that isn't reasonable for on-chip networks. Anyway the execution of prior routing techniques are not fulfilled for 3D NOCs, very limited work has been done to focus on faulty links and deadlock issues in 3D NoCs. Although, some existing routing algorithms show fault-tolerant routing that has been designed for 3D NoCs, the proposed methodologies were not able to completely rectify the routing problems in NoCs [36]. Also, they did not provide an efficient congestion and deadlock control mechanism to make optimal routing decisions for complex traffic conditions and maintain system performance in terms of throughput, low latency.

The aim of this research is to design and propose a routing algorithm which is adaptive in nature and suites the requirements of 3D Network-on-Chip. The main objective is to provide substantial support to experiment with NoC design in terms of routing algorithms and applications on different topologies. Much research has been carried out around 2D Network-on-Chip, but there is no effort in the area of adaptive routing for 3D Network-on-Chip which have the ability to avoid congested path and balance the traffic overload, secondly which have the capability to tolerate fault at the time of failures of links. Hence this research work is planned to develop a novel Triggered Faulty-free Route Forwarding (TFRF) model for routing in 3D mesh-based Network-on-Chips.

7.2 TRIGGERED FAULT-FREE ROUTE FORWARDING MODEL (TFRF) FOR 3D NETWORK-ON-CHIPS

In order to decrease the overhead of the fault-tolerant routing with the smooth degradation of ordinary performance, we propose the new model TFRF through two major aspects. One is the way how to implement the routing model to improve all possible parameters like end-to-end delay, throughput, latency, overhead and buffer occupancy to enhance the fault tolerance while routing in 3D NOCs; the other is the key task how to enhance the traffic stream parity of the 3D swing model to ensure the better execution and unwavering quality of 3DNoCs. TFRF demonstrate oversees clog in defective networks by picking routing confinements legitimately. Especially, limitation situations are streamlined to the application's network traffic. We accomplish this objective by presenting a lot of turn actuating decides that grant to rapidly prune the inquiry of stop free courses in broken network topologies. We influence these guidelines to convey versatile, application-mindful courses for packets streaming into the network, abusing the quantity of unequivocal available routing ways. The proposed model can be connected to any sporadic topologies got from a 3D mesh network by including flaws. Unique in relation to prior topology-freethinker routing models, TFRF receipts into thought transmission capacity requests, notwithstanding issue areas, while setting the turn initiating rules. In addition, it likewise keeps its calculation lightweight, when contrasted with existing 3D Network-on-Chip routing models. Figure 7.1 shows the 3D mesh NoC of $4 \times 2 \times 4$.

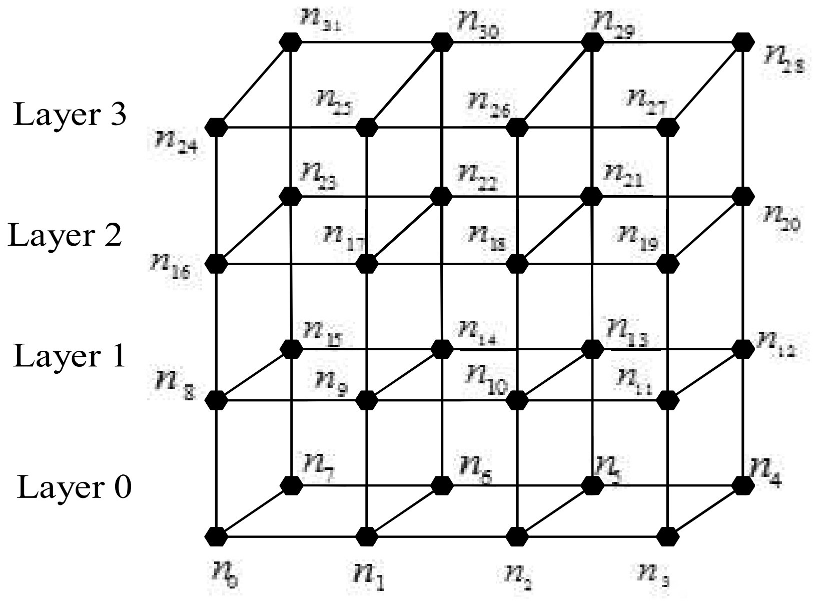


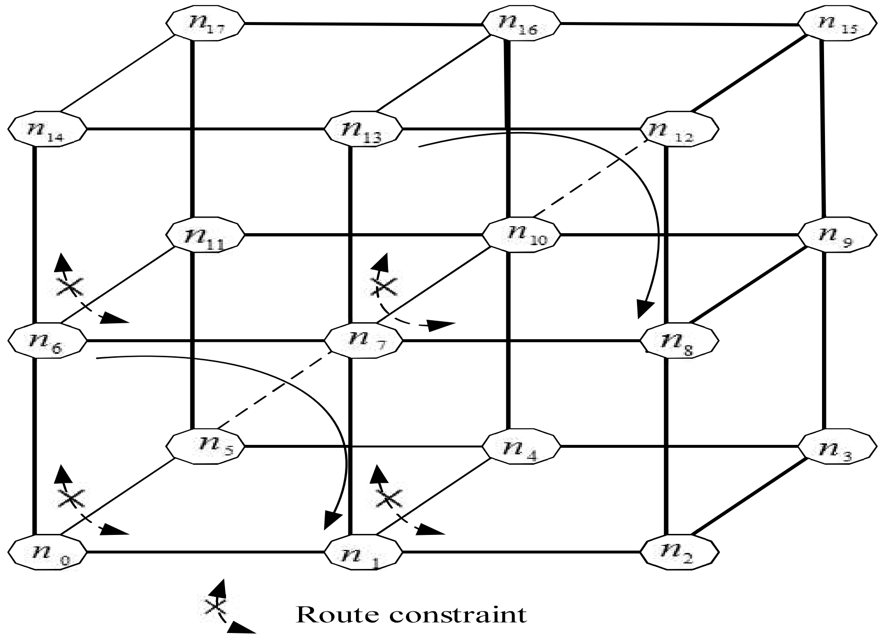
Fig. 7.1: A typical $4 \times 2 \times 4$ 3D mesh NoC

TFRF model rapidly finds a close best routing task: it first registers the insignificant integer of turn constraint that should be put in the network. This significance can be effectively gotten from the aggregate number of cycles in the topology. The development is then considering an iterative investigation, where turn requirement put each one in turn. Subsequent to setting each turn limitation, TFRF model sends its turn-actuating tenets to empower turns that must be dynamic with a particular ultimate objective to keep up availability in light of the most recent turn-confinement choice. Moreover, in picking the region of each new turn limitation, TFRF model impacts traffic load evaluates that it gets from the communication plans separated through application investigation. On the off chance that a halted or detached routing capacity is experienced in the investigation, TFRF model uses backtracking to grow the hunt until it finds a reasonable arrangement. Finally, the network switches are reconstructed using the new routing capacity.

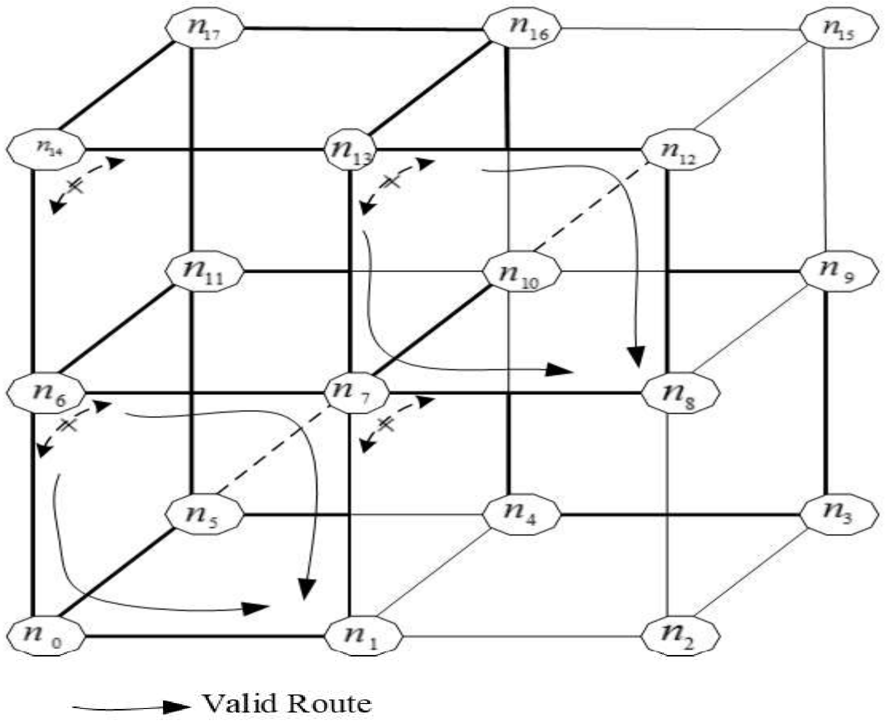
TFRF model expect that data about the application's communication configuration is open. SoC applications can utilize runtime profiling or can be seen in Markov chains, and after that showed through Markov chains [119,120]. As in various blame tolerant routing techniques, we additionally accept that the 3D NoC is furnished with blame determination frameworks where the OS knows about new blame location and can dispatch TFRF model routing calculation, refreshing routing tables likewise. This supposition probably won't hold in a few frameworks. For those frameworks, TFRF can be received by conveying a submitted network director. In this setup, TFRF figures ahead of time streamlined routing capacities for agent communication examples and stores them in memory. At runtime, the network chief surveys current traffic examples and picks the best way as looked at against the examples of the put away ones.

7.2.1 AVOIDING DEADLOCK BY REMOVING CYCLIC RESOURCE DEPENDENCIES

Deadlock states can happen when packets wait for one another in a cyclic way. Such states can be escaped by removing cyclic resource dependencies, deactivating at least one of the turns that contribute to the cycle. For illustration, figure 7.2a restricts the turn $n_{14} - n_6 - n_7$ so that packets routes do not include the sequence $n_{14} \to n_6 \to n_7$ or $n_7 \to n_6 \to n_{14}$. This turn constraint breaks the cycle along nodes n_{14}, n_{13}, n_7 and n_6. Note that TFRF model utilizes bidirectional turn constraint.



a. Routing function 1



b. Routing function 2

Figure 7.2: Traffic-aware routing constraints

The purpose of application-specific routing methods is that it provides complete flexibility in the turn-constraint placement. They inspect an application's communication ways and specifically wipe out a portion of those ways until no cyclic asset reliance exists. Figure 7.2 demonstrates the traffic-mindful routing requirements. To conquer the versatility issue, there are two stages. It at first diminishes the quantity of cyclic conditions and after that can use VCs to break the rest of the cycles. In any case, using VCs to maintain a strategic distance from stop is an expensive alternative in asset obliged 3D NoCs. Conversely, our methodology does not require VCs for stop opportunity, and VC assets can be completely committed to organize traffic or lessen blockage.

7.3 TFRF MODEL-TURN ACTIVATING RULES

This segment clarifies the tenets that direct which turns must be initiated as a result of another turn being deactivated. These guidelines, accumulated into basic and improved, can be connected in any request, and are delineated in figure 7.3, 7.4 and 7.5. We initially delineate how the principles work with ordinary meshes, and a while later connected them to defective 3D mesh topologies. Note that our TFRF model limits the quantity of turn imperative, and as such intensify complete transmission capacity to all goals; anyway it doesn't really initiate insignificant length courses.

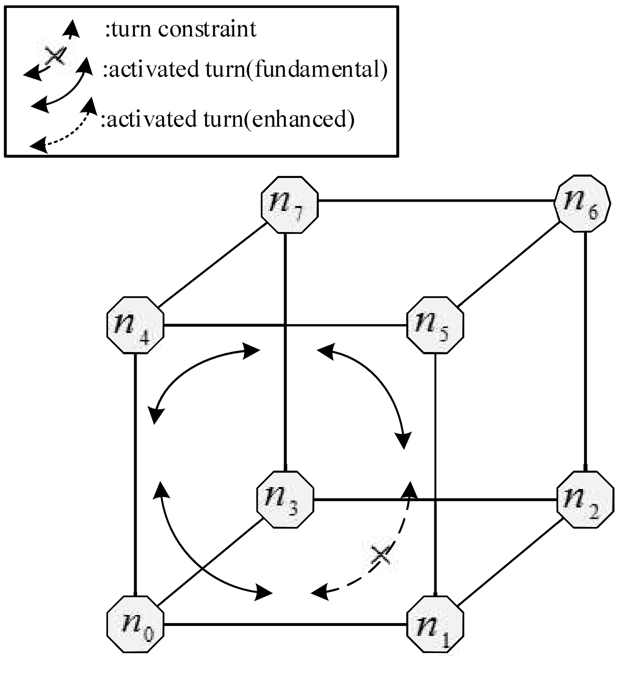
1) Fundamental rules recognize which turns to need to be activated because of a turn constraint applied due to cyclic rotation, node intersection, and link formation rules. They are illustrated in figure 7.3.

Rule 1 (Cyclic Rotation) - Once a turn in a cycle is deactivated, every single other turn in the same cycle ought to be activated so that all nodes in the cycle can still communicate which is shown in figure 7.3a

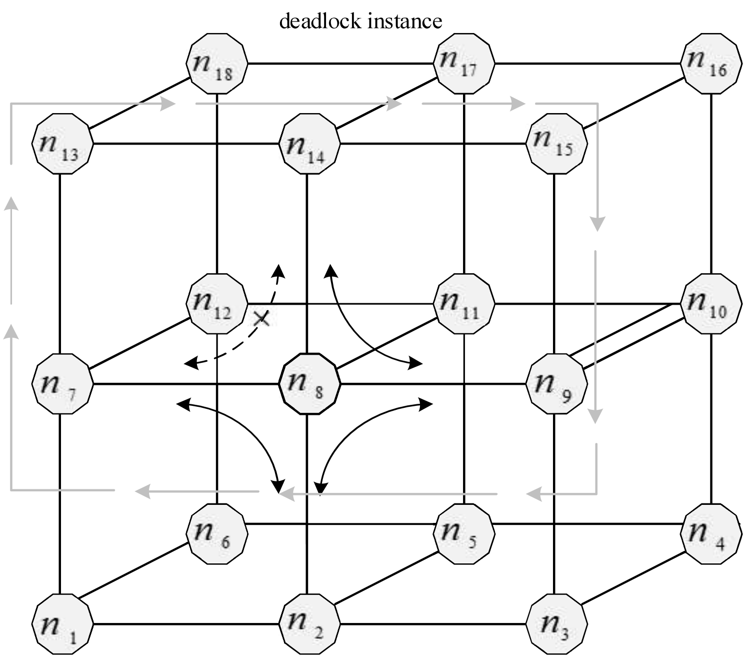
Rule 2 (Node Intersection) - Once a node (router) in turn is deactivated, every single other turn demanding the same node ought to be activated that is shown in figure 7.3b

Rule 3 (Link Formation) - Once a turn adjacent a link is deactivated, the turn to the same link, on the opposite side respects to this turn and not demanding the same router ought to be activated which is shown in figure 7.3c

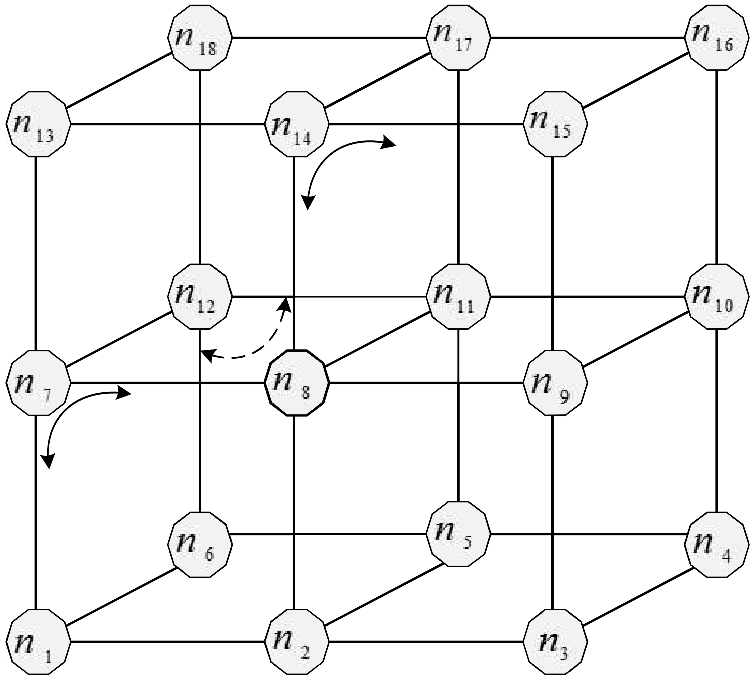
In a blame free 3D mesh network, there is just a single turn that ought to be actuated for each connection influenced by a deactivated turn because of standard 3. Note that rules 2 and 3 are not required to guarantee the availability of the network, however any infraction of these tenets would manual for an excess number of turn deactivating. For example, if both the turns and were crippled in the center network of figure 7.3b, at that point the network would at present permit a reliance along the way as appeared in the figure (dark bolts), and we would need to incorporate an extra swing limitation to break it. So also, with reference to the correct network in figure 7.3a, if turn were incapacitated, a cyclic reliance would exist along the way.



(a) Rule 1: cyclic rotation



(b) Rule 2: node intersection



(c) Rule 3: link formation

Fig. 7.3: TFRF fundamental rule (a, b, c)

2) Enhanced Rules force TFRF model to activate turns that are remote with respect to the constraint turn. They permit to aggressively trim the search space towards a result with a minimal number of deactivated turns. Figure 7.4 and 7.5 illustrates the two rules below, and we denote constraint turn with a dashed arrow with the cut mark, the normal arrow depicts the turns activated by the fundamental rules and in dashed arrow the turns activated by the enhanced rule being illustrated.

Rule 4 (Common Link Formation): if a cycle has just double uncertain turns that share a typical connection, at that point all turns that are nearby that connection and lie outside the cycle, should be enacted.

This standard can be gathered from guidelines 1 and 3. For a cycle with double unsure turns, by principle 1, one of the double ought to be deactivated. In the event that these double turns are contiguous (i.e., sharing a connection), deactivating both of the double turns will dependably include the common connection. We apply rule 3 to this connection with the goal that we don't allow another cycle to put its turn imperative on this connection. An occasion is appeared on the left half of figure 7.4, where the cycle has double unsure turns in the wake of putting the turn limitation at the turns and, sharing the connection. This common connection is additionally nearby two different turns and (both have a place with the cycle), which should be actuated by standard 4. Note that standard 4 can be connected on both parallel and vertical headings of actuating turns. Additionally, it can be iteratively performed, to more lessen the searching region of deactivated turn deployment.

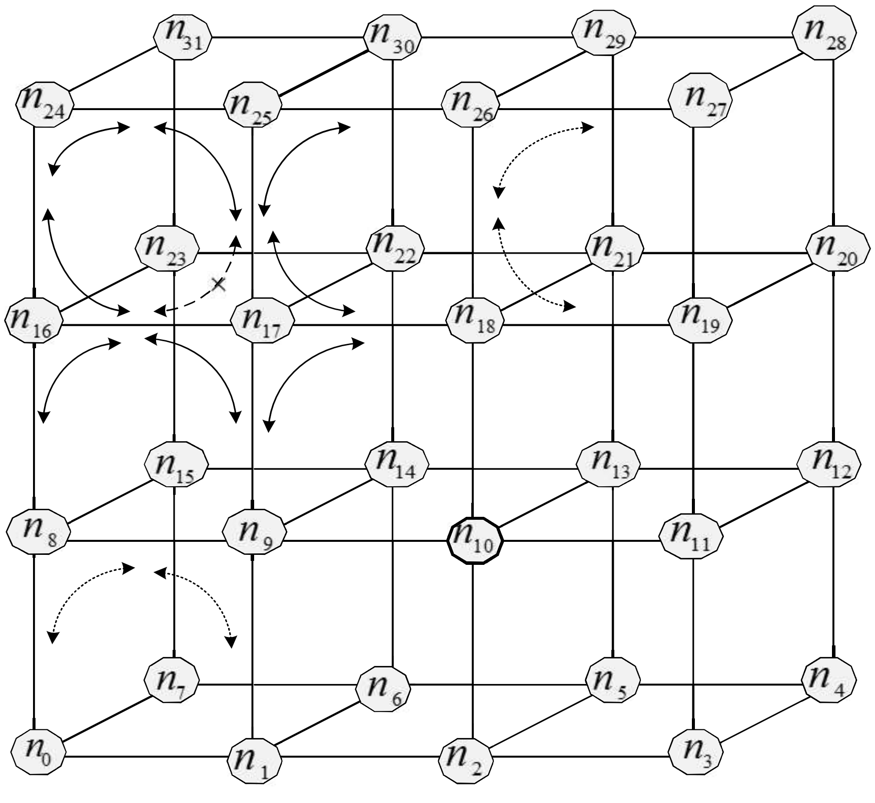


Fig. 7.4: Rule 4: common-link (enhanced rule)

Rule 5 (Opposite-Corner Turn) - If dual turns are situated on opposite nodes in a cycle, and the dual turns are not adjacent to any of the cycle’s links, then only one of them can be deactivated.

We say that such dual turns are situated in opposite-corner locations. The logic behind this rule could be comprehended by the instance on the right part of figure 7.5 if we had deactivated both the opposite-corner turns $n_{25}-n_{17}-n_{16}$ and $n_{11}-n_{10}-n_{2}$, at that point we could no more stay away from a deadlock. Certainly, by rule 2, only dual turns would remain undecided in the central cycle: $(n_{17}-n_{18}-n_{10}-n_{9})$ and $n_{17}-n_{18}-n_{10}$ and $n_{17}-n_{9}-n_{10}$, and by rule 1, we would have to deactivate one of them. Nevertheless, deactivating one of these turns makes a cyclic dependency that could guide to deadlock. As an instance, the figure illustrates, we acquire the cyclic dependency if we deactivate turn $n_{17}-n_{9}-n_{10}$ and in addition, the dual opposite-corner turns. Note that it is feasible to apply rule 5 recurrently by considering increasingly larger cycles. For instance, the rule might be executed to the cycle $n_{17}-n_{19}-n_{3}-n_{1}$, and force the node n_3 placed on the corner of the southeast turn onto be activated (Suppose in the case, that we had a bigger network where that turn existed).

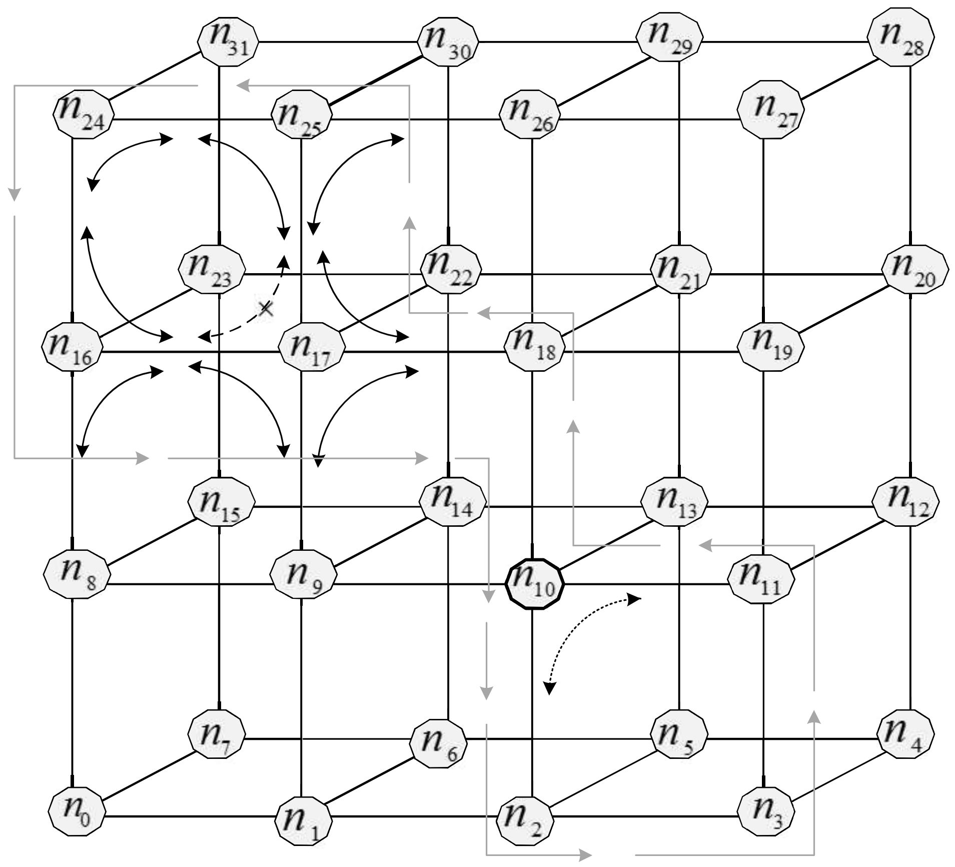
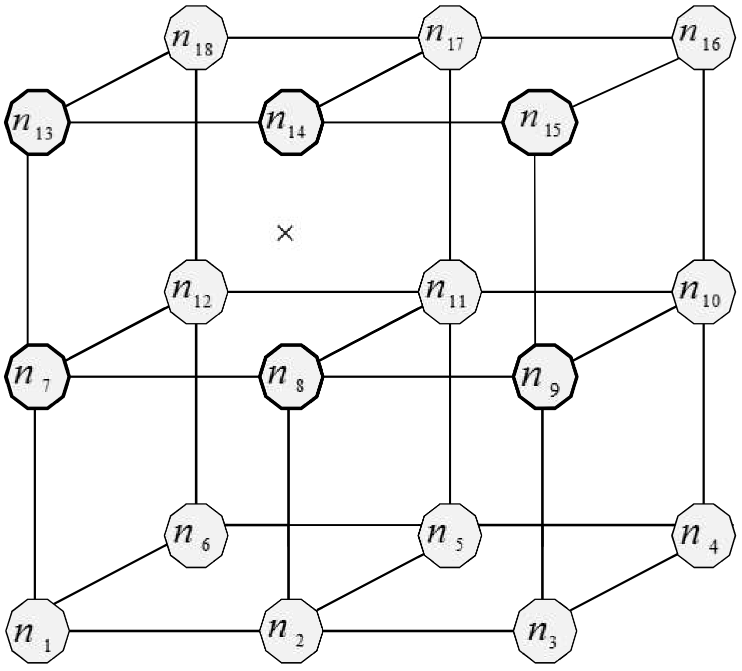


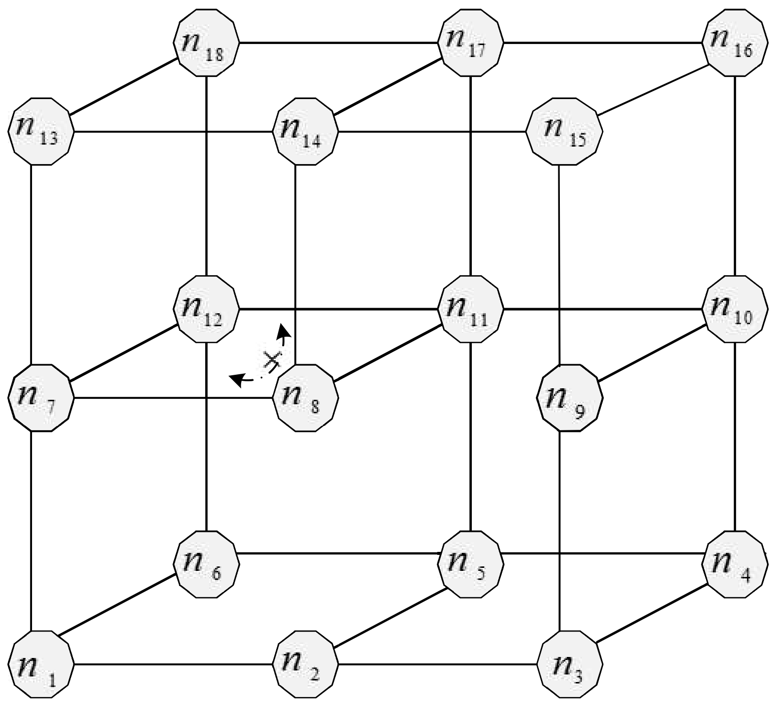
Fig. 7.5: Rule 5: opposite-corner turn (enhanced rule)

7.4 TURN-ACTIVATING RULES IN FAULTY 3D MESH NETWORK TOPOLOGIES

Among the fundamental rules, rule 1 is the one that is influenced the most by faults: cycles might get to be merged in the presence of faults, as appeared in the example of figure 7.6a (link $n_{14} - n_8$ is faulty). In figure 7.6, the two cycles $n_{13} - n_{14} - n_8 - n_7$ and $n_{14} - n_{15} - n_9 - n_8$ no more exist, and they are merged in the cycle $n_{13} - n_{14} - n_{15} - n_9 - n_8 - n_7$. Additionally, faulty networks might have both inwards curved and outwards curved cycles (unlike fault-free 3D mesh networks, which have only outwards curved cycles) and rule 1 must be correctly applied in the case of inwards curved cycles. At the point when a turn is basic to two cycles (e.g., $n_7 - n_8 - n_{14}$ in figure 7.6b), then deactivating that turn can just be computed to removing one of the two cycles, not both.



(a) Cycle recomputation



(b) Mutual Turn

Fig. 7.6: Extending turn-enabling rules to faulty topologies

Rules 2 and 3 remain unaltered for faulty topologies. Suppose link $n_7 - n_8 - n_9$ in figure 7.6a and 7.6b as an instance: if we were to deactivate the turn $n_{15} - n_9 - n_8$, rule 3 would need the turn $n_9 - n_8 - n_2$ to be activated.

Moreover, we constrain the application of rule 4 to links adding to cycles that have not been influenced by faults. For instance, with reference to the left network in figure 7.6b, if the link $n_{17} - n_{18}$ were to be faulty, we would not apply rule 4 to link $n_{26} - n_{18}$, since it adds to a cycle that has been opened by a fault; however, we would even now apply the rule to link $n_8 - n_9$.The reason we restrict the application of rule 4 is that it could get to be mind-boggling to recognize which turns ought to be activated when a link traverses different routers.

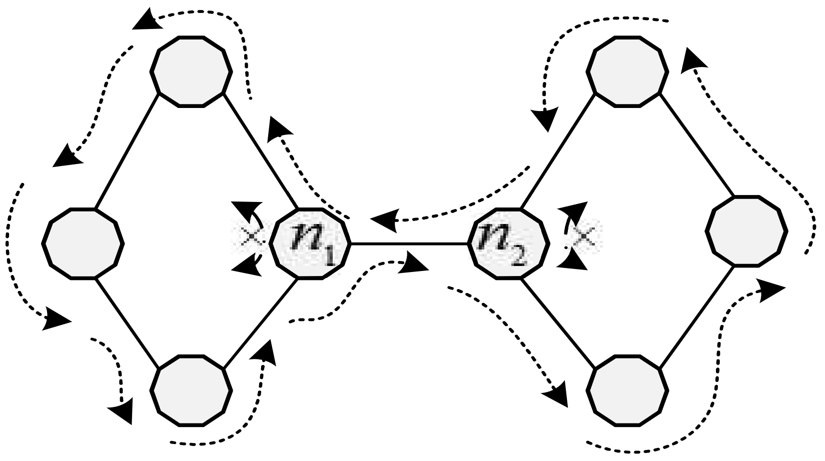


Fig. 7.7: Avoid deadlock

Finally, we basically apply rule 5 as we delineated for ordinary meshes. Applying this standard in flawed networks licenses us to forcefully prune the scan for ideal deactivated-turn situation. In spite of the fact that some actuated turns are superfluously causing a halt in defective networks, and therefore they ought not be enacted. On the off chance that the actuated turns had been deactivated, our backtracking step would address the circumstance. To dodge the backtracking, it is likewise attainable to preventively check whether a turn actuated through standard 5 could prompt a gridlock circumstance. Figure 7.7 tells the best way to check halt for this reason: when there are two unique goes connected through a way beginning at node n_1 and consummation at node n_2, we can't deactivate both the turns demonstrated in figure 7.7, in light of the fact that this arrangement would initiate the gridlock cycle appeared dashed bolt and ending at node, we cannot deactivate both the turns indicated in figure 7.7, because this placement would activate the deadlock cycle shown in dashed arrow.

7.5 TFRF ROUTING ALGORITHM

In this area, we propose TFRF heuristic algorithm to discover a routing capacity with stop free routing and negligible routing imperative. TFRF depends on the data it acquires about the examples of communication to endeavor to put turn requirement on low-traffic joins. At the point when the TFRF routing algorithm starts, all turns are uncertain. Turn requirement is then set each one in turn, start from the locales exchanging the high level of traffic. After setting every limitation, the turn-initiating rules are connected to actuate the related arrangement of turns. This procedure is rehashed until each turn is either enacted or deactivated.

To discover which swing to deactivate straightaway, we at first gauge the traffic load on all cycles, turn and connection. We at that point deactivate the turn that somewhat crumbles the heap exchanging join having a greatest weight, choosing among turns on the most extreme weight load-exchanging cycle. The nature behind this decision is that we have to redress the routing capacity at first around the locales (i.e., cycles) exchanging the most noteworthy traffic so we have a great deal of adaptability in our decisions.

7.5.1 LOAD ESTIMATION OF LINK, TURN, AND CYCLE

To estimate the load on all networks' link, turn and cycle, we examine the single source to destination pair delivered by the network application launching time. For all sources to destination pair, we compute all the feasible paths that packets can take from source to destination, and we then acquire the fraction of traffic that would go via each link. Figure 7.8 shows the format of the routing packet.

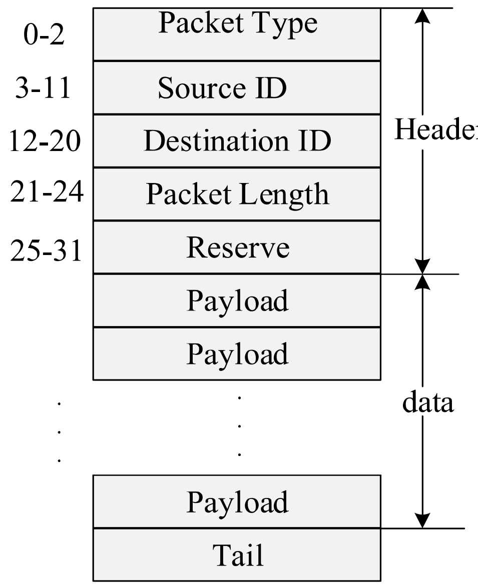


Fig 7.8: Routing packet format

The computation of all the loads follows the four stages given below:

Stage 1: Path Diversity Computation

We compute the number of various possible routes (i.e., path diversity) to reach all links from the source node. In this path diversity computation process, we just permit minimum distance routes inside the limits of the turn constraints that are previously placed.

Stage 2: Link-Load Estimate Computation

We currently utilize the outcomes of stage 1 to estimate the load on all links taking into account the path diversity presented. We compute the total path diversity at all hops from the source and split the link's path diversity by the total diversity. Alternatively, link load can be calculated as

Link Load =Average no of flits per link / Standard deviation of flits per link

Stage 3: Turn-Load Estimate Computation

To estimate the load at all turn, we split the input load from the source direction of the turn by the number of output links permitted for that source.

$$\text{Turn -Load}=\sum \quad Link\ Load\ (in\ turn)$$

Stage 4: Cycle-Load Estimate Computation

For all cycle, the heap is registered by just summing the heaps on every one of the turns in the cycle. After all heap gauges have been registered, we can apply the routing algorithm to finding ideal least routing capacity for a source to goal pair. The routing algorithm for TFRF model starts with choosing a swing to deactivate, choosing the turn with minimal effect on the weightiest interface load, among those in the weightiest load cycle. When the swing to be deactivated is chosen, we apply the turn-actuating tenets to initiate however many different turns as could reasonably be expected. On the off chance that the arrangement of turns initiated/deactivated advisers for a deadlock or a separated network, we backtrack and refresh the rundown of clashing choices.

7.5.2 PROVISIONAL RELAY MECHANISM

Provisional Relay (PR) is utilized to minimize the distance to Des_R (router for destination) by decreasing routing Path Diversity (PD). Additional path diversities lead to a packet to be routed via extra intermediate routers. In other words, the packet has to employ more processing time to (Des_R) and then increases d_{avg} which is the average latency. Besides, previous works need to detect global network information before routing. In our PR, the sender only detects the local network information from a sender to next PR before routing. Hence, it can save time and resource to detect the network information. In addition, we have to select the suitable PR nodes (routers) which forward packets to Des safely. In order to approach deadlock-free and more adaptiveness in non-uniform traffic patterns, our PR node sequentially uses three routing modes: odd-even adaptive routing, xy routing, and downward routing.

When Sr (source) sends a packet to Des (destination), Sr computes the distance of Sr and Des ($|Des - Sr|$) which is used to determine the number of PR nodes before routing (see line 1 in algorithm 1). The distance can be calculated as the number of hops. For each pair of source and destination Max|Des-sr| and Min|Des-Sr| are calculated. Sr detects the network information between Sr and PR. Our PR can save time and resource for detecting network information. By using the PR mechanism, the feasible path diversity is computed to transmit the data packets effectively. The number of PR node is expressed as follows as in (1)

$$\min|Des - Sr| \leq PR_{num} \leq \max|Des - Sr|, \qquad (1)$$

Where PR_{num} denotes the number of PR node.

If PR_{num} is equal to max|Des- Sr|, PR has the highest adaptiveness and the highest PD. Sr needs to detect the global network information from Sr to Des. Contrarily, if PR_{num} is equal to min $|Des-Sr|$, PR uses the shortest path to forward packets from Sr to Des as the deterministic algorithm. In this case, it obtains the lowest PD and the lowest Des_R . Sr only detects the local network information from Sr to next PR, as shown in figure 7.9, but it loses flexibility and fault tolerance in high traffic load. In 3D mesh based NoC, the smallest detecting scope is $4 \times 2 \times 4$.The smallest detecting scope is presented that Sr collects the information of one-hop neighbors so Sr cannot easily determine the correct routing path. Therefore, it easily causes traffic congestion because of imbalance traffic, especially in high traffic load.

Sr can choose the best routing path from three directions: x coordinate, y coordinate, or z coordinate according to the smallest local information of routing table (the information of Sr and neighbors). In order to keep the smallest detecting scope as possible and improve traffic balance in high traffic load, finally, we also choose more PR nodes for routing efficiently. Along with this, the Faulty Area (FA) would also be identified, i.e., a node with faults , deadlocked nodes, highest path diversity, lowest path diversity, congestion, etc. and the Non-faulty Area (NA), i.e., node without fault is analyzed in order to determine faulty free nodes during the packet transmission. FA_{num} is a number of fault nodes presented, four levels are set such as $P_{RT}, P_{RT_{Large}}, P_{RT_{Medium}}, P_{RT_{Small}}$ according to FA conditions in the network where P_{RT} represents the all the routers with FA appearance in the table, subsequently $P_{RT_{Large}}$ has 75% or 0.75 number of routers in FA and it is equal to 75% P_{RT} . $P_{RT_{Medium}}$ has 50% or 0.50 number of routers in FA and is equal to 50% P_{RT} , consequently $P_{RT_{Small}}$ has 25% or 0.25 number of routers in FA and is equal to 25% P_{RT} respectively. Based on the percentage corresponding to $\max|Des - Sr|, P_{RT}, PR_{num}$ where $\max|Des - Sr|$ is the maximum distance between Sr and Des, the current $|Des - Sr|$ and FA_{num} are determined. The flow diagrams are shown in figure 7.9, and 7.10 and further pseudo codes are presented.

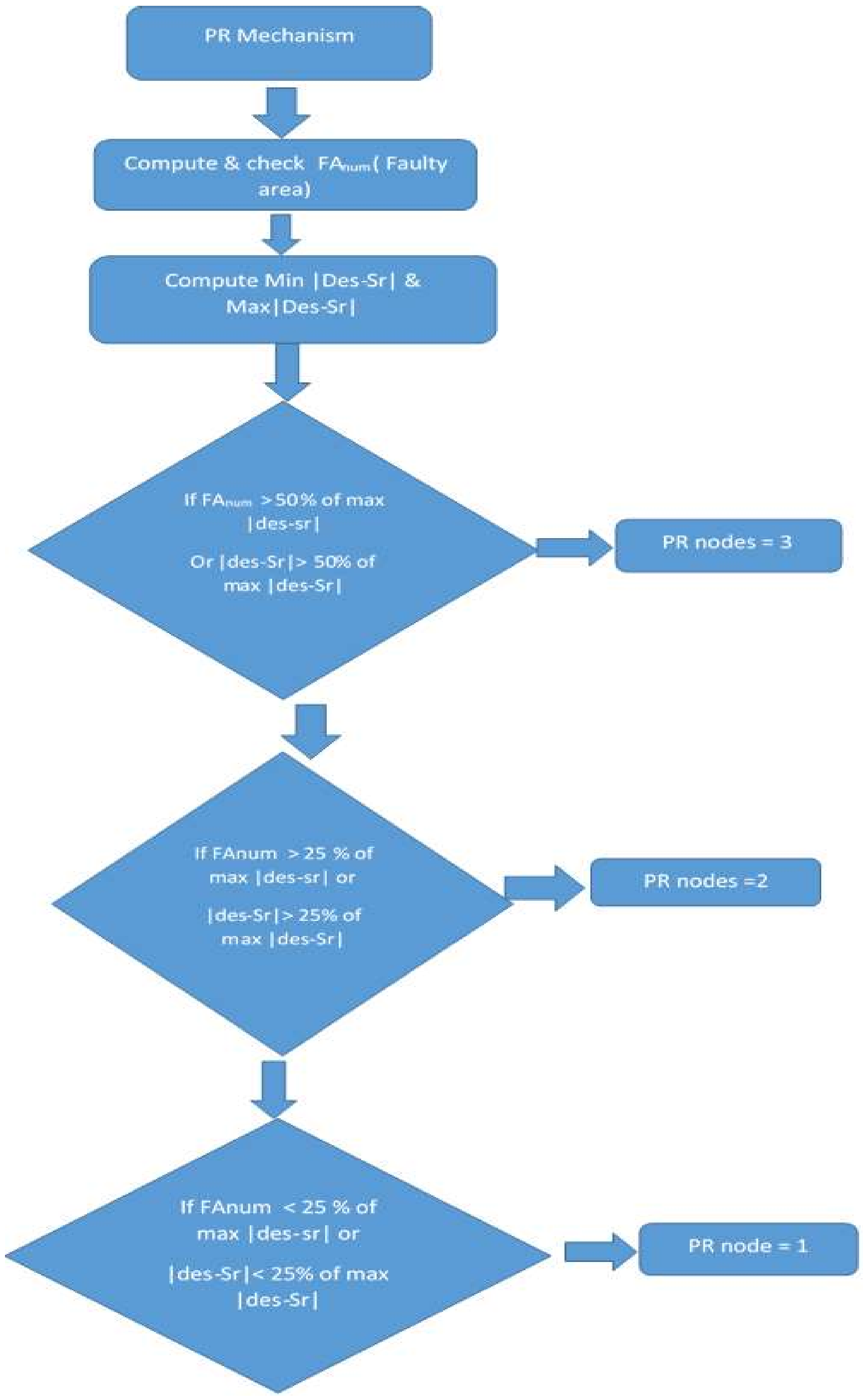
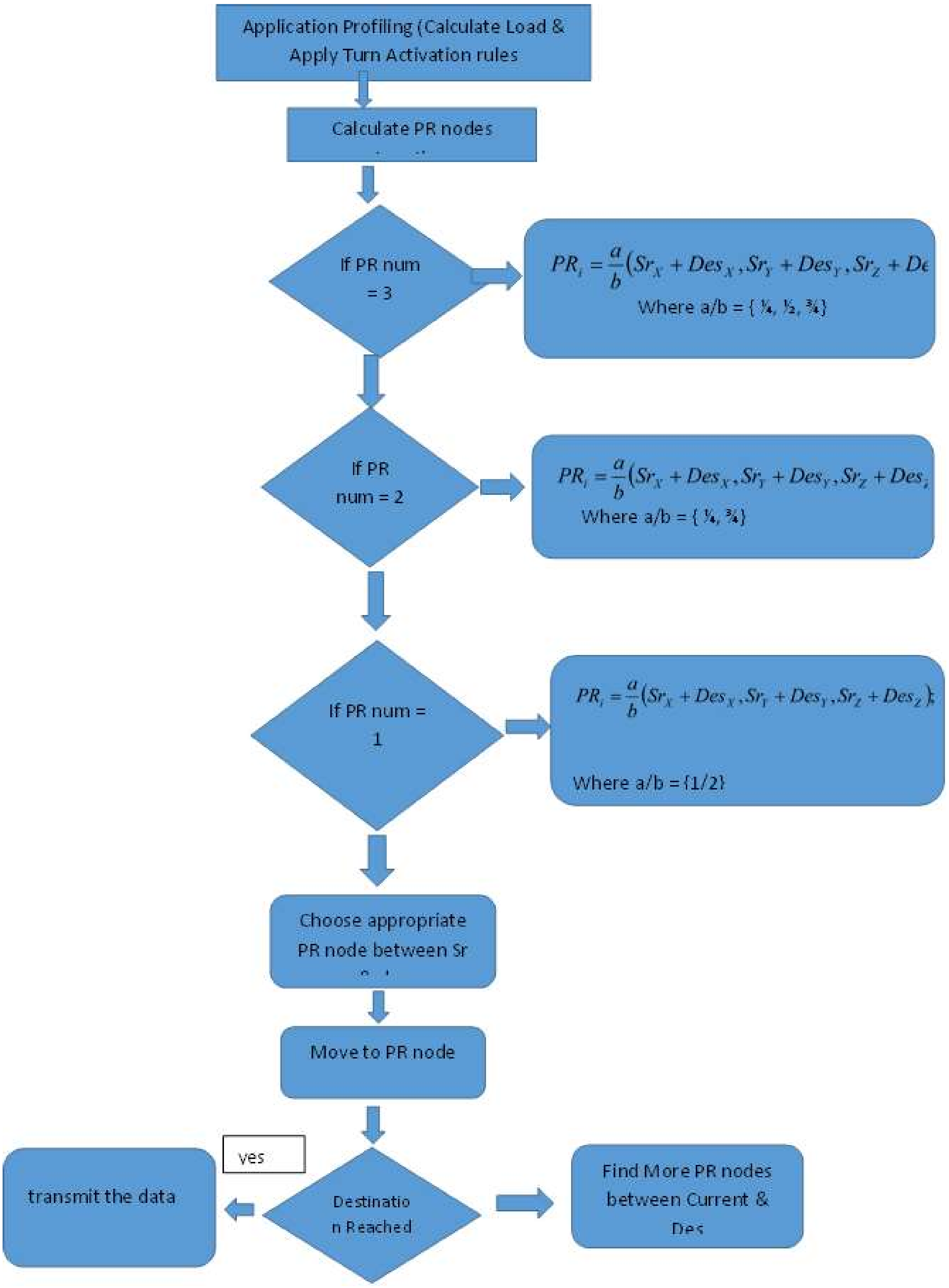

PR Mechanism
Compute & check FAnum (Faulty area)
Compute Min |Des-Sr| & Max|Des-Sr|
If FAnum >50% of max |des-sr|
Or |des-Sr|> 50% of max |des-Sr|
PR nodes = 3
If FAnum >25 % of max |des-sr| or
|des-Sr|> 25% of max |des-Sr|
PR nodes =2
If FAnum < 25 % of max |des-sr| or
|des-Sr|< 25% of max |des-Sr|
PR node = 1

Fig. 7.9: Flow chart of calculation of PR nodes



Application Profiling (Calculate Load & Apply Turn Activation rules
Calculate PR nodes
If PR num = 3
$PR_i = \frac{a}{b}(Sr_X + Des_X, Sr_Y + Des_Y, Sr_Z + De$
Where a/b = { ¼, ½, ¾}
If PR num = 2
$PR_i = \frac{a}{b}(Sr_X + Des_X, Sr_Y + Des_Y, Sr_Z + Des_Z$
Where a/b = { ¼, ¾}
If PR num = 1
$PR_i = \frac{a}{b}(Sr_X + Des_X, Sr_Y + Des_Y, Sr_Z + Des_Z)$
Where a/b = {1/2}
Choose appropriate PR node between Sr
Move to PR node
yes
transmit the data
Destination Reached
Find More PR nodes between Current & Des

Fig. 7.10: Calculation of the PR node location

Pseudo code for TFRF Routing Algorithm:

TFRF ($Sr_{x,y,z}$, $Des_{x,y,z}$)

Variables: PR_{num} , FA_{num} , $P_{RT}, P_{RT_{Large}}, P_{RT_{Medium}}, P_{RT_{Small}}$

```
while forward packets do
    Calculate path diversity for Sr and Des
    Apply PR mechanism to find feasible path diversity
    Compute $|Des - Sr|$;
    Compute $FA_{num}$;
    Compute link, turn, cycle_loads()
        cycle = cycle_with_heaviest_load()
        deactivated_turn = turn_with_smallest_link_load_increase(cycle)
        deactivated_turn = apply_turn_activating_rules(deactivated_turn)
        if not (check_deadloack() or check_disconnnected())
            deactivate(deactivated_turn); activate(activated_turns)
        else
            update_conflict_history(); backtrack()
    Now Evaluating Path between source and destination
        if $P_{RT} \geq FA_{num} \geq P_{RT_{Large}}$ or $|Des - Sr| \geq 0.75 \max |Des - Sr|$ then
            $PR_{num} = 3;$
            $PR_i = \frac{a}{b}(Sr_X + Des_X, Sr_Y + Des_Y, Sr_Z + Des_Z);$
            where $\frac{a}{b} = \{0.25, 0.5, 0.75\};$
            Evaluate_and_move (PR Node)
        else if $P_{RT_{Large}} \geq FA_{num} \geq P_{RT_{Medium}}$ or
          $0.75 \max |Des - Sr| \geq |Des\_Sr| \geq 0.5 \max |Des - Sr|$ then
            $PR_{num} = 2;$
            $PR_i = \frac{a}{b}(Sr_X + Des_X, Sr_Y + Des_Y, Sr_Z + Des_Z);$
            where $\frac{a}{b} = \{0.25, 0.75\};$
            Evaluate_ and_ move (PR node);
        else if $P_{RT_{Medium}} \geq FA_{num} \geq P_{RT_{Small}}$ or
            $0.5\max|Des - Sr| > |Des\_Sr| \geq 0.25\max|Des - Sr|$ then
            $PR_{num} = 1;$
            $PR_i = \frac{a}{b}(Sr_X + Des_X, Sr_Y + Des_Y, Sr_Z + Des_Z);$
            where $\frac{a}{b} = \{0.5\};$
            Evaluate_ and _move (PR node);
              else
                  $PR_{num} = 0;$
        end if
end while
```

Pseudocode for Evaluate_and _Move (PR Node)

```
Destination node = $Des_{z,x,y,}$
Source node = $Src_{z,x,y,}$
Current node = $Curr_{z,x,y}$
PR_Node is an array that stores all preferred nodes that algorithm can use for routing.
Variable = Pref_PR_Node
```

Start

```
    if ($Des_{z,x,y}$ = $Curr_{z,x,y}$)
        Destination_arrived,
        Exit();
    if PR_Node[2] !=Null  && PR_Node[2] is not blocked due to turn_activating rules, fault or deadlocked
        Pref_PR_Node = PR_Node[2]
    else if PR_Node[1] !=Null && PR_Node[1] is not blocked due to turn_activating rules, fault or deadlocked
        Pref_PR_Node = PR_Node[1]
      else Pref_PR_Node = PR_Node[0]
    repeat untill PR_Node != Null
        move_to Pref_PR_Node using ZXY algorithm
            Current_node = Pref_PR_Node
        if current_node = destination node
            Destination_arrived,
            Exit;
            Pref_PR_Node = PR_Node[1]
    if current_node! = destination node
        calculate More PR nodes location between current node and destination node
        Evaluate_and _Move (PR_node)
End
```

ZXY algorithm

```
    Current Node = curr_x, Curr_y, curr_z
    Destination Node = Pref_PR_Node(x,y,z)
```

```
Begin:
        if curr_x = Pref_PR_Node_x, curr_y = Pref_PR_Node_y, curr_z = Pref_PR_Node_Z
                Destination arrived();
                Exit ();
        elseif Curr_z < Pref_PR_Node_z
                Move_to UP;
            elseif Curr_z >Pref_PR_Node_z
                    Move_to down;
                elseif  Curr_z = Pref_PR_Node_z & Curr_x < Pref_PR_Node_x
                        Move_to east;
                    elseif  Curr_z = Pref_PR_Node_z & Curr_x > Pref_PR_Node_x
                            Move_to west;
                        elseif  Curr_z = Pref_PR_Node_z & Curr_y < Pref_PR_Node_y
                                Move_to north;
                            elseif  Curr_z = Pref_PR_Node_z & Curr_y > Pref_PR_Node_y
                                    Move_to south;
        endif
End;
```

The value of PR_{num} is determined as, if FA_{num} is larger than $P_{RT_{Large}}$ or $|Des - Sr|$ is larger than $0.75 \max |Des - Sr|$ then PR_{num} is equal to three, or else if FA_{num} is between $P_{RT_{Large}}$ and $P_{RT_{Medium}}$ or $|Des - Sr|$ is between $0.75 \max |Des - Sr|$ and $0.50 \max |Des - Sr|$ then PR_{num} is equal to two. Also if FA_{num} is between $P_{RT_{Medium}}$ and $P_{RT_{Small}}$ or $|Des - Sr|$ is between $0.50 \max |Des - Sr|$ and $0.25 \max |Des - Sr|$ then PR_{num} is equal to one. Finally if all the above conditions not occur and if FA_{num} is smaller than $P_{RT_{Small}}$ or $|Des - Sr|$ is smaller than $P_{RT_{Small}}$, PR_{num} is equal to zero. After the determination of PR_{num} the locations of PR nodes is determined as in (2)

$$PR_i = \frac{a}{b}\left(Sr_X + Des_X, Sr_Y + Des_Y, Sr_Z + Des_Z\right); \qquad (2)$$

Where $\frac{a}{b}$ is the proportion based on the topology size. If PR_{num} is equal to three then $\frac{a}{b} = \{0.25, 0.5, 0.75\}$ determined from the four-level set value of the FA nodes as $\left\{\frac{1}{4}, \frac{2}{4}, \frac{3}{4}\right\}$, consequently if PR_{num} is equal to two then $\frac{a}{b} = \{0.25, 0.75\}$ and if PR_{num} is equal to one then $\frac{a}{b} = \{0.5\}$.

At last, the TFRF routing algorithm may require backtracking if the arrangement of turn confinements set up takes into consideration deadlock (typically along a mind boggling cycle) or disengages the network. These issues may emerge in light of the fact that the TFRF's principles don't consider every one of the ramifications of a turn limitation, yet just the more straightforward ones, with the goal that their application is computationally shoddy.

So as to assess the execution, unwavering quality, and overhead of the proposed fault tolerant routing plan, we used a 3D cycle-exact NoC dependent on Access Noxim simulator. The PARSEC benchmarks were assessed to approve the upgrades of TFRF. In addition, we additionally utilized rundown creator compiler to break down the significant overhead.

In the tests, we make an examination among TFRF and other existing 3D Network-on-Chip routing models HARS[122], AFRA[121] and HamPA[120]. All the three algos are the most recent advances that meant to endure the lasting connection issues without packet repetition in 3D mesh NoCs. For the far reaching correlation, the execution (under both engineered traffic examples and genuine applications traffic), unwavering quality and overhead (region and vitality) are estimated and delineated through the Access Noxim simulation.

“It is important that TFRF is proposed for 3D mesh NoCs in any scale. For the experiments, we use the common $4 \times 3 \times 3$ 3D mesh NoCs to demonstrate. The general simulation settings of Access Noxim listed in table 7.1. Here, we adopt three major synthetic traffic patterns to implement the schemes: Uniform, transpose, and hotspot. In the uniform pattern, a PE sends a packet to any other PE with equal probability. In the transpose pattern, a PE at (i, j, k) only sends packets to the PE at $(I-i-1, J-j-1, K-k-1)$ while (I, J, K) is the dimensions of the 3D mesh NoC. A PE at some node is designed as the hotspot which receives h% more traffic in addition to the regular uniform pattern in the hotspot pattern.

It should be noted that, for the set fault rates, the faulty links (including both the horizontal and vertical links) are inserted randomly in numerous experiments to get an average value of the related metrics. If the number of faulty links according to the fault rate is not an integer, it will be rounded off except the ones less than 1, which will be regarded as 1.

7.6 RESULTS AND DISCUSSION

7.6.1 PERFORMANCE UNDER SYNTHETIC TRAFFIC PATTERNS

Global Average Latency (GAL) and the throughput are the two key indicators of the QoS of NoCs and is calculated by using the formula given as in (4)

$$GAL = (number\ of\ packet / clockp) - 0.20 * 100 \tag{4}$$

Where $clockp$ denotes the clock time. They also stand for the performance of the corresponding routing scheme. In this section, each indicator will be utilized in the simulation under the synthetic traffic patterns.

Table 7.1: Simulation settings

Parameters	Settings
Flit Size	32 bits
Input Buffer Depth	8 flits
Packet Width	32 flits
Warm-up Time	1000 cycles
Simulation time	100,000 cycles
Topology	$4 \times 3 \times 3$ 3D Mesh

The increasing trend of GALs of the communication in the networks is described with smooth curves under the incremental injection rates. The flatter the curves are, the higher performances of the fault-tolerate routing scheme can guarantee. In figure 7.13, the four schemes are compared according to the GALs under different packet injection rates and traffic patterns in $4 \times 3 \times 3$ 3D mesh NoC.

Figure 7.11 and 7.12 illustrates the proposed TFRF scheme evaluated under the fault rates of links at 0% (fault-free) and 1% (with faults). As shown in figure 7.13, the four schemes are compared under the fault rates of links at 0% (fault-free) and 1% (a common case in practice [37,41]) for illustration. We can observe from the simulation results that TFRF performs better than the others at the two fault rates under the transpose and hotspot (h% = 10%) patterns. Notice that AFRA outperforms TFRF in the 0%-fault case under uniform pattern. This is because of the compatibility between the dimension order routing algorithm and the uniform pattern. Precisely, this is mostly due to that the ZXY/XZXY routing algorithms include more global, long-term information about the characteristics of the uniform traffic, which may cause to more even distribution of traffic. However, for most applications, each node will communicate with some other nodes more frequently, which is under the non-uniform patterns." Table 7.2 shows the evaluation parameters of the proposed model.

The throughput calculation is given as in (5)

$$tp = \frac{number\,of\ packet * packetsize * flitsize}{nsimc * clockp} \qquad (5)$$

Where tp is the throughput, $nsimc$ is the simulation time.

Table 7.2: Parameter evaluation of the proposed model

Injection Rate (flit/node/cycle)	**Global Average Latency (GAL) (Cycle)**		**Injection Rate (packets/node/cycle)**	**Throughput Flits/cycle/node)**	
	TFRF (0%)- Fault Free	**TFRF (1%)- With Faults**		**TFRF (0%)- Fault-Free**	**TFRF (1%)- With Faults**
0.001	43	90	0.001	0.027	0.025
0.0025	165	254	0.0025	0.076	0.074
0.004	505	565	0.004	0.14	0.12
0.0055	1102	1264	0.0055	0.21	0.2
0.007	2230	2460	0.007	0.24	0.22

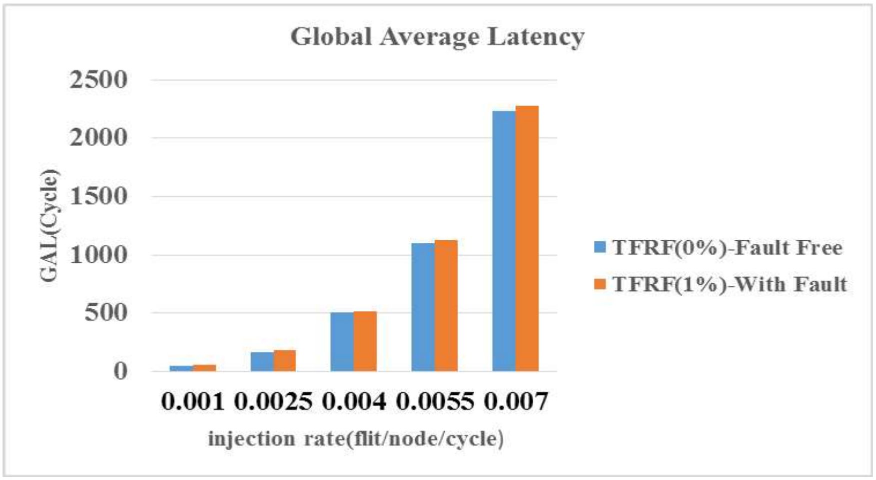


Fig. 7.11: Global average latency of TFRF

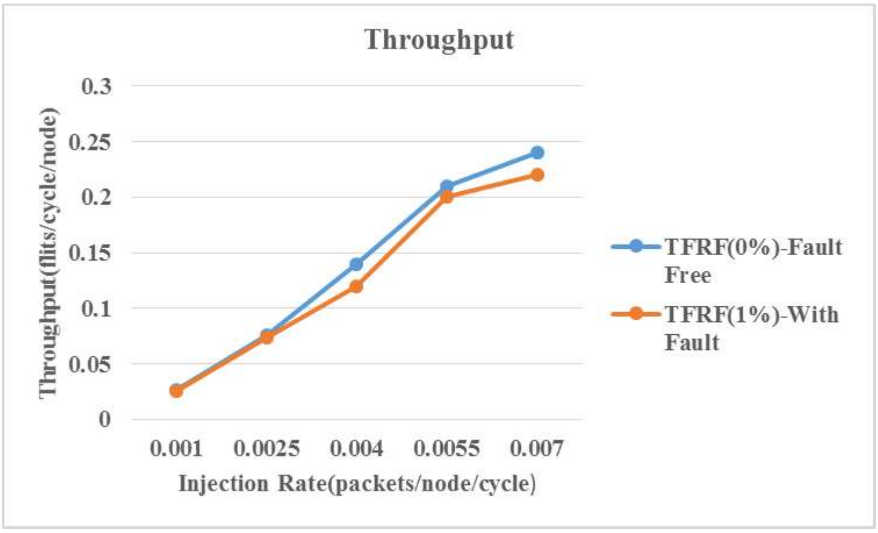


Fig 7.12: Throughput of TFRF model

7.6.2 COMPARATIVE ANALYSIS

TFRF and other existing 3D Network-on-Chip routing models HARS [30], AFRA [37] and HamPA [41].

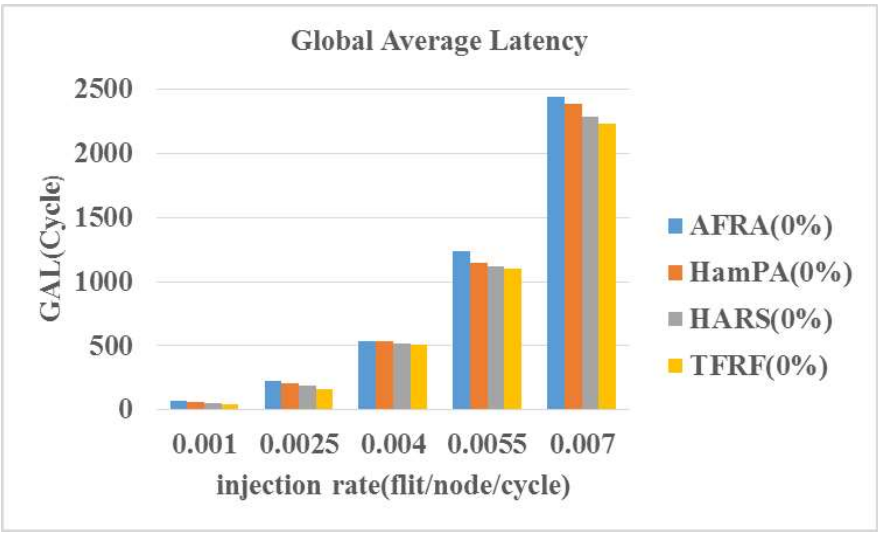


Fig. 7.13a: Comparison of Global Average Latency (fault-free)

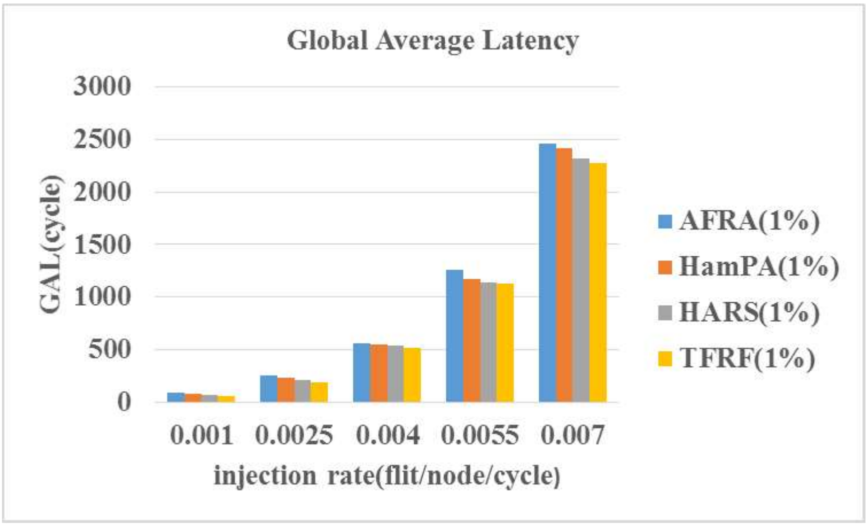


Fig. 7.13b: Comparison of Global Average Latency (with faults)

As shown in figures 7.13a, 7.13b, 7.13c, and 7.14d, the four plans are thought about under the blame rates of connections at 0% (blame free) and 1% for representation. We can see from the reproduction results that TFRF performs superior to the others. In particular, this is basically because of that the 3D routing algorithms join increasingly worldwide, long haul data about the qualities of the uniform traffic, which may prompt an all the more even conveyance of traffic. In any case, for most applications, every node will speak with some different nodes all the more much of the time, which is under the non-uniform examples.

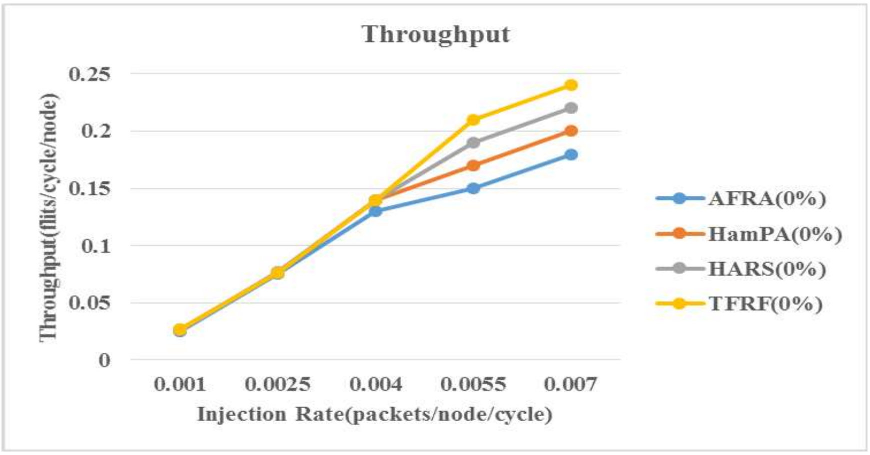


Fig. 7.13c: Comparison of throughput (fault-free)

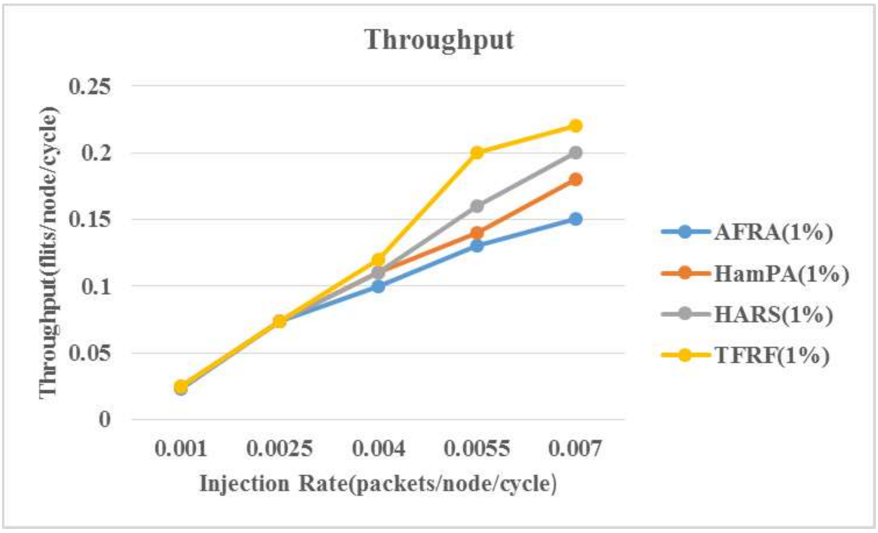


Fig. 7.13d: Comparison of throughput (with fault)

Like GAL, the throughput is another critical measurement to assess the execution of the routing plans. The changing pattern of throughput is fundamentally steady with the GAL as per the rising infusion rate. It shows that the throughput will begin a procedure of moderate ascent and after that decrease when the bends turn out to be level because of the generally high GAL.

7.6.3 PERFORMANCE UNDER REAL APPLICATIONS' TRAFFIC

As per the above reproduction, it very well may be discovered that TFRF could get a decent execution under engineered traffic designs. So as to offer a progressively exact investigation of execution, the PARSEC benchmark suits were embraced to be the outstanding burden of 3D NoCs. Applications follows can be acquired from GEMS. In figure 7.14, the packet idleness over the PARSEC benchmark suits is appeared. The reenactment results are standardized to the inactivity of AFRA at the blame rate of 1%.

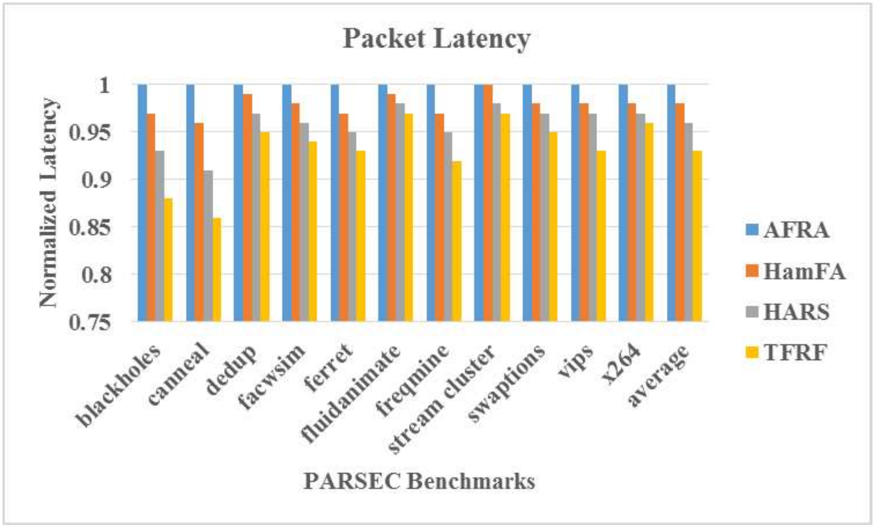


Fig. 7.14: Packet latency across the PARSEC benchmark

From figure 7.14, it very well may be uncovered that TFRF can adequately decrease the packet inactivity. To be explicit, for the high-satisfied applications, as *blackholes* and *canneal*, the enhancement is moderately high. In any case, for the low-placated applications, for example, liquid quicken and stream bunch, the decrease is moderate. All in all, the packet inactivity is diminished by 14.45% maximally and 6.48% overall.

7.6.4 RELIABILITY ANALYSIS UNDER DIFFERENT TRAFFIC PATTERNS

As the center measurement, in this area, the unwavering quality of the blame tolerant routing plan is to be dissected to analyze the robustness of NoCs at explicit blame rates. Right off the bat we present the meaning of the unwavering quality.

Definition: The reliability of the model R can be calculated as the ratio of the number of the received packets (N_A) by the destination nodes to the number of the packets injected into the network (N_S) within the fixed cycles [30,41]:

$$R = (N_A / N_S) \times 100\%$$

Note that the dependability of a routing plan may be under 100% notwithstanding when there is no blame in the networks. It is on the grounds that the packets are infused into the 3D NoCs ceaselessly and the simulation time is restricted.

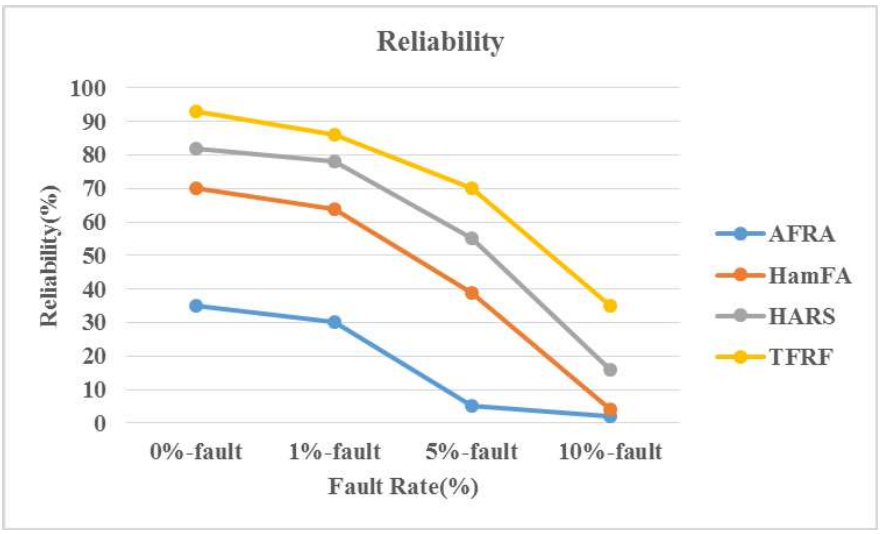


Fig. 7.15: Reliability analysis

Note that the reliability of a routing scheme might be less than 100% even when there is no fault in the networks. It is because the packets are injected into the 3D NoCs continuously and the simulation time is limited.

The reliabilities of AFRA, HamFA, HARS and TFRF under the hotspot pattern (h% = 10%) at different fault rates are shown in figure 7.15. The results are obtained using 100,000 iterations with the injection rate at 0.003.The reliability of TFRF can reach 87.3%, 71.6%, 35.2% in $4\times3\times3$ 3D mesh NoCs with the fault rates of routers at 1%, 5%, and 10% respectively, performing beyond AFRA, HamFA and HARS. Comparative near outcomes were likewise accomplished under the uniform and transpose designs. As a result of the once in a while high blame rates, their liabilities are broadly low in the 5%-blame and 10%-blame cases. All things considered, every one of the outcomes can be enhanced to changing degrees by bringing down the infusion rate and h% of the networks. In the 1%-blame case, for instance, the unwavering quality of TFRF at the infusion rate of 0.001 can ascend to 97.3%.

As a blame endure routing model proposed for the steady connection shortcomings for the overhead-delicate 3D circuits, TFRF additionally claims the adequate overhead or vitality on the zone and power utilization. As a blame endure model, TFRF receives proficient rationale based routing without costly VCs, accordingly requesting less storage room than alternate models requiring additional cradle region and complex control rationale to actualize the table based routing with VCs. Through the combination of Synopsys plan compiler with the UMC 90nm standard cell library, we locate that under 5% development of the router zone (for transmitting and putting away the significant blame data) contrasted and the typical router. In particular, the territory of the ordinary router is 149, 422 μm2, while our router's region is 156,873 μm2. Considering the territory of center and stores, the region increment is irrelevant. In the event that the router zone can be accepted to take up 11% zone of a tile (a tile of the 3D chip incorporates a PE and a router, in some cases the NI is inserted in the PE)[25], the development of the entire zone will be under 1%.

Table 7.3: Operation energy for analysis at an injection rate of 0.001 (Unit: μJ)

Routing Scheme	**Uniform**	**Transpose**	**Hotspot**
AFRA	392.153	454.276	418.244
HamFA	403.397	443.492	407.471
HARS	398.612	441.641	405.346
TFRF	384.712	427.346	388.212

Table 7.4: Operation energy for analysis at an injection rate of 0.002. (Unit: μJ)

Routing Scheme	**Uniform**	**Transpose**	**Hotspot**
AFRA	797.312	916.473	931.546
HamFA	805.193	912.176	926.342
HARS	791.342	903.647	915.473
TFRF	735.941	857.345	860.509

Table 7.5: Operation energy for analysis at an injection rate of 0.003(Unit: μJ)

Routing Scheme	Uniform	Transpose	Hotspot
AFRA	1196.34	1447.49	1568.63
HamFA	1293.67	1422.83	1520.72
HARS	1264.15	1401.94	1482.47
TFRF	1138.65	1286.72	1340.91

The simulation results are listed in table 7.3, 7.4 and 7.5 at the infusion rate of 0.001, 0.002 and 0.003 individually, and blame rate of 1%. It tends to be uncovered from these three tables that the estimations of vitality required by TFRF are by and large lower than the other three plans, particularly under hotspot design. Then again, when the infusion rate gets higher, the hole of the working vitality among TFRF and the others will turn out to be progressively clear in light of the fact that the pattern of the vitality requires dis for the most part reliable with that of the GAL. For example, contrasted and AFRA, TFRF can spare 33.14% vitality at the infusion rate of 0.005 under the hotspot design (h%=10%). This hole is a lot bigger than 6.91% at the infusion rate of 0.001.

7.7 CONCLUSION

In this section, we proposed TFRF, a low-overhead blame tolerant routing plan for 3D NoCs. As a negligible routing, TFRF embraces a rationale based algorithm guided by another 3D turn model and helped by a proficient determination system. Besides, TFRF can guarantee the live lock-freeness and ensured the deadlock-freeness with no staggering expense VCs. The hypothetical investigation and test results demonstrate that TFRF has better execution, enhanced unwavering quality and lower overhead in examination with the current related works. Since the particular network topologies are not considered amid routing, TFRF is versatile to perform on 3D NoCs in different topologies. In addition, the exploration on the similarity with the router shortcomings is the significant work sooner rather than later.

CHAPTER 8. CONCLUSION AND FUTURE SCOPE

Three-dimensional NoCs have been created for complex and high-performance SoCs as the most efficient and scalable communication structures. These three-dimensional structures are a layered architecture that exhibits many advantages like smaller footprint area, high density and higher bandwidth that emerge as a reliable, efficient and scalable communication system. Similar to two-dimensional NoCs, three-dimensional NoCs suffers from various challenges like congestion, faults, deadlock, and livelock.

The thesis work is presented in three phases. The first phase covered in chapter five is the analysis of two-dimensional NoCs. The four basic topologies Mesh, Torus, CMesh, and Fat-tree are evaluated in terms of throughput and latency using dimension order routing algorithm (XY). The algorithm is implemented with the help of cycle accurate Booksim 2.0 simulator. The throughput and latency comparison is shown against a varying range of injection rate for all the four topologies. The tabular values and graphs presented shown that mesh and torus topology performs better in terms of throughput and latency. The average number of hops is also calculated for all the four topologies where Fat-tree outperforms all other topologies. The impact of virtual channels on throughput is also presented for all four topologies. Usually, wormhole technique is used as a flow control mechanism for on-chip communication, but as the network load increases, it leads to blocking that can be solved by virtual channel. As we increase the number of virtual channels, throughput values get increased. After this comparative study, it can be analyzed that torus & mesh can be the good choice for NoC architecture as both perform almost the same. Both topologies have scalable and symmetric architecture. The torus will have additional wiring complexity and more power consumption due to the wrap around edges. Further for a better understanding of routing algorithms odd-even routing and adaptive routing algorithm are also implemented and evaluated with the help of Booksim 2.0 simulator. The throughput and latency values are presented against the injection rate for both the algorithms. Finally, the performance metrics are calculated for XY, odd-even and adaptive algorithms. Performance metrics (P) are the ratio of average latency to average throughput. It is one of the important parameters for evaluating routing algorithm performance. The values calculated shown that adaptive algorithm outperforms XY and odd-even.

The second phase presented in chapter six analyses the 3D NoC. 3D NoCs are adopted as they exhibit shorter global interconnects lengths, less delay, better scalability, and heterogeneous integration. In this phase 3D mesh based NoC is analyzed using Access Noxim simulator. The routing algorithms like XYZ, adaptive and turn model are implemented for 144-node and 256-node 3D mesh NoC. For a range of injection rates within the simulation period, the average latency values and throughput are calculated through various values it can be concluded that adaptive algorithms outperform XYZ algorithm. Further, turn model algorithm Negative-First is also implemented using Access Noxim. The turn model algorithms are widely implemented an algorithm for handling deadlocks and livelocks.

The third phase presented in chapter seven is the proposed Triggered Fault-free Route Forwarding (TFRF) model. The proposed model TFRF is developed and implemented by considering two major aspects. One is the way how to implement the routing model to improve all possible parameters like end-to-end delay, throughput, latency, overhead and buffer occupancy to enhance the fault tolerance while routing in 3D NOCs and the other is the key operation how to improve the traffic flow balance of the 3D turn model to guarantee the better performance and reliability of 3DNoCs. Turn model is implemented by introducing five turn activation rules. The turn activation rules are applied by estimating load on links, turns and cycles. Provisional relay mechanism is used for routing. In provisional relay mechanism, number of PR nodes is calculated first required to transfer the data between source and destination. Once it is done, the location of PR nodes is calculated and using that PR location the data is transferred between source and destination using ZXY algorithm. The proposed TFRF model is compared with the three latest schemes AFRA, HamPa, HARS. All the three algorithms and proposed TFRF algorithm are compared in absence and presence of faults. The parsec benchmarking is also used to evaluate the performance of all four algorithms. From the comparisons presented, it can be revealed that TFRF can effectively reduce the packet latency. To be specific, for the high-contented applications, like *blackholes* and *canneal*, the improvement is relatively high. But for the low-contented applications, such as fluid animate and stream cluster, the reduction is moderate. In conclusion, the packet latency is reduced by 14.45% maximally and 6.48% on average. Finally the reliability is also measured for all the four algorithms to compare the robustness. The reliability of TFRF can reach 87.3%, 71.6%, 35.2% in $4 \times 3 \times 3$ 3D mesh NoCs with the fault rates of routers at 1%, 5% and 10% respectively, performing beyond AFRA, HamFA and HARS. TFRF can ensure the live lock-freeness, and guaranteed the deadlock-freeness without any high-cost VCs. The theoretical analysis and experimental results show that TFRF has better performance, improved reliability and lower overhead in comparison with the existing related works. Because the specific network topologies are not considered during routing, TFRF is scalable to perform on 3D NoCs in other topologies. Moreover, the research on the compatibility" with the router faults is the major work in the near future.

References

[1] Bottoms, B. (2007). The International Roadmap for Semiconductors 2007. 2007 8th International Conference on Electronic Packaging Technology.

[2] Benini, L., & Micheli, G. D. (2002). Networks on chips: A new SoC paradigm. Computer, 35(1), 70-78.

[3] Adriahantenaina, A., Charlery, H., Greiner, A., Mortiez, L., & Zeferino, C. (2003). SPIN: A scalable, packet switched, on-chip micro-network. 2003 Design, Automation, and Test in Europe Conference and Exhibition, Munich, Germany, 70-73.

[4] Guerrier, P., & Greiner, A. (2000). A generic architecture for on-chip packet-switched interconnections. Proceedings Design, Automation and Test in Europe Conference and Exhibition 2000 (Cat. No. PR00537), Paris, France, 250-256.

[5] Ansari, Abdul Quaiyum & Mohammad, Ayoub. (2012). A Journey from Computer Networks to Networks-on-Chip. IEEE Beacon. 31. 71-77.

[6] Dally, W., & Towles, B. (2001). Route packets, not wires: On-chip interconnection networks. Proceedings of the 38th Design Automation Conference, Las Vegas, NV, USA, 684-689.

[7] Kumar, S., Jantsch, A., Soininen, J., Forsell, M., Millberg, M., Oberg, J., Hemani, A. (2002). A network on chip architecture and design methodology. Proceedings IEEE Computer Society Annual Symposium on VLSI. New Paradigms for VLSI Systems Design. ISVLSI 2002, Pittsburgh, PA, USA, 117-124.

[8] Wang, J., Gu, H., Yang, Y., & Wang, K. (2013). An energy-and buffer-aware fully adaptive routing algorithm for Network-on-Chip. Microelectronics Journal, 44(2), 137-144.

[9] Taylor, M., Kim, J., Miller, J., Wentzlaff, D., Ghodrat, F., Greenwald, et al. (2002). The Raw Microprocessor: A computational fabric for software circuits and general-purpose programs. IEEE Micro, 22(2), 25-35.

[10] Gangwal, O. P., Rădulescu, A., Goossens, K., Pestana, S. G., & Rijpkema, E. (2005). Building Predictable Systems on Chip: An Analysis of Guaranteed Communication in the Aethereal Network on Chip. Philips Research Dynamic and Robust Streaming in and between Connected Consumer-Electronic Devices, 1-36.

[11] Access Noxim. (n.d.). Retrieved January 30, 2019, from http://access.ee.ntu.edu.tw/noxim/index.html

[12] Ozel, O., Uysal-Biyikoglu, E., & Girici, T. (2010). Optimal buffer partitioning on a multiuser wireless link. 2010 Information Theory and Applications Workshop (ITA), 1-10.

[13] Goossens, K., Dielissen, J., Meerbergen, J. V., Poplavko, P., Rădulescu, A., Rijpkema, E., . . . Wielage, P. (2003). Guaranteeing the Quality of Services in Networks on Chip. Networks on Chip, 61-82.

[14] Natalie, E. J., and T, L.-S. P. (2009). On-Chip Networks. Morgan and Claypool Publishers.

[15] Sigüenza-Tortosa, D., Ahonen, T., & Nurmi, J. (2004). Issues in the development of a practical NoC: The Proteo concept. Integration, 38(1), 95-105.

[16] Fu, B., Han, Y., Ma, J., Li, H., & Li, X. (2011). An abacus turn model for time/space-efficient reconfigurable routing. ACM SIGARCH Computer Architecture News, 39(3), San Jose, CA, 259-270.

[17] Ni, L., & Mckinley, P. (1993). A survey of wormhole routing techniques in direct networks. Computer, 26(2), 62-76.

[18] Semiconductors, Philips. (2002). Device Transaction Level (DTL) Protocol Specification.

[19] Gebremichael-Tesfagiorgis, B & W. Vaandrager, F & Zhang, M. (2005). Formal Models of Guaranteed and Best-Effort Services for Network on Chip. Journal of Neurophysiology, ICIS, Radboud University Nijmegen.

[20] Palesi, M., Kumar, S., Holsmark, R., & Catania, V. (2007). Exploiting Communication Concurrency for Efficient Deadlock-Free Routing in Reconfigurable NoC Platforms. 2007 IEEE International Parallel and Distributed Processing Symposium, Rome, 1-8.

[21] Dally, W., & Aoki, H. (1993). Deadlock-free adaptive routing in multicomputer networks using virtual channels. IEEE Transactions on Parallel and Distributed Systems, 4(4), 466-475.

[22] Karim, F., Nguyen, A., & Dey, S. (2002). An interconnect architecture for networking systems on chips. IEEE Micro, 22(5), 36-45.

[23] Kumar, S., Jantsch, A., Soininen, J., Forsell, M., Millberg, M., Oberg, J., . . . Hemani, A. (2002). A network on chip architecture and design methodology. Proceedings IEEE Computer Society Annual Symposium on VLSI. New Paradigms for VLSI Systems Design. ISVLSI 2002, Pittsburgh, PA, USA, 117-124.

[24] Chan, E., Chao, C., Jheng, K., Hsin, H., & Wu, A. (2010). ACO-based Cascaded Adaptive Routing for traffic balancing in NoC systems. The 2010 International Conference on Green Circuits and Systems, Shanghai, 317-322.

[25] Duato, J. (1995). A necessary and sufficient condition for deadlock-free adaptive routing in wormhole networks. IEEE Transactions on Parallel and Distributed Systems, 6(10), 1055-1067.

[26] Seiculescu, C., Murali, S., Benini, L., & Micheli, G. D. (2010). A method to remove deadlocks in Networks-on-Chips with Wormhole flow control. 2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010), Dresden, 1625-1628.

[27] Jerger, N. E., Krishna, T., & Peh, L. (2017). On-Chip Networks, Second Edition. Synthesis Lectures on Computer Architecture, 12(3), 1-210.

[28] Patooghy, A., & Miremadi, S. G. (2009). XYX: A Power & Performance Efficient Fault-Tolerant Routing Algorithm for Network on Chip. 2009 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing, Weimar, 245-251.

[29] Li, M., Zeng, Q., & Jone, W. (2006). DyXY - a proximity congestion-aware deadlock-free dynamic routing method for network on chip. 2006 43rd ACM/IEEE Design Automation Conference, San Francisco, CA, 849-852.

[30] Hu, J., & Marculescu, R. (2004). DyAD. Proceedings of the 41st Annual Conference on Design Automation - DAC 04, San Diego, CA, USA, 260-263.

[31] Dehyadgari, M., Nickray, M., Afzali-Kusha, A., & Navabi, Z. (2005). Evaluation of Pseudo Adaptive XY Routing Using an Object-Oriented Model for NOC. 2005 International Conference on Microelectronics, 1-5.

[32] Bobda, C., Ahmadinia, A., Mateusz, M., Teich, J., Fekete, S., & Veen, J. V. (2005). DyNoC: A dynamic infrastructure for communication in dynamically reconfigurable devices. International Conference on Field Programmable Logic and Applications, 2005., Tampere, 153-158.

[33] Schwiebert, L., & Jayasimha, D. N. (1995). Optimal fully adaptive minimal wormhole routing for meshes. Journal of Parallel and Distributed Computing, 27(1), 56-70.

[34] Atagoziyev, M. (2007). Routing algorithms for on-chip networks. M. Sc. Thesis, Middle East Technical University.

[35] Shafiee, A. M., Montazeri, M., & Nikdast, M. (2008). An Innovational Intermittent Algorithm in Networks-On-Chip (NOC). World Academy of Science, Engineering, and Technology International Journal of Computer and Information Engineering, 2(9), 2907-2909.

[36] Glass, C., & Ni, L. (1992). The Turn Model for Adaptive Routing. [1992] Proceedings the 19th Annual International Symposium on Computer Architecture, Gold Coast, Australia, 278-287.

[37] Chiu, G. (2000). The odd-even turn model for adaptive routing. IEEE Transactions on Parallel and Distributed Systems, 11(7), 729-738.

[38] Fu, B., Han, Y., Li, H., & Li, X. (2014). The abacus turn model. In Routing Algorithms in Networks-on-Chip (pp. 69-103). Springer, New York, NY.

[39] Zou, Y., & Pasricha, S. (2010). NARCO: Neighbor Aware Turn Model-Based Fault-Tolerant Routing for NoCs. IEEE Embedded Systems Letters, 2(3), 85-89.

[40] Chaix, F., Avresky, D., Zergainoh, N., & Nicolaidis, M. (2010). Fault-Tolerant Deadlock-Free Adaptive Routing for Any Set of Link and Node Failures in Multi-cores Systems. *2010* Ninth IEEE International Symposium on Network Computing and Applications, Cambridge, MA, 52-59.

[41] Tsai, W., Chu, K., Hu, Y., & Chen, *S. (2013*). Non-minimal, turn-model based NoC routing. Microprocessors and Microsystems, 37(8), 899-914.

[42] Liu, J., Harkin, J., Li, Y., & Maguire, L. (2015). Low-cost fault-tolerant routing algorithm for Networks-on-Chip. Microprocessors and Microsystems, 39(6), 358-372.

[43] Pirretti, M., Link, G., Brooks, R., Vijaykrishnan, N., Kandemir, M., & Irwin, M. (2004). Fault tolerant algorithms for network-on-chip interconnect. IEEE Computer Society Annual Symposium on VLSI, Lafayette, LA, USA, 46-51.

[44] Kim, K., Lee, S., Lee, K., & Yoo, H. (2005). An Arbitration Look-Ahead Scheme for Reducing End-to-End Latency in Networks on chip. 2005 IEEE International Symposium on Circuits and Systems, 3, 2357-2360.

[45] Ali, M., Welzl, M., & Hellebrand, S. (2005). A dynamic routing mechanism for network on chip. *2005 NORCHIP*, Oulu, Finland, 70-73.

[46] Chen, Y., Chang, E., Hsin, H., Chen, K. J., & Wu, A. A. (2017). Path-Diversity-Aware Fault-Tolerant Routing Algorithm for Network-on-Chip Systems. IEEE Transactions on Parallel and Distributed Systems, 28(3), 838-849.

[47] Rahmani, A., Liljeberg, P., Plosila, J., & Tenhunen, H. (2010). BBVC-3D-NoC: An Efficient 3D NoC Architecture Using Bidirectional Bisynchronous Vertical Channels. 2010 IEEE Computer Society Annual Symposium on VLSI, Lixouri, Kefalonia, 452-453.

[48] Hu, S., & Lin, X. (2012). A Symmetric Odd-Even Routing Model in Network-on-Chip. 2012 IEEE/ACIS 11th International Conference on Computer and Information Science, Shanghai, 457-462.

[49] Ebrahimi, M., Daneshtalab, M., Farahnakian, F., Plosila, J., Liljeberg, P., Palesi, M., & Tenhunen, H. (2012). HARAQ: Congestion-Aware Learning Model for Highly Adaptive Routing Algorithm in On-Chip Networks. *2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*, Copenhagen, 19-26.

[50] Farahnakian, F., Ebrahimi, M., Daneshtalab, M., Plosila, J., & Liljeberg, P. (2012). Adaptive reinforcement learning method for networks-on-chip. *2012 International Conference on Embedded Computer Systems (SAMOS)*, Samos, 236-243.

[51] Matsutani, H., Koibuchi, M., & Amano, H. (2007). Tightly-Coupled Multi-Layer Topologies for 3-D NoCs. *2007 International Conference on Parallel Processing (ICPP 2007)*, Xi'an, 75-75.

[52] Tyagi, S., Maheshwari, P., Agarwal, A., & Avasthi, V. (2017). Exploring 3D Network-on-Chip Architectures and Challenges. *2017 International Conference on Computer and Applications (ICCA)*, Doha, 97-101.

[53] Rahmani, A., Liljeberg, P., Plosila, J., & Tenhunen, H. (2012). An Efficient Hybridization Scheme for Stacked Mesh 3D NoC Architecture. *2012 20th Euromicro International Conference on Parallel, Distributed and Network-based Processing*, Garching, 507-514.

[54] Jouybari, H. N., & Mohammadi, K. (2014). A low overhead, fault tolerant and congestion-aware routing algorithm for 3D mesh-based Network-on-Chips. Microprocessors and Microsystems, 38(8), 991-999.

[55] Feero, B. S., & Pande, P. P. (2009). Networks-on-Chip in a Three-Dimensional Environment: A Performance Evaluation. IEEE Transactions on Computers, 58(1), 32-45.

[56] Bahmani, Maryam & Sheibanyrad, Abbas & Pétrot, Frédéric. (2011). Vertical Partial 3D Mesh-Based NoC architecture. SoC / SiP GDR Symposium, Cergy.

[57] Viswanathan, N., Paramasivam, K., & Somasundaram, K. (2012). Exploring Hierarchical, Cluster-based 3D Topologies for 3D NoC. Procedia Engineering, 30, 606-615.

[58] Paulo, V. D., & Ababei, C. (2010). 3D Network-on-Chip Architectures Using Homogeneous Meshes and Heterogeneous Floorplans. International Journal of Reconfigurable Computing, 2010, 1-12.

[59] Li, M., Gu, H., Yang, Y., & Wang, K. (2014). A 3D topology based-on partial overlapped clusters for NoC. *IEICE Electronics Express*, 11(19), 20140790-20140790.

[60] Ghidini, Y., Webbed, T., Moreno, E., Quadros, I., Fagundes, R., & Marcon, C. (2012). Topological impact on latency and throughput: 2D versus 3D NoC comparison. *2012 25th Symposium on Integrated Circuits and Systems Design (SBCCI)*, Brasilia, 1-6.

[61] Dally, W. (1992). Virtual-channel flow control. *IEEE Transactions on Parallel and Distributed Systems*, 194-205.

[62] Yin, A. W., Xu, T. C., Liljeberg, P., & Tenhunen, H. (2009). Explorations of Honeycomb Topologies for Network-on-Chip. *2009 Sixth IFIP International Conference on Network and Parallel Computing, Gold Coast*, QLD, 73-79.

[63] Pavlidis, V. F., & Friedman, E. G. (2007). 3-D Topologies for Networks-on-Chip. *IEEE* Transactions on Very Large Scale Integration (VLSI) Systems, 15(10), 1081-1090.

[64] Guerrier, P., & Greiner, A. (2008). A generic architecture for on-chip packet-switched interconnections. In Design, Automation, and Test in Europe (pp. 111-123). Springer, Dordrecht.

[65] Park, D., Eachempati, S., Das, R., Mishra, A. K., Xie, Y., Vijaykrishnan, N., & Das, C. R. (2008). MIRA: A Multi-layered On-Chip Interconnect Router Architecture. 2008 International Symposium on Computer Architecture, Beijing, 251-261.

[66] Chen, Y., Hu, J., & Ling, X. (2009). De Bruijn graph-based 3D Network on Chip architecture design. *2009 International Conference on Communications, Circuits and Systems*, Milpitas, CA, 986-990.

[67] Ramanujam, R. S., & Lin, B. (2009). A Layer-Multiplexed 3D On-Chip Network Architecture. *IEEE Embedded Systems Letters*, 1(2), 50-55.

[68] Rahmani, A. M., Liljeberg, P., Plosila, J., & Tenhunen, H. (2010, July). BBVC-3D-NoC: an efficient 3D NoC architecture using bidirectional bisynchronous vertical channels. In 2010 IEEE Computer Society Annual Symposium on VLSI (pp. 452-453). IEEE.

[69] Schonwald, T., Zimmermann, J., Bringmann, O., & Rosenstiel, W. (2007). Fully Adaptive Fault-Tolerant Routing Algorithm for Network-on-Chip Architectures. 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007), Lubeck, 527-534.

[70] Pasricha, S., & Zou, Y. (2011). A low overhead fault-tolerant routing scheme for 3D Networks-on-Chip. 2011 12th International Symposium on Quality Electronic Design, Santa Clara, CA, 1-8.

[71] Ahmed, A. B., & Abdallah, A. B. (2012). Low-overhead Routing Algorithm for 3D Network-on-Chip. 2012 Third International Conference on Networking and Computing, Okinawa, 23-32.

[72] Ahmed, A. B., Ahmed, A. B., & Abdallah, A. B. (2013). Deadlock-Recovery Support for Fault-tolerant Routing Algorithms in 3D-NoC Architectures. 2013 IEEE 7th International Symposium on Embedded Multicore Socs, Tokyo, 67-72.

[73] Ahmed, A. B., & Abdallah, A. B. (2014). Graceful deadlock-free fault-tolerant routing algorithm for 3D Network-on-Chip Architectures. Journal of Parallel and Distributed Computing, 74(4), 2229-2240.

[74] Somasundaram, K., Plosila, J., & Viswanathan, N. (2014). Deadlock-free routing algorithm for minimizing congestion in a Hamiltonian connected recursive 3D-NoCs. Microelectronics Journal, 45(8), 989-1000.

[75] Dubois, F., Sheibanyrad, A., Petrot, F., & Bahmani, M. (2013). Elevator-first: A deadlock-free distributed routing algorithm for vertically partially connected 3d-nocs. IEEE Transactions on Computers, 62(3), 609-615.

[76] LIU, Z., WU, N., ZHOU, L., & YAN, G. (2015). A Path Optimized Multicast Routing Algorithm for 3D Network-on-Chip. In Proceedings of the World Congress on Engineering and Computer Science 2015 (WCECS-2015), San Francisco, USA, 1.

[77] Ying, H., Jaiswal, A., Hollstein, T., & Hofmann, K. (2013). Deadlock-free generic routing algorithms for 3-dimensional Networks-on-Chip with reduced vertical link density topologies. Journal of Systems Architecture, 59(7), 528-542.

[78] Khan, M. A., & Ansari, A. Q. (2011). Quadrant-based XYZ dimension order routing algorithm for 3-D Asymmetric Torus Routing Chip (ATRC). 2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC), Udaipur, 121-124.

[79] Chen, K. C., Wu, A. Y. A., Lin, S. Y., & Hung, H. S. (2012). Topology-aware adaptive routing for non-stationary irregular mesh in throttled 3D NoC systems. IEEE transactions on parallel and distributed systems, 99(1), 1.

[80] Jiang, X., Zeng, L., & Watanabe, T. (2014). A sophisticated routing algorithm in 3d noc with fixed tsvs for low energy and latency. Information and Media Technologies, 9(4), 404-412.

[81] Ebrahimi, M., Salamat, R., Bagherzadeh, N., & Daneshtalab, M. (2015). A Fault Resilient Routing Algorithm for Sparsely Connected 3D NoCs. In MEDIAN 2015: The 4th Workshop On Manufacturable and Dependable Multicore Architectures at Nanoscale, Grenoble, France, 17-20.

[82] Ying, H., Jaiswal, A., Hollstein, T., & Hofmann, K. (2013). Deadlock-free generic routing algorithms for 3-dimensional Networks-on-Chip with reduced vertical link density topologies. Journal of Systems Architecture, 59(7), 528-542.

[83] Naghibijouybari, H., & Mohammadi, K. (2015). FT-Z-OE: A Fault-Tolerant and Low Overhead Routing Algorithm on TSV-based 3D Network on Chip Links. International Journal of Computer Applications, 115(2), 23-29.

[84] Viswanathan, N., Paramasivam, K., & Somasundaram, K. (2012). Exploring Optimal Topology and Routing Algorithm for 3 D Network on Chip. American Journal of Applied Sciences, 9(3), 300-308.

[85] Opoku Agyeman, Michael. (2015). A Low Overhead Fault Reporting Scheme for Resilient 3D Network-on-Chip Applications. Communications on Applied Electronics, 2, 43-48.

[86] Lin, C., Hsin, H., Chang, E., & Wu, A. A. (2013). ACO-based fault-aware routing algorithm for Network-on-Chip systems. SiPS 2013 Proceedings, Taipei City, 342-347.

[87] Wang, J., Gu, H., Yang, Y., & Wang, K. (2013). An energy-and buffer-aware fully adaptive routing algorithm for Network-on-Chip. Microelectronics Journal, 44(2), 137-144.

[88] Wang, C., & Bagherzadeh, N. (2014). Design and evaluation of a high throughput qos-aware and congestion-aware router architecture for network-on-chip. Microprocessors and Microsystems, 38(4), 304-315.

[89] ChaochaoFeng, Minxuan Zhang, Jinwen Li, Jiang JiangZhonghai Lu and Axel Jantsch. (2011). A Low-overhead Fault-aware Deflection Routing Algorithm for 3D Network-on-Chip. IEEE Computer Society Annual Symposium, Chennai, 19-24.

[90] Dahir, N., Al-Dujaily, R., Yakovlev, A., Missailidis, P., & Mak, T. (2012). Deadlock-free and plane-balanced adaptive routing for 3D networks-on-chip. Proceedings of the Fifth International Workshop on Network on Chip Architectures - NoCArc 12, Vancouver, British Columbia, Canada, 31-36.

[91] Dahir, N., Yakovlev, A., Al-Dujaily, R., & Mak, T. (2013). Highly adaptive and deadlock-free routing for three-dimensional networks-on-chip. IET Computers & Digital Techniques, 7(6), 255-263.

[92] Lee, J., & Choi, K. (2013). A deadlock-free routing algorithm requiring no virtual channel on 3D-NoCs with partial vertical connections. 2013 Seventh IEEE/ACM International Symposium on Networks-on-Chip (NoCS), Tempe, AZ, 1-2.

[93] Ebrahimi, M. (2013). Fully adaptive routing algorithms and region-based approaches for two-dimensional and three-dimensional networks-on-chip. IET Computers & Digital Techniques, 7(6), 264-273.

[94] Ebrahimi, M., Chang, X., Daneshtalab, M., Plosila, J., Liljeberg, P., & Tenhunen, H. (2013). DyXYZ: Fully Adaptive Routing Algorithm for 3D NoCs. 2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, Belfast, 499-503.

[95] Hu, J., & Marculescu, R. (2004, June). DyAD: smart routing for networks-on-chip. In Proceedings of the 41st annual Design Automation Conference (pp. 260-263). ACM.

[96] Hoda Naghibi Jouybari and Karim Mohammadi. (2014). A low overhead, fault tolerant and congestion-aware routing algorithm for 3D mesh-based Network-on-Chips. Microprocessors and Microsystems, 38(8), 991-999.

[97] Runge, A. (2015). Fault-tolerant Network-on-Chip based on Fault-aware Flits and Deflection Routing. Proceedings of the 9th International Symposium on Networks-on-Chip - NOCS 15, article No. 9.

[98] Feng, C., Lu, Z., Jantsch, A., Li, J., & Zhang, M. (2010). A reconfigurable fault-tolerant deflection routing algorithm based on reinforcement learning for network-on-chip. Proceedings of the Third International Workshop on Network on Chip Architectures - NoCArc 10, Atlanta, Georgia, USA, 11-16.

[99] Hsieh, K., Cheng, B., Gu, R., & Li, K. S. (2011). Fault-tolerant mesh for 3D network on chip. 2011 6th International Microsystems, Packaging, Assembly and Circuits Technology Conference (IMPACT), 5(2), 202-205.

[100] Jiang, X., & Watanabe, T. (2013). A novel fully adaptive fault-tolerant routing algorithm for 3D Network-on-Chip. 2013 IEEE International Conference of IEEE Region 10 (TENCON 2013), Xi'an, 1-4.

[101] Hosseini, A., Ragheb, T., & Massoud, Y. (2008). A fault-aware dynamic routing algorithm for on-chip networks. 2008 IEEE International Symposium on Circuits and Systems, Seattle, WA, 2653-2656.

[102] Werner, S., Navaridas, J., & Luján, M. (2016). A Survey on Design Approaches to Circumvent Permanent Faults in Networks-on-Chip. ACM Computing Surveys, 48(4), Article No. 59, 1-36.

[103] Loi, I., Angiolini, F., Fujita, S., Mitra, S., & Benini, L. (2011). Characterization and Implementation of Fault-Tolerant Vertical Links for 3-D Networks-on-Chip. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 30(1), 124-134.

[104] Deorio, A., Fick, D., Bertacco, V., Sylvester, D., Blaauw, D., Hu, J., & Chen, G. (2012). A Reliable Routing Architecture and Algorithm for NoCs. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 31(5), 726-739.

[105] Wang, C., Hu, W., & Bagherzadeh, N. (2013). Scalable load balancing congestion-aware Network-on-Chip router architecture. Journal of Computer and System Sciences, 79(4), 421-439.

[106] Gratz, P., Grot, B., & Keckler, S. W. (2008). Regional congestion awareness for load balance in networks-on-chip. 2008 IEEE 14th International Symposium on High-Performance Computer Architecture, Salt Lake City, UT, 203-214.

[107] Gutmann, R., Lu, J., Kwon, Y., Mcdonald, J., & Cale, T. (2001). Three-dimensional (3D) ICs: A technology platform for integrated systems and opportunities for new polymeric

adhesives. First International IEEE Conference on Polymers and Adhesives in Microelectronics and Photonics. Incorporating POLY, PEP & Adhesives in Electronics. Proceedings (Cat. No.01TH8592), Potsdam, Germany, 173-180.

[108] Horak, M. N., Nowick, S. M., Carlberg, M., & Vishkin, U. (2011). A Low-Overhead Asynchronous Interconnection Network for GALS Chip Multiprocessors. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 30(4), 494-507.

[109] Murali, S., Theocharides, T., Vijaykrishnan, N., Irwin, M., Benini, L., & Micheli, G. D. (2005). Analysis of Error Recovery Schemes for Networks on Chips. IEEE Design and Test of Computers, 22(5), 434-442.

[110] Ebrahimi, M., Daneshtalab, M., & Plosila, J. (2013). Fault-Tolerant Routing Algorithm for 3D NoC Using Hamiltonian Path Strategy. Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013, Grenoble, France, 1601-1604.

[111] Zhang, W., Hou, L., Wang, J., Geng, S., & Wu, W. (2009). Comparison Research between XY and Odd-Even Routing Algorithm of a 2-Dimension 3X3 Mesh Topology Network-on-Chip. 2009 WRI Global Congress on Intelligent Systems, Xiamen, 329-333.

[112] Kim, J., Nicopoulos, C., & Park, D. (2006). A Gracefully Degrading and Energy-Efficient Modular Router Architecture for On-Chip Networks. 33rd International Symposium on Computer Architecture (ISCA06), 4-15.

[113] Pavlidis, V. F., & Friedman, E. G. (2005). Interconnect delay minimization through interlayer via placement in 3-D ICs. Proceedings of the 15th ACM Great Lakes Symposium on VLSI - GLSVSLI 05, Chicago, Illinois, USA 20-25.

[114] Davis, W. R., Wilson, J., Mick, S., Xu, J., Hua, H., Mineo, C& Franzon, P. D. (2005). Demystifying 3D ICs: The pros and cons of going vertical. IEEE Design & Test of Computers, 22(6), 498-510.

[115] Bower, C. (2008). 3D Read-Out Integrated Circuits for Advanced Sensor Arrays. Handbook of 3D Integration, 689-701.

[116] Xie, Y., Loh, G. H., Black, B., & Bernstein, K. (2006). Design space exploration for 3D architectures. ACM Journal on Emerging Technologies in Computing Systems (JETC), 2(2), 65-103.

[117] Ramanujam, R. S., & Lin, B. (2009). A Layer-Multiplexed 3D On-Chip Network Architecture. IEEE Embedded Systems Letters, 1(2), 50-55.

[118] Dubois, F., Sheibanyrad, A., Petrot, F., & Bahmani, M. (2013). Elevator-first: A deadlock-free distributed routing algorithm for vertically partially connected 3d-nocs. IEEE Transactions on Computers, 62(3), 609-615.

[119] Chen, Y., Xie, L., Li, J., & Lu, Z. (2011). A deadlock-free fault-tolerant routing algorithm based on pseudo-receiving mechanism for Networks-on-Chip of CMP. 2011 International Conference on Multimedia Technology, Hangzhou, 2825-2828.

[120] Ebrahimi, M., Daneshtalab, M., & Plosila, J. (2013, March). Fault-tolerant routing algorithm for 3D NoC using Hamiltonian path strategy. In Proceedings of the Conference on Design, Automation and Test in Europe (pp. 1601-1604). EDA Consortium.

[121] Akbari, S., Shafiee, A., Fathy, M., & Berangi, R. (2012). AFRA: A low-cost high-performance reliable routing for 3D mesh NoCs. 2012 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, 332-337.

[122] Zhou, J., Li, H., Fang, Y., Wang, T., Cheng, Y., & Li, X. (2014). HARS: A High-Performance Reliable Routing Scheme for 3D NoCs. 2014 IEEE Computer Society Annual Symposium on VLSI, Tampa, FL, 392-397.

[123] Chen, Y., Xie, L., Li, J., & Lu, Z. (2011, July). A deadlock-free fault-tolerant routing algorithm based on pseudo-receiving mechanism for Networks-on-Chip of CMP. In 2011 International Conference on Multimedia Technology (pp. 2825-2828). IEEE.

[124] Schwiebert, L., & Jayasimha, D. N. (1995). Optimal fully adaptive minimal wormhole routing for meshes. Journal of Parallel and Distributed Computing, 27(1), 56-70.

[125] Liu, J., Harkin, J., Li, Y., & Maguire, L. (2015). Low cost fault-tolerant routing algorithm for networks-on-chip. Microprocessors and Microsystems, 39(6), 358-372.

[126] Sood, S. K., Sarje, A. K., & Singh, K. (2011). A secure dynamic identity based authentication protocol for multi-server architecture. Journal of Network and Computer Applications, 34(2), 609-618.

[127] Rathore, S. S., & Kumar, S. (2018). An Approach for the Prediction of Number of Software Faults Based on the Dynamic Selection of Learning Techniques. IEEE Transactions on Reliability, (99), 1-21.

[128] Sumit, **Sandeep Kumar**, K. Kumar, L. K. Singh, Evaluating Technologies for Reliable Software in Consumer Electronics: Survey of Component Failure Modeling based System Reliability Prediction Models, **IEEE Consumer Electronics Mag.**, In Press, 2017. SCI Impact Factor: 1.153

[129] Mansi Panwar, SD Samantaray, "An Improved E-DEEC Protocol using Periodic and Threshold-Sensitive Data Transmission in Heterogeneous Wireless Sensor Network"

published in International Journal of Computer Science Engineering (IJCSE), ISSN: 2319-7323, Vol. 4 No.04 ,July 2015, pp.155-161

[130] Sain, M., Bruce, N., Kim, K. H., & Lee, H. J. (2017, February). A communication security protocol for ubiquitous sensor networks. In 2017 19th International Conference on Advanced Communication Technology (ICACT) (pp. 228-231). IEEE.

BIOGRAPHY OF THE AUTHOR

I have an experience approx 10+ years in Research and Teaching at Post Graduation Level in the area of IT & computer science. My area of interest are Networks on chip, Big data analytics, Database Management system . Presently, I am associated with IMT Ghaziabad as Research Associate.

PUBLICATIONS DURING PH.D

1. **Evaluation of Scalable 3-D Network on Chip Architectures**

   Tyagi, Sapna, and Amit Agarwal. "Evaluation of scalable 3-D Network on Chip architectures." Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on. IEEE, 2015

2. **Exploring 3D Network-on-Chip Architectures and Challenges**

   Tyagi, Sapna, et al. "Exploring 3D Network-on-Chip Architectures and Challenges." Computer and Applications (ICCA), 2017 International Conference on. IEEE, 2017.

3. **An automated fault-tolerant route discovery with congestion control using TFRF model for 3D network-on-chips**

   S. Tyagi, A. Agarwal, V. Avasthi, and P. Maheshwari, International Journal of Communication Networks and Distributed Systems(Inderscience Publishers Ltd.) (In Publication) (Scopus indexed)

4. **A comprehensive perspective on 3D NOC Routing Challenges**

   S. Tyagi, A. Agarwal, V. Avasthi, and P. Maheshwari, International Journal of Computer Engineering and Applications, Volume XII, Issue V, May 18, www.ijcea.com ISSN 2321-3469 (UGC)

5. **Evolution & Performance study of 2D NoC Topologies**

   Sapna Tyagi, Amit Agarwal, Vinay Avasthi, Piyush Maheshwari International Journal of Engineering and Advanced Technology(IJEAT) (accepted in publication , Scopus indexed)