Performance-Aware Test-Time Optimization Schemes for Analysis of Logic Level Faults in Channels of On-Chip Networks

Biswajit Bhowmik

Performance-Aware Test-Time Optimization Schemes for Analysis of Logic Level Faults in Channels of On-Chip Networks

Thesis submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

by

Biswajit Bhowmik

Under the supervision of

Santosh Biswas
Jatindra Kumar Deka

Department of Computer Science and Engineering
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
Guwahati, Assam – 781039, India
April, 2018

This Work Is Dedicated

To

My Mother

I Love You Maa With All My Heart...

Declaration

I certify that

a. The work contained in this thesis is original and has been done by myself under the general supervision of my supervisors.

b. The work has not been submitted to any other institute for any degree or diploma.

c. Whenever I have used materials (data, theoretical analysis, results) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.

d. Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Place: IIT Guwahati
Date: April 29, 2018

Biswajit Bhowmik
Research Scholar
Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati, Assam - 781039, India

CERTIFICATE

This is to certify that the thesis entitled **"Performance-Aware Test-Time Optimization Schemes for Analysis of Logic Level Faults in Channels of On-Chip Networks"** being submitted by **Mr. Biswajit Bhowmik** to the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati, is a record of bona fide research work under our supervision and is worthy of consideration for the award of the degree of Doctor of Philosophy of the Institute.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree or diploma.

Dr. Santosh Biswas
Associate Professor
Dept. of Computer Sc. and Engg.
Indian Institute of Technology Guwahati
Guwahati, Assam - 781039, India

Place: IIT Guwahati
Date: April 29, 2018

Prof. Jatindra Kumar Deka
Professor
Dept. of Computer Sc. and Engg.
Indian Institute of Technology Guwahati
Guwahati, Assam - 781039, India

Place: IIT Guwahati
Date: April 29, 2018

Abstract

Today manycore, multiprocessor systems-on-chip (MPSoCs) have been introduced to cope with the growing demand for high-speed communication requirements of intensive-computing applications. However, in spite of rapid advancements in deep-submicron (DSM) technology and the seamless integration of intellectual property (IP) modules in the SoCs, these bus-based interconnection architectures have become unable to meet the performance requirements – bandwidth, throughput, latency, power, etc. in the applications where high-performance computation and communication is the dominant consideration. In other words, such SoCs often fail to sustain high-volume computation and high-speed communication among their components due to the use of global buses as the interconnects. The network-on-chip (NoC) as an alternate prevalent interconnection infrastructure has been continuously occupying the space of the SoC. An NoC comprises a large number of IP cores, routers, and high-speed channels (interconnects) that construct a structure (topology) spanning across the chip. However, aggressive CMOS scaling expedites interconnect and transistor wear-out, shortening the lifespan of these basic components which are often vulnerable to a number of manufacturing and transient faults due to aging, physical defects, or hostile attacks invoked by malicious third parties. For instance, basic classical logic level faults treated as the manufacturing faults, such as stuck-at, open, and shorts in NoC channels cause various system-level failures and subsequent degradation of reliability, yield, and performance of the computing platform. One approach to tackle channel-faults in NoCs is to replace the faulty channel-wires with spare wires. Such scheme is not cost-effective as the area overhead is substantially increased. Another approach is to exercise a fault-tolerant routing algorithm that directs traffic (application data packets) over channels by avoiding the faulty wires or an alternative fault-free path in order to connect the source and destination nodes, keeping the NoC functional. However, most of the fault-tolerant schemes do not consider any test mechanism for channels. One prerequisite of these approaches includes the knowledge about the health status of the channels which can only be inferred by an effective test method. Most of the commonly practiced approaches acknowledge numerous disadvantages in terms of test issues like high test volume, high silicon area overhead, high test time, low fault coverage, and lack of scalability.

This dissertation pursues efficient self-test approaches for manufacturing faults including a little emphasis on the transient faults and other logic level faults, e.g., packet deadlock, in the NoC channels in order to improve the yield and reliability in NoC-based communication systems. The main goal of this dissertation is to develop an environment for the test-time minimization problem in addition to associated issues, such as test area overhead, performance overhead at the runtime, etc. In order to reduce the test cost in terms of low test time, low area overhead, low-performance overhead, etc., the proposed test approach is divided into two parts: test algorithm and test scheduling. As the first pillar towards the goal, an on-line, distributed built-in-self-test (BIST) oriented test algorithm is proposed, that with the help of a test module (must be included in a modern NoC) has the capability in detecting a channel fault and identifying the faulty wires from the channel. As the second pillar to reach the goal, a suitable test scheduling scheme is proposed, which makes the present test solution scalable with respect to different network architectural characteristics: network size, channel width, and network type, i.e., large-scale NoC architectures in general. This dissertation contains five contributed works. First three contributions explore the test of stuck-at, open, and short faults in NoC channels. Here, the channels are assumed to follow the single fault model. The fourth contribution includes the test of manufacturing and transient faults in the channels. The final contribution includes the test of coexistent manufacturing channel-faults. In the last two contributions, a channel can be assumed to follow the multiple fault model. In all the contributions, experimental results reveal that the proposed solutions impose less test time, area, and performance overhead as compared to existing testing techniques.

Acknowledgments

First and foremost, I would like to convey my deepest gratitude and prayer to my mother, whose love, care, bless, support, inspiration, and sacrifice have made my success possible, provide me the liberty to be fearless and explore new things in life.

With a sense of adoration, I do express many thanks and a lot of love to Sovana who always guided me in the proper way, and grew interests in life. I cannot forget her continuous encouragements that keep me energetic. Probably, I would have never decided to pursue a PhD without her encouragement. Without her devotion and effort, it would also become very difficult to complete the thesis in time.

Especially, I would like to be evidence for my honor to Dr. Rabindranath Bhowmik (Sejda), Assistant Professor, Department of Physics, University of Pondicherry, and Mr. Pradip Kumar Maity (my favorite Pradipda) who put the first brick in building my career, and helped and supported me by all the ways to become what I am at present.

I would like to express my special appreciation and thanks to my supervisors, Prof. Santosh Biswas and Prof. Jatindra Kumar Deka, whose joy and enthusiasm, selfless time and care, meticulous and contagious attention helped me to grow as a researcher. This thesis is a culmination of a perfect working relationship with my supervisors. Their brilliant advice and constant supports towards a quality research as well as on my career always remain priceless.

With profound glee, I extend my special thanks to Prof. Bhargab B. Bhattacharya, Advanced Computing and Microelectronics Unit, Indian Statistical Institute Kolkata, India – 700108, who provided me immense support during my journey towards PhD, and generously paved the way for my development as a research scientist. I am very grateful to him for sharing his knowledge and novel ideas with me and bearing his valuable time towards preparing my research manuscripts.

I am also highly grateful to my Doctoral Committee members, Prof. Diganta Goswami, Prof. Arnab Sarkar, Prof. Partha Sarathi Mandal for sharing their insightful comments and suggestions on my research work. I express my sincere thanks to Prof. Sukumar Nandi, Prof. Hemangee K. Kapoor, Dr. Deepanjan Kesh, Dr. Sushanta Karmakar, Prof. Purandar Bhaduri, Dr. V. Vijaya Saradhi, Dr. Arijit Sur, Dr. T. Venkatesh, Prof. Pinaki Mitra, Prof. Gautam Barua, Prof. S. V. Rao, Dr. Benny George K, and other faculty members of the

Department of Computer Science and Engineering, for providing a nice research environment in the department, and support my research works in many ways. I also thank department's Technical Officers Mr. Bhriguraj Bora and Mr. Nanu A. Kachari, Technical Superintendents Raktajit Pathak, Nava Kumar Boro, and Pranjit Talukdar, and Administrative Staffs Monojit Bhattacharjee, Gauri Khuttiya Deori, and Prabin Bharali for extending their helping hands in solving many issues which I faced during my PhD.

I cherished my stay in the institute largely due to my friends. I thank Niladri Sett, Chandan Kalita, Sandip Chakraborty, Pradeep Kumar Biswal, Subhrendu Chattopadhyay, Rahul Gangopadhyay, Shirshendu Das, Shounak Chakraborty, Mayank Naresh Agarwal, Shuvendu Rana, Sibaji Gaj, Satish Kumar, Awnish Kumar, Mrityunjay Singh, Madhusudan Paul, Rajesh Devraj, Durgesh Kumar, Mousum Handique, Parikshit Saikia, Lalatendu Behera, Ranajit Senko, Manas Kumar Panda, Amit Kumar Srivastava, Kunwer Mrityunjay Singh, Sanjit Kumar Roy, Manojit Ghose, Pradeep Kumar Sharma, Rakesh Pandey, Sukarn Agarwal, Piyoosh Purushothaman Nair, Sangeet Saha, Akash Anil, Subrata Tikadar, Nagaraj Adiga, Bidisha Sharma, Shrestha Tripathy, and many others for their support, encouragement, and delightful company. I am also thankful to Nikhil Bandhu Pal, Graphite India Ltd., Durgapur, and my colleagues and friends of Bengal College of Engineering and Technology Durgapur, West Bengal-713212, who encouraged me to pursue this PhD.

The development of this thesis would not have been possible without financial support from the MHRD, Govt. of India. I would also like to thank the Ministry of Electronics and Information Technology (Meity), Govt. of India for financial support from the budget of the sponsored project titled "On-Line Testing of Complex VLSI Circuits using Failure Detection and Diagnosis Theory of Discrete Event Systems" to publish my research papers in many prestigious international conferences and after the tenure of my MHRD fellowship was over.

Lastly, I express my deepest worship to "Eswar", the supernatural being worshiped by people, and nature, where the basic source of my life energy resides: words cannot express how grateful I am.

Before I finish, I would like to share that when one writes a research article and thesis then he realizes the true power of the Latex and MS-Word from grammar checks to replace all. Simply, without this software as a document preparation system, this thesis would not possibly be written. Thanks to the American computer scientist Prof. Leslie Lamport, and the Philanthropist Bill Gates and his Microsoft Corporation!

Place: IIT Guwahati
Date: April 29, 2018

Biswajit Bhowmik

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

MPSoCs	Multiprocessor Systems-on-Chip
DSM	Deep-Submicron
IP	Intellectual Property
NoC	Network-on-Chip
BIST	Built-In-Self-Test
μm	Microns
nm	Nanometer
FPGA	Field Programmable Gate Array
MEMS	Micro Electro Mechanical Systems
RF	Radio Frequency
SoC	System-on-Chip
ITRS	International Technology Roadmap for Semiconductors
EMI	Electromagnetic Interference
HPCC	High Performance Computation and Communication
P2P	Point-to-Point
CMP	Chip Multiprocessor
NI	Network Interface
QoS	Quality of Service

TAM	Test Access Mechanism
RLBs	Routing Logic Blocks
FIFO	First-In-First-Out
CSSAFs	Co-Existent Short and Stuck-At Faults
HCI	Hot Carrier Injection
ECC	Error Correction Code
ARQ	Automatic Repeat Request
FEC	Forward Error Correction
E2E	End-to-End
Subnet	Subnetwork
ASIC	Application-Specific Integrated Circuit
EDA	Electronic Design Automation
UART	Universal Asynchronous Receiver/Transmitter
CPUs	Central Processor Units
HDL	Hardware Description Language
TDM	Time Division Multiplexing
SAF	Store and Forward Switching Technique
VCT	Virtual Cut Through Switching Technique
WH	Wormhole Switching Technique
NF	Negative First Routing
WF	West First Routing
GALS	Globally Asynchronous Local Synchronous
CLICHE	Chip-Level Integration of Communicating Heterogeneous Elements
C-Mesh	Concentrated Mesh
BFT	Butterfly Fat Tree

EBFTI	Extended BFT Interconnection
FBFT	Flattened BFT
R_i	The i^{th} Router
(R_i, R_j)	An Interswitch Channel
(R_i, C_i)	A Local Channel
C_i	The i^{th} IP Core
MoT	Mesh-of-Tree
EMI	Electromagnetic Interference
NBTI	Negative Bias Temperature Instability
$d/0$	d is tied to Logic-0
$d/1$	d is tied to Logic-1
SA0	Stuck-at-0 Fault
SA1	Stuck-at-1 Fault
SSA	Single Stuck-at Fault
MSA	Multiple Stuck-at Fault
I/I	Input to Input
I/O	Input to Output
O/O	Output to Output
MAF	Maximum Aggressor Fault
CUT	Circuit Under Test
ATE	Automatic Test Equipment
FTVs	Functional Test Vectors
HRs	Hardware Redundancies
TPG	Test Pattern Generator
TRA	Test or Output Response Analyzer

NoCUT	An NoC Under Test
IP-CUT	IP Core Under Test
RAM	Random Access Memory
SI/SO	Serial-In, Serial-Out
ATPG	Automatic Test Pattern Generation
DPCFs	Dual-Port Coupling Faults
BE	Best Effort
GS	Guaranteed Service
DS	Differentiated Service
bps	Bits Per Second
MAF	Maximal Aggressor Fault
CADEC	Crosstalk Avoiding Double Error Correction Code
TDGs	Test Data Generators
TEDs	Test Error Detectors
RUT	Router Under Test
TIs	Testing Issues
A0	All-Zero Test Vector
A1	All-One Test Vector
W1	Walking One Sequence
TM	Test Module
n	Channel width.
l_k	A channel-wire of single bit width; $1 \leq k \leq n$
f_{sa0}	An SA0 fault on an l_k
f_{sa1}	An SA1 fault on an l_k
x_k	Single/multiple occurrences of an SA0 fault on the l_k

#SA0	Size of SA0 faults in a channel
y_k	Single/multiple occurrences of an SA1 fault on the l_k
#SA1	Size of SA1 faults in a channel
#SA	Size of SAFs in a channel
#Ch	Number of unidirectional channels in an NoC
#SAF	Size of SAFs in an NoC
eop	End of Packet Signal
bop	Beginning of Packet Signal
DWs	Data Wires of a Channel
CWs	Control Wires of a Channel
HWs	Handshake Wires of a Channel
FDM	Fault Diagnosis Module
TSG	TRA's Signal Generator
I-TPG	Integrated TPG
MWU	Multicast Wrapper Unit
PE	Payload Error
ME	Misrouted Error
TE	Timeout Error
TRs	Number of Test Rounds
Tits	Number of Test Iterations
D_{odd}	Number of Odd Diagonal Levels
D_{even}	Number of Even Diagonal Levels
$\mathbb{G}$	A Graphical Representation of an NoC
$\aleph$	The Set of Nodes in an NoC
$\aleph_i$	The i^{th} Node in an NoC

$\mathfrak{C}$	A Set of NoC Channels
E_k	The k^{th} Channel
$\Im$	A Matrix of an NoC
r	The Number of Rows in the $\Im$
c	The Number of Rows in the $\Im$
γ	Symbol for Labeling Diagonal Level Numbering
$\Re$	The Set of Distinct Colors
α	A Color
β_α	A Set of Test Iterations
T_{ch}	The Test Time Needed for a Channel
T_{gen}	The Test Set Generation Time
T_{tpo}	The Test Set Organization Time
T_{tpt}	The Test Set Transportation Time
T_{tra}	The Test Response Set Analysis Time
T_{it}	The Test Time Needed in an Iteration
$T_{n/w}$	The Test Time Needed for the Channels of an NoC
2D	Two Dimensional
$M \times N$	The Size of an NoC
#R	The Number of Routers in an NoC
#C	The Number of IP Cores in an NoC
#Ch	The Number of Channels in an NoC
#R-R	The Number of Interswitch in an NoC
#R-C	The Number of Local Channels in an NoC
#W	The Number of Communication Wires in an NoC
TPG-R	The TPG Placed in a Router

TPG-C	The TPG Placed in an IP Core
TRA-C	The TRA Placed in a Router
TRA-C	The TRA Placed in an IP Core
GC	Gate Count
LCM	Link or Test Coverage Metric
FCM	Fault Coverage Metric
PIR	The Packet Injection Rate
XY	A Routing Algorithm
SONoC	Square-Octagon NoC Architecture
CLCM	Cumulative LCM
CFCM	Cumulative FCM
D-Model	The Diagonal Node Selection Based Test Approach
2×2-Model	The 2×2 Subnet Selection Based Test Approach
P-2×2-Model	The Partial 2×2-Model
S-Model	The Sequential Router Selection Based Test Approach
H-Model	Hierarchical Test Approach
OE-Model	The Odd-Even Node Selection Based Test Approach
COFs	Channel-Open Faults
f_o	Notation for an Open Fault
f_q	Single or Multiple Instances of an Open Fault
O_{ch}	The Size of Open Faults in a Channel
O_{NoC}	The Size of Open Faults in an NoC
BFS	Breadth First Search Traversal
D_{max}	The Maximum Distance from the Root to a Leaf Node in an Execution Tree
SD-Model	The Self-Diagnosis Test Approach

4×4-Model	The Iterative 4×4 Subnet Selection Based Test Approach
2hop-Model	The Two Hop Node Selection Based Test Approach
Diag-Model	The Diagonal Node Activation Model
SFM	Single Fault Model
HPC	High-Performance Computing
$\delta^+(N_i)$	The Out-Degree of a Node N_i at an Odd Diagonal Level.
$\delta^+(N_i)$	The Out-Degree of a Node N_i at an Even Diagonal Level
$\mathbb{G}^+$	The Instance of $\mathbb{G}$ Constructed with respect to $\delta^+(N_i)$
$\mathbb{G}^-$	The Instance of $\mathbb{G}$ Constructed with respect to $\delta^-(N_i)$
$\mathbb{C}_4^+$	The Cluster Set of the Nodes with $\delta^+(N_i) = 4$
$\mathbb{C}_3^+$	The Cluster Set of the Nodes with $\delta^+(N_i) = 3$
$\mathbb{C}_2^+$	The Cluster Set of the Nodes with $\delta^+(N_i) = 2$
$\mathbb{C}_1^+$	The Cluster Set of the Nodes with $\delta^+(N_i) = 1$
$\mathbb{C}_4^-$	The Cluster Set of the Nodes with $\delta^-(N_I) = 4$
$\mathbb{C}_3^-$	The Cluster Set of the Nodes with $\delta^-(N_i) = 3$
$\mathbb{C}_2^-$	The Cluster Set of the Nodes with $\delta^-(N_i) = 2$
$\mathbb{C}_1^-$	The Cluster Set of the Nodes with $\delta^-(N_i) = 1$
f_{sab}	A Short Fault Between Wires l_a, l_b
g	Fault Group
x_g	The Single or Multiple Instance of Intra-Channel Shorts
y_g	The Single or Multiple Instance of Inter-Channel Shorts
#S1	The Size of Intra-Channel Shorts
m	The Number of Channels of a Node
#S2	The Size of Inter-Channel Shorts at a Node
#S	The Size of Channel Shorts in an NoC

LGs	Logic Gates
TM_1	The TM at an IP Core
TM_2	The TM at a Router
C-Model	The Cluster-Set Driven Test Model
LSB	Least Significant Bit
T_{node}	The Test Time at a Node for Its Channels
T_{subnet}	The Time Required to Test Channels of a Subnet
E_{tm}	Energy Dissipated Per Flit by the TMs
E_{ch}	Energy Dissipated Per Flit by a Channel Under Test
α_{tm}	The Signal Activity of a TM
α_{ch}	The Signal Activity of a Channel
C_{tm}	The Total Capacitance of a TM
C_{ch}	The Total Capacitance of a Channel
υ	The Number of Flits in a Test Packet
E_{pkt}	The Energy Consumed by a Test Packet
$\wp$	The Total Number of Test Packets Transported on a Channel
$\overline{E_{pkt}}$	The Average Energy Per Test Packet
Q-Model	The Quadrant Test Model
QoRP	Quality of Reliability and Performance
$Xckt$	Integrated Extra Circuit Block
MFM	Multiple Fault Model
DFs	Diagnosable Faults
NDFs	Non-Diagnosable Faults
κ-octagon	Interconnection of $\kappa \geq 2$ Basic Octagon Architectures
B	Bridge Node in a κ-Octagon

$\Phi(\aleph)$	Wiring Cost of a κ-Octagon
$\Psi(T_{n/w}, \aleph)$	The Test Cost Function
$\Upsilon(\kappa)$	The Size of an *e-Damaru*
λ	Maximum Routing Distance Between Two Nodes on an NoC
E2E-M	End-to-End Test Approach
SC-M	Single Channel Selection Based Test Approach
Port-M	Router-Port Based Test Model
3D	Three Dimensional
TSVs	Through-Silicon Vias
WiNoC	Wireless NoC
mm	Millimeter
GHz	Giga Hertz
CNTs	Carbon Nanotubes
PhNoC	Photonic NoC

Chapter 1

Introduction

1.1 Introduction

Over the past two to three decades, the rapid advancement in semiconductor manufacturing technology has witnessed the decrease in feature size from four microns (4 μm) to forty five nanometers (45 nm). This continuous shrinkage of feature size of embedded systems has made a dramatic impact on design and test because millions to billions of transistors that are running in high operating frequencies are contained in the embedded systems. The wide range of designs may include analog, digital, optical, memory, field programmable gate array (FPGA), micro electro mechanical systems (MEMS), mixed-signal, and radio frequency (RF) circuits [1]. Simultaneously, with this revolutionary changes in the design methodologies, the multi-scale, multicore embedded systems, e.g., system-on-chip (SoC) platforms are supporting large number of embedded processing cores involving a set of heterogeneous or homogeneous components. Each component may have irregular and regular blocks. The SoC platform then results to a multiprocessor SoC (MPSoC) and implies integration of many intellectual property (IP) blocks (processing cores) seamlessly [2, 3] to boost the performance and efficiency up in MPSoCs for supporting wide range of applications. For example, Tilera [4] has launched various high-end MPSoC products to support large-scale high performance computing and communication applications – complex and advanced networking, digital multimedia, cloud computing, wireless infrastructures [5–7], etc.

This chapter first discusses the challenges in SoC integration, the paradigm shift of an SoC to an on-chip network, various research dimensions, and the necessity of testing channels in on-chip networks. Subsequently, literature survey, research overview, motivation and research problem formulation, the list of major contributions and the organization of the thesis are described.

1.2 SoC Integration and Its Challenges

Arbitrated shared buses are the communication backbone in an SoC architecture. As the advantages, this architecture has simple topology, low area cost, and extensibility. As the disadvantage, only one communication at a time is allowed by a shared bus while preventing other buses in the hierarchy. Consequently, this architecture does not scale the system performance if more IP cores are embedded. In other words, with the increase in the number of IP cores, traditional bus-based interconnections in SoC designs prevent large and complex applications from satisfying the required performance. Subsequently, shared global buses in SoC-based systems are unable to fulfill sufficient bandwidth demand since all IP cores share this bandwidth. Even the problem of bandwidth sharing remains same if the shared bus is segmented for usage [8, 9]. According to the International Technology Roadmap for Semiconductors (ITRS) report in 2001, the delay on every process generation is reduced with local wires. This delay, on the contrary, with global wires increases exponentially or at best linearly on inserting repeaters. This delay is caused due to intrinsic parasitic resistance and capacitance of a relatively long bus. Now, as the IP cores share this bus, they add capacitance to the bus resulting enhanced delay. Also, long wires introduce many signal integrity problems, such as crosstalk, electromagnetic interference (EMI), etc. Moreover, global wires along their drivers and repeaters signify overall power budget of an SoC. In the deep submicron (DSM) era, on-chip communication efficiency in terms of performance and cost has become a key factor. Major challenges towards the goal is to turn an SoC-based architecture into a structured, reusable, scalable interconnection architecture that can meet the demand of high performance computation and communication (HPCC).

1.3 SoC to NoC: A Paradigm Shift

Researchers from academia and industries have haunted for the high speed communication backbone in the next-gen MPSoCs that would support new inter-core HPCC demands. Dedicated channels in point-to-point (P2P) architectures with limited IP cores can be considered to be a good alternative to a global bus-based SoC and achieve certain performance in terms of latency, power consumption, and so on. The number of channels however exponentially increases with the increase in the number of IP cores. Consequently, a routing problem may originate in such large systems. Few limitations of the buses though can be overcome with a centralized crossbar switch but the connection of a large number of IP cores to the switch makes the design ultimately unscalable. Therefore, a traditional bus-based SoC with certain number of IP cores can be considered as an intermediate solution for the expected system performance [8, 10]. But, the tendency of increasing IP cores in a many-core regime is noticed since last decade due to shifting of the focus from computation to communication. It is necessary to search for a systematic design of communication backbone in

chip multiprocessor (CMP) architectures, i.e., MPSoCs. The solution subsequently has turned an SoC into a network-on-chip (NoC) which is commonly known as an "on-chip network", "on-chip communication network", or "on-chip interconnection network", and so on. An NoC has therefore emerged as a revolutionary scalable communication mechanism in the place and acts as a viable solution over SoC's communication bottleneck [11, 12].

It can be mentioned that the concept of on-chip interconnection network (NoC) has been borrowed from off-chip interconnection networks. In later class of networks, every chip implements a router and the bandwidth is typically lower. Also, these networks are constrained by bit width because every extra bit costs one more pin. Additionally, routers are explicitly connected by board traces which aggravate the synchronization problem and affect the overall latency [13, 14]. The concept of an NoC for the design of modular and scalable communication architectures is first suggested by Benini *et al.* [15, 16]. Authors have used the traditional computer networking mechanism for communicating application data in the form of packets via routing paths across several routers and channels. The IP cores, each wrapped with a network interface (NI), as the source and destination thus communicate via router-based network. An NoC architecture is partitioned into three basic building blocks which are processing elements commonly known as IP cores, data forwarding switches commonly called routers, and data transmission medium commonly known as channels (also called as on-chip channels, on-chip network channels, or on-chip interconnection channels). These components are interconnected in various ways resulting a topology, such as mesh, torus, octagon and so on [8, 17, 18]. Various NoC topologies as the on-chip communication or interconnection networks are being studied for a stable foundation of design approaches. Today NoC is therefore an emerging topic for the research communities.

1.4 Research on NoC Architectures

Researches on NoC designs are fundamentally classified into four major and broad dimensions-*communication infrastructure, communication paradigm, evaluation framework, and application mapping*, as suggested in [10, 19–21]. The first research dimension focuses on the communication infrastructure that acts as the backbone for NoCs. In this dimension, researchers show interest to address different key design aspects of NoCs, such as topology selection, router design in terms of proper buffer utilization, clocking strategy, channel configuration in terms of determining its width, and floor planning and layout design. Once the communication infrastructure is finalized, it is necessary to design a communication scheme between source and destination IP cores via the established network. The second research dimension deals with the communication paradigm that sets up a number of policies, such as routing strategies, switching schemes, congestion control mechanisms, quality of service (QoS), reliability and fault tolerance issues, and thermal and power management. As the NoC topologies

in complex systems are integrated with a large number of IP cores that communicate via routers and channels, one must have the clear idea about the performance achieved by these networks at the pre- and post-manufacturing stages of the systems. The potential faults must be detected and drawbacks, if any, must be identified in order to avoid huge loss in terms of severe performance degradation after getting NoCs in the silicon chips. Next research dimension pays attention to the evaluation framework designs in presence of the stochastic and application-specific traffic. The job of the NoC simulators, such as Noxim [22] is to replicate similar behavior as seen in the actual NoC. Often synthetic traffic instead of actual application traffic patterns are used in these simulators to mimic the behavior. With the confidence gained in the network performance through simulations, the designers can soon estimate the area and power consumption in the network because a significant portion of the overall SoC cost budget is taken by these factors that influence the network performance. The final research dimension addresses the mapping issues of IP cores onto regular and irregular NoCs to achieve the required performance. An important problem of this dimension is the performance, traffic, and energy-aware task scheduling for heterogeneous NoCs. Besides the four research dimensions, another important aspect for NoCs is testing of their basic components as it takes major part in any system development process. It may become complicated whence test data size becomes voluminous and are necessary to be transported from system inputs to outputs during testing of an NoC component. This gives rise to an optimization problem in terms of the test scheduling. The testing and related scheduling schemes occupy a set of aspects designated for the second and third research dimensions that support QoS, reliability, fault-tolerance, and accepted network performance level.

1.5 Necessity of NoC Channel Testing

NoCs are regarded as the revolutionary approach for addressing communication demands of MPSoCs. Source and target IP cores in an NoC communicate by exchanging data packets via a routing path of routers and channels. The routers forward the packets while channels transport them. The high bandwidth requirement of MPSoCs is overcome by the placement of huge number of metallic wires as communication channels in NoCs. It can be found that communication channels compose sizable part in NoCs. On the other hand, continuous shrinking of chips on a die results to many transient and manufacturing faults in NoC channels as well as other NoC components. Consequently, these defects may put the NoC into various system failure modes that appear as channel errors and subsequently have direct influence on the system performance degradation, and even may lead to complete system failure [23]. Reliability, yield, and fault tolerance issues have become major concerns in the current as well as future generation NoC-based SoC communication systems because of hefty use of DSM technologies [24].

Reliability during communication in a system can be ensured through testing and fault tolerance mechanisms. The testing mechanism defines the reliability with reference to manufacturing and operational defects. On the other hand, the fault tolerance guarantees the reliability with respect to avoiding the faulty parts detected by testing. One has to unwillingly accept the fact that some manufacturing products, say NoCs are expected to be faulty or may become faulty during the operational time. Testing of the NoCs is divided into three phases: testing of IP cores, testing of routers, and testing of channels. Test of each of these components needs to rely on each other. Generally, these phases are accomplished separately, otherwise the process becomes complex. Test of NoC channels must be attempted first. Because, testing of other phases needs to communicate test data over the channels. Therefore, the correctness of the channels must be ensured before conducting a test for IP cores and routers. One can also use the knowledge of channel's health status for the verification of correct implementation, manufacturing, and operation of NoCs. Furthermore, NoCs play a central role while system-reliability is under consideration. The NoCs therefore must be operating correctly both at normal and test modes. Thus, an NoC should at least include a test mechanism for detecting manufacturing and operational faults in order to enhance not only system reliability, serviceability, and dependability, but also better yield and increased system lifetime [24].

1.6 Literature Review

Different aspects of design and implementation of an NoC architecture have been addressed in [2,5,11,25]. Note that an NoC architecture primarily consists of three basic building blocks: IP core, router, and channel. These components may malfunction because of the presence of various transient or permanent faults [26, 27]. Radiation-induced soft errors, crosstalk, voltage-induced delay errors often result in transient faults in an NoC system [28,29], whereas manufacturing defects, hardware aging, thermal and physical stress may cause permanent faults such as logical stuck-at and shorts [5, 15]. In order to perform reliable computation with MPSoCs, the underlying NoC fabric that is being engaged must be guaranteed to be fault free.

Testing for faults in an NoC architecture should cover these three components:IP cores, routers, and channels. In other words, the test of the NoC architecture characteristically is classified into three broad areas: (a) testing of IP cores, (b) testing of routers, and (c) testing of communication channels. The testing of IP cores is based on the reuse of the NoC infrastructure as a test access mechanism (TAM) and has been studied in the past [30–32]. On the other hand, test strategies for routers in terms of testing of arbiter, routing logic blocks (RLBs), I/O ports, and first-in-first-out (FIFO) buffers are well studied [6, 33–35]. In both cases, the test methods assumed the correctness of the communication channels to carry test

data and test responses. Therefore, the sequence of testing NoCs' basic components matter and must be prioritized in the testing cycle. The testing of NoC channels in the priority sequence must be conducted earlier than the routers which must again be done before the cores because channels are the primary means of transportation for both test and application data, and involve significant portion of the network area [36]. One must then be ensured about the correct functionality of the channels before using them as the test instrument in testing of routers and cores. In this thesis, different manufacturing channel faults in an NoC are primarily considered. These channels suffer from poor observability and controllability because of their placement and density. Although all channels connected to a node can exercise the same test set, optimization of test time, fault coverage metrics, and performance overhead pose a challenge while applying the same test set at the node [37, 38].

The search for designing a suitable test paradigm for permanent (manufacturing) faults in NoC channels had been a topic of research for quite some time. For example, Cota *et al.* [37, 38] proposed an off-line and high fault-coverage test model that addresses pairwise shorts in the channels of a 2×2 neighborhood. The four IP cores simultaneously transmit the test sets to each other separated at four hops in the neighborhood. A hop is defined as the single channel length. The model can be used to test larger size mesh NoCs by testing several 2×2 sub-meshes that cover it for the detection of short-channel faults. Although, the 2×2-Model provides high coverage metrics, it requires high test time. The situation worsens when the model is applied in the on-line mode as it needs increased number of test iterations. The same test model is extended by Herve *et al.* [39]. The extended post-burning off-line test method accounts for co-existent short and stuck-at faults (CSSAFs) in channels of a 2×2 network neighborhood. The method as before, incurs both high test area overhead and time due to the analysis of test responses after traveling four hops. Similarly, the test time is increased while the model is applied in on-line mode. Moreover, it works with traditional mesh-type NoCs only due to this 2×2 test configuration. Today, NoC-based systems often have both conventional topologies like mesh networks and unconventional topologies like octagon, spidergon networks. If one wants to account channel faults in unconventional NoCs, the 2×2-Model should not be preferred, rather one may employ the test model discussed in [40, 41] for torus NoC. In this scheme, the channel-short faults on a neighborhood of 2×1 that consists of an interswitch channel and its adjacent local channels get tested. Multiple instance of this neighborhood is applied concurrently for detecting channel shorts on a larger torus network. In the on-line mode, one can iterate this 2×1 neighborhood to cover channel faults on a general network. Although the scheme enhances the scalability issue irrespective of network size and type but results in high test time and hardware area overhead on these networks. Strano *et al.* [42] have proposed a self-diagnosis test method that detects a stuck-at fault in the interswitch channels. In this method, a router and its neighbor routers construct a neighborhood. A router in one direction transmits test sets to another router in the opposite

direction in the neighborhood. The method does not only take higher test area overhead but also needs high test time to detect the fault. The area and time will be more while short faults are considered additionally. Kakoee *et al.* [6] have proposed an on-line test approach to address the stuck-at faults in interswitch channels of general NoCs. The approach sequentially selects a router to test its interswitch channels. The method claims that it can be used to detect short faults on the interswitch channels. Although the approach offers a scalable feature irrespective of the network type, it is not cost-efficient for larger NoCs since test time linearly grows with the NoC-size. Further, faults in local channels like the previous scheme [42] are not addressed while faults on these channels are as natural as in the interswitch channels. Later this sequential router selection based test methodology is extended in [7] to detect stuck-at faults in different components of NoC routers.

Next to the permanent faults, channels are also exposed to transient faults, e.g., crosstalk and faults due to aging, such as hot carrier injection (HCI). These faults are temporary and recoverable. The leading error correction code (ECC) techniques, such as automatic repeat request (ARQ), forward error correction (FEC), or a mixture of both schemes i.e., hybrid ARQ/FEC procedure, are followed in practice to tackle these temporary faults [26, 43]. Many approaches include these faults alongside the permanent faults in channels. For example, Amirali *et al.* [28] have presented an end-to-end (E2E) on-line fault detection methodology that accounts for transient faults in NoC interswitch channels. The method is based on the FEC scheme that corrects transient faults in the channels. Further, the observed FEC syndromes are reused to detect the channel's permanent faults. Liu *et al.* [44] have proposed an on-line fault detection technique for the transient faults in NoC interconnects. The method works by sequentially selecting a channel shared by a router pair. Also, it is supposed that the method can detect stuck-at and short faults as the permanent faults in channels.

Prior works may compliment each other with respect to a specific issue but one needs an advanced approach that can meet most of the quality characteristics in general. From the above literature survey, one may classify the limitations of the prior works in terms of the following quality characteristics:

- *Maintaining System Reliability and Yield*– It can be achieved by considering an cost-efficient test scheme.
- *Test Size Reduction*– a test mechanism should exhibit low test area overhead.
- *Test Time Reduction*– The overall test time should be at the lowest possible value.
- *Fault Coverage Metric*– The test mechanism should reach a high fault coverage value that may be up to 100%.
- *Performance Overhead*- Application of a test scheme should incur low performance overhead at the run time.

- *Method Scalability*– The scope of a test scheme should not be confined to a particular topology, network size, and channel width.
- *Fault Efficacy*– Along with stuck-at and short faults, a test mechanism should have the capability to detect other faults, such as open fault.

1.7 Research Overview

The NoC research in this dissertation is oriented in terms of on-line channel testing towards following key issues: *test algorithm, area overhead, test time, test scheduling, network performance analysis, and scalability.* The property of tolerating faults in NoC channels at the cost of its limited functionality or compromising its performance to certain level is termed as *graceful degradation* [45]. It is thus believed that the graceful degradation and the fault tolerance are two sides of the same coin. For reliable communication to be ensured in the NoCs, different fault-tolerant approaches [46–48] must have the prior information about the faulty channels in order to take decision on them whether to exploit or discard for transporting packets. This health status of channels is a prerequisite for graceful degradation of NoCs. The channel's health status, however, can only be supplied after testing of channels using a test algorithm that should conduct efficient detection and diagnosis procedures. The test algorithm does not only enable graceful degradation in a granular way but also improve NoC's robustness [35]. The amount of time needed to complete the on-going execution of a test algorithm depends on many factors, such as the working principle of the test algorithm, size and type of test data exercised for a fault, delivery of test data using unicast/multicast routing, routing path length between source and destination, etc. These factors also determine the hardware area overhead required to implement the test algorithm. The test time also acts as a dominant factor to performance degradation at least in terms of latency, power consumption especially in the on-line mode because application packets forcibly wait at the routers whose channels are currently under test. In the on-line mode, a part of an NoC as subNoC whose channels, for instance, are considered under test while rest part of the NoC can be allowed to continue applications. Therefore, the subNoCs must be selected in such a way that the channels in the corresponding NoC can be tested at the cost of few test iterations. The test scheduling addresses these issues. Faults in channels and other components of NoCs have considerable influences on the performance deterioration. Network performance analysis for a test scheme helps to uncover the impact of faults on various performance parameters or metrics, such as throughput, latency, power, etc. Scalability is another requirement for any test mechanism designed for an NoC system. The test mechanism should scale with different aspects of the NoCs, for instance, network size, channel width, and network type. Different progressively improved mechanisms are proposed for testing of the channel faults in this dissertation.

1.8 Motivation and Problem Formulation

Despite several advantages, NoC channels are more vulnerable to common manufacturing (permanent) faults: short, stuck-at, and open faults which put the NoC into different failure modes. Consequently, testing of channels has become an essential part in the NoC to prevent it from severe performance degradation. A set of test solutions for testing of these channel faults have been discussed in the literature over few years. These solutions may compliment each other on a set of issues. Despite their several advantages, one can observe three basic issues: test area overhead, test time overhead, and performance overhead in these techniques that subsequently incur higher test cost. Considering the first aspect, it is necessary to implement a test mechanism with lower hardware or test area overhead that depends on the test data and test response volume for a fault. The test time overhead incurred by the mechanism defines the overall time taken to address faults in all channels of an NoC. The metric basically depends on the working principle of the test mechanism, the test size, and the test scheduling. Various failure modes due to channel faults are manifested as channel errors that may have severe impact on the system performance. This performance is degraded while the test time seems to be high at the system run time. Thus, the motivation of the current thesis is designing a suitable test solution that can be applied both in the off-line as well as the on-line mode of the NoCs.

Consider an NoC architecture that has n-bit communication channels. Also, suppose that the channels experience a manufacturing and/or a temporary fault. Then, the main objectives of this thesis for the testing of the fault in NoC channels can be enumerated as follows.

(1) Designing of a time-efficient, distributed, test algorithm at a node that has the capability in detecting the fault for checking the health status in terms of faultiness or non-faultiness of channel wires shared by the node.

(2) Extension of the test algorithm for diagnosing the wires to identify the faulty wires in a channel and report possible channel errors caused due to these faulty wires.

(3) Reduction of the test cost, at least in the form of low hardware area, and the overall test time and associated performance overhead through designing a suitable test scheduling scheme.

(4) Improving the coverage metric towards reaching 100%.

(5) Overwhelming the fault efficacy property of the prior works.

(6) Suggesting a test energy model for the network resource utilization in terms of energy dissipation during testing of NoC channels.

(7) Widening the scope of the existing test solutions. The proposed solution, therefore, should not be limited to traditional networks, such as mesh but at the same time, it should be applied on the untraditional networks, such as an octagon, κ-octagon, and spidergon. In other words, the proposed solution should be scalable and adaptable to the NoCs with respect to their size, channel width, and type.

The proposed test solution has two parts. One is the test algorithm and another is the test scheduling. The main purpose of test algorithm is to address a manufacturing fault in NoC channels and extend it to a temporary channel fault. Testing is performed by selecting suitable test stimuli (data) that consist of finite number of test vectors, applying these vectors to the channels under test, and then comparing the received test vectors as the test responses with the expected responses. The proposed test algorithm is implemented using a special machine called built-in-self-test (BIST) structure. Two BIST structures are designed. One is placed at the sender side and another is placed at the receiver side. Former BIST module derives, packetize, and applies the required test stimuli. Later BIST module, on the other hand, does comparison between the set of test responses and the set of expected responses. A difference between the two responses results to an existence of the fault in the channels. An advantage of using BIST structures is that they lower the test area overhead by exercising small test set. Further, two types of routing- unicast and multicast, are included in the sender BIST block. Also, the BIST block in the receivers does comparisons in parallel. The objective is to lower the test time that the algorithm takes. The NoCs have a great advantage that allows parallelism in the operations. By taking this benefit, the test algorithm can be concurrently initiated at multiple nodes towards meeting the objective of lowering the overall test time for the fault. The objective can be fulfilled quickly on considering a suitable test scheduling scheme. Interconnection of multiple nodes taken from the NoC is treated as a subnetwork (subnet), also called a subNoC. The basic job of the proposed test scheduling scheme is to select a subNoC for its channels to be put under test in a test round.

1.9 Major Contributions of the Thesis

The main contributions in the dissertation are briefly mentioned as follows.

A. Addressing Stuck-at Faults in Channels of Networks-on-Chip

This work is dedicated to a low-cost and fast test solution for addressing channel's stuck-at faults in NoCs. The proposed test algorithm detects both stuck-at-0 and stuck-at-1 faults in order to observe the correctness of the channel wires and measure the effect of channel's stuck-at faults in terms of packet corruption, packet misrouting, and packet dropping errors. To execute the test algorithm at nodes concurrently, two test scheduling schemes- diagonal node selection and graph coloring models are proposed that reduce the overall test time and make the proposed test solution scalable with general NoCs.

B. Maximal Connectivity Detection in On-Chip Communication Networks
This work is dedicated to the maximal connectivity test with open faults in NoC channels. The connectivity between source and destination nodes in the NoCs is ensured by detecting open faults and locating a faulty channel wire. Through the testing of channel-open faults, the proposed test algorithm addresses both partial and full packet loss caused by the faults in the network. The scheduling scheme in the work introduces the 4-corner principle which drives the test algorithm on the selection of multiple nodes at a test iteration.

C. Impact of Short-Channel Faults in On-Chip Interconnection Networks
This work proposes a time-independent scheme for the analysis of short faults in NoC channels. The proposed test algorithm targets both intra-channel and inter-channel short faults. In addition to ensuring the correctness of channel wires, the test algorithm measures the packet duplication, packet misrouting, and packet dropping failure modes. The cluster-based test scheduling scheme makes the proposed solution time independent with respect to NoCs in general.

D. Test Time and Energy Optimized Scheme for On-Chip Channel-Faults
This work deals with both transient and manufacturing short faults in NoC channels. Algorithmic construction is based on the strategy that in one hand detects channel-short faults and diagnoses these faults to identify faulty channel wires, and states an observation method in another hand to detect a transient fault in the channels. A partition-based new test-time independent scheduling scheme is proposed. Also the work introduces a theoretical test energy model that can be used to observe the amount of NoC resources accessed during channel testing.

E. Optimal Detection and Diagnosis of Coexistent On-Chip Channel-Faults
This work focuses on the detection of coexistent manufacturing channel faults in NoC-based systems. Here, both short and stuck-at faults are considered simultaneously to occur in channels. In addition to detection and diagnosis of these faults, the issue of fault non-diagnosability is discussed. A variant of the partition-based test scheduling scheme is considered to apply the proposed test solution on both conventional and unconventional networks at the cost of constant test time.

1.10 Organization of the Thesis

The rest of the dissertation is organized as follows.

Chapter 2: This chapter presents detailed state-of-the-art on the testing of transient and manufacturing channel faults in NoCs.

Chapter 3: This chapter is dedicated to addressing stuck-at faults in the NoC channels.

Chapter 4: This chapter targets the testing of open faults in channels in view of maximal connectivity detection between nodes in NoCs.

Chapter 5: This chapter analyzes the short-channel faults in on-chip networks.

Chapter 6: This chapter deals with the detection of both manufacturing and transient channel faults in NoCs.

Chapter 7: This chapter focuses on the testing of coexistent manufacturing faults in the communication channels of NoCs.

Chapter 8: This chapter concludes the thesis and forecasts the immense scope of the proposed works.

Chapter 2

Literature Survey

2.1 Introduction

To meet the modern requirements of high performance computing and communication, multicore processing systems have been considered as the widely accepted architecture. Consequently, many microprocessor manufacturers are currently migrating to chip microprocessors (CMPs), e.g., SoCs where the number of processors with time has seamlessly increased from a few cores (e.g., Intel's quad-core processors [49]) to thousands of cores (e.g., Adapteva's Epiphany [50]). As the complexity in the high performance multicore systems is increasing, NoC has become the dominant architecture [28].

An NoC architecture as a promising solution to multicore SoCs has provided several advantages, such as (1) it is a communication structure that ensures energy efficiency, (2) it is a high performance interconnect fabric that ensures the quality of services (QoS) in terms of guaranteed bandwidth and reliability, (3) it validates functionality and performance at various (electrical to transaction) levels using own communication infrastructures [5, 51]. In spite of many advantages reaped for an NoC architecture, various issues regarding the challenges in design complexity, synthesis of the NoC architecture have arisen and are subsequently discussed in [5, 51–53]. The design process for NoC-based systems is considered from the performance and VLSI design perspectives. In the first perspective, high-performance requirements, such as low latency and high throughput are the desirable characteristics. The characteristics are met with intellectual property (IP) cores that communicate with one another through intelligent routers and high bandwidth channels. Communication designs are generally considered at a high abstraction level. On the other hand, energy dissipation profile of the interconnect architecture that results in silicon area overhead is a notable characteristic in the second perspective. Long wire segments in channels show high energy dissipation. The designers have overcome the issue by optimizing the communication medium. A salient feature of the NoC that decouples the communication fabric from IP cores is used to do so.

According to the report published by the International Technology Roadmap for Semiconductors (ITRS) [54], traditional scaling will no longer satisfy the performance requirements in both homogeneous and heterogeneous network architectures in the long term. The continued progress of interconnect performance consequently requires radically new interconnect paradigms. Multiple approaches have been envisioned so far as emerging interconnect technologies that enable a high degree of integration [52]. Further, when an NoC architecture is operated in a system or manufactured for the system, various transient and manufacturing faults due to aggressive technology scaling are induced on various components of the NoC architecture. This on-chip architecture (NoC), as a consequence, is adversely forced to change its normal operation and thus the system malfunctions. In the report of the ITRS [54], it is announced that 1% chips per day are found defective during their operational lifetime. The manufacturing fault rate may become $\approx 10^3$ defects/m^2 and the trend may adequately be boosted in the near future dedicated to the next generation NoC-based high-performance communication infrastructures. Testing of NoCs has thus become a necessity to keep the chip functional and maintain the quality of service at the best effort delivery.

Rest of the current chapter is organized as follows. The NoC basics that describe key information of the primary NoC elements, and routing and switching techniques are studied in Section 2.2. Possibilities of organizing the basic building units of NoCs to emerging interconnect technologies are mentioned in Section 2.3. Section 2.4 presents basic fault modeling schemes which are generally used in VLSI circuits and may be considered for studying the faults in NoC's components, i.e., IP cores, routers, and communication channels. Preliminaries about the NoC testing is provided in Section 2.5. Detailed literature and the merits and demerits of the art mechanisms for the testing of communication channels in NoCs are discussed in Section 2.6 and Section 2.7, respectively. Motivation and the list of contributions which are the base in rest of the chapters of this thesis are mentioned in Section 2.8.

2.2 NoC Basics

The concept of NoC is first suggested by Benini *et al.* [15, 16]. Authors have used the traditional computer networking mechanism for communicating application data in the form of packets via routing paths across several routers. Figure 2.1 represents a general view of an NoC paradigm. The idea of using NoCs as the interconnection networks or on-chip communication architecture on the multiprocessor SoCs (MPSoCs) is considered as the viable solution that has been gaining traction since last decade. Furthermore, such attempt scales down the concepts of large-scale networks into parts and apply them to the domain that embeds SoCs [17]. An NoC as the on-chip communication network consists of three basic

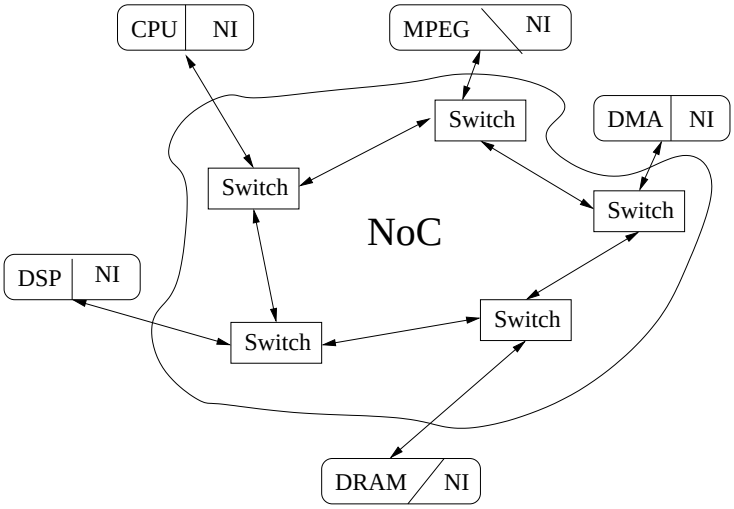


Figure 2.1: High level view of an NoC paradigm.

components namely processing element or IP core, message forwarding router or switch, and communication channel or interconnect. Unlike the SoCs, the NoC is a packet switched network that routes application data packets from an IP core as the source to another IP core as the destination. Application data to packets are organized by employing a switching mechanism and transported by using routing algorithms. The packets are routed via a routing path that consists of an interconnection of routers and channels between source and destination in the NoC architecture.

2.2.1 IP Cores

The primary source of message generation in the on-chip interconnection networks (NoCs) is the intellectual property (IP) core. It is the first building block of the networks and treated as a predesigned and pre-verified logic block or data segment which is used in making a field programmable gate array (FPGA) or application-specific integrated circuit (ASIC) for a product. The IP cores are generally obtained from internal sources, or different third parties, vendors and embedded deeply or hierarchically on a single chip [1, 55]. IP cores or blocks as essential elements in design reuse have become parts of the growing electronic design automation (EDA) industry. An IP core wrapped up with a network interface (NI) is entirely designed to be portable so that it can easily cope with any new design methodology or vendor technology.

The number and types of IP cores can be diverse and primarily fall into two categories: hard cores, and soft cores [56–58]. Hard IP cores are physical manifestations of the IP design and synthesized blocks that can be fabricated, placed in the FPGA. For the layout designs, these IP cores in a layout format are mapped to a process technology following the final layout of a chip. On the basis of diverse functionalities, hard IP cores duly include

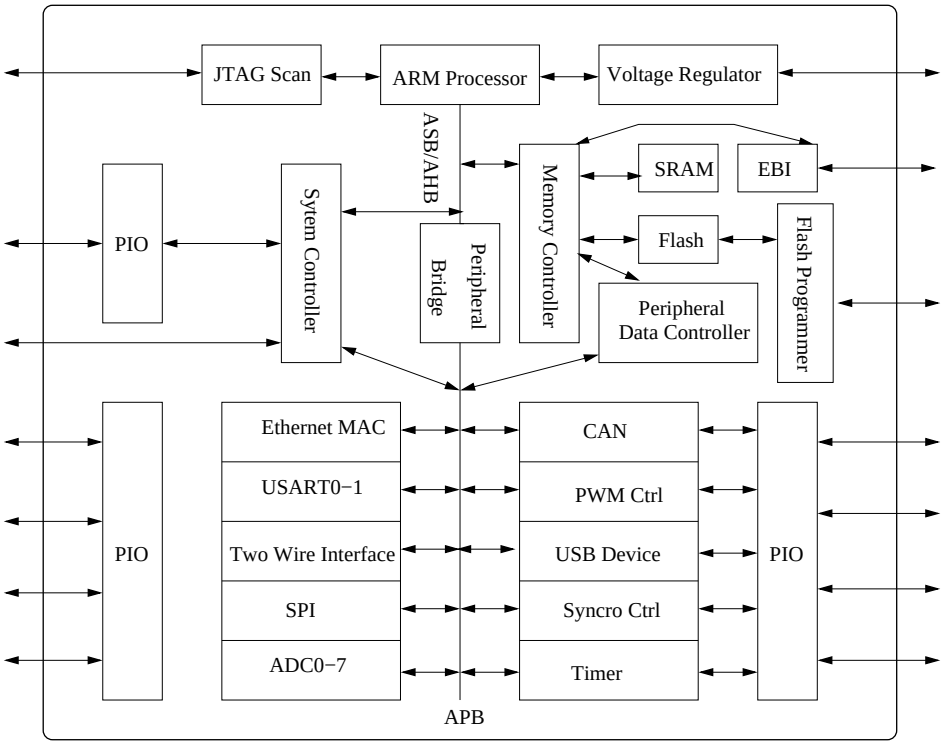


Figure 2.2: Various IP cores in a Microcontroller-based system on a chip.

wide range of devices, e.g., Universal Asynchronous Receiver/Transmitter (UART), memory controller (ROM, RAM, FIFO), central processor units (CPUs, e.g., atlas processor, plasma processor), DSP Multiplier, Flash memory (boot, user), Ethernet controllers, PCI and PCIe interfaces [49, 59–61]. Figure 2.2 represents a microcontroller-based system on a chip that embeds various hardware blocks, such as embedded processors, interface blocks, memory blocks, and other components as IP cores [62]. Note that all of these IP cores can be designed stand alone for different designs that handle application specific processing functions. These are best for plug-and-play applications [55]. By the nature of their low-level representation, hard IP cores offer better predictability of chip performance in terms of silicon area and latency. At the same time, this representation does not allow chip designers to modify the core's application function to be meaningfully modified. These cores are therefore customized for dedicated process technologies on chips and sometimes termed as analog IP cores [63].

Soft IP cores as the name suggests are the most flexible IP blocks that generally exist either as a generic gate-level netlist or hardware description language (HDL) code. A netlist as the soft IP core is a list of logic gates and associated interconnection that make up an integrated circuit. On the other hand, an HDL code block as the soft IP core is offered as a synthesizable RTL model which is developed using one of the Hardware description language like System Verilog or VHDL [56–58]. Many hardware components used as the hard IP cores can be made synthesizable to soft IP cores that permit chip designers to modify designs at

least at the functional level. Each module, in this case, is delivered with a reference design and a testbench in VHDL/Verilog. Thus, the IP core implemented as a soft core is portable to any process technology. For the reason, soft IP cores, e.g., a DRAM controller IP, Ethernet MAC IP, AMBA bus protocol IPs etc. are generally considered as digital logic IP cores. In addition to these IP cores in a reusable form, other soft cores may include real-time operating systems and kernels, library functions, and device drivers [55, 56, 62, 63]. Beside the hard and soft IP cores, designers consider another class of IP cores which are less portable and flexible than the soft type of cores but are more modifiable than the hard type of cores. This new core class is termed as the firm or sometimes called semi-hard IP cores. Firm cores like another type of cores also carry placement data and are configurable to various applications.

The goal of a system on a chip is to achieve large productivity gains in the system. It can be achieved using an IP-based approach along with other components. The SoC/NoC designers in the system integrate them on the chip in order to consummate complex functions in a relatively short amount of time. Further, the integration process involves connecting the IP blocks to the communication network that interconnects the message forwarding routers and high speed communication channels [55].

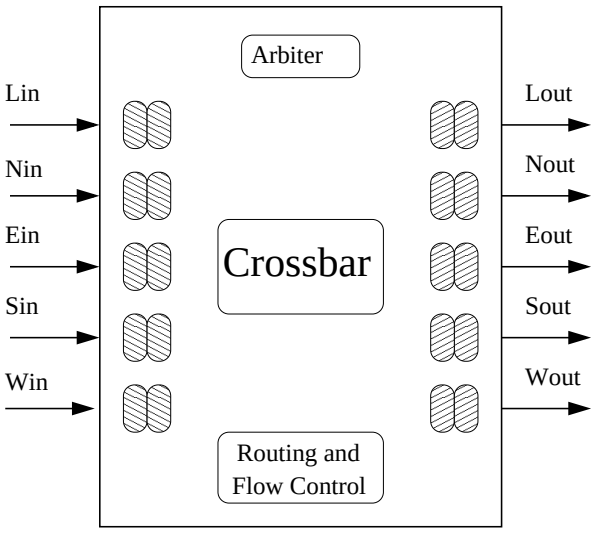


Figure 2.3: General architectural view of an NoC router with its main components.

2.2.2 Routers

A router also known as a switch is another basic building block of the interconnection networks (NoCs). Routers constitute the backbone of an NoC architecture and are responsible for routing packets from sources to destinations. Design of a router therefore critically affects the network performance both in terms of throughput and latency, and design cost in terms of silicon area overhead. A router consists of a number of components, such as FIFO buffers,

switch allocator, arbiter, routing logic block (RLB), multiplexer, and I/O ports. The I/O ports are separated into two classes. The first class of I/O ports is utilized for the communication among the routers. The second class of I/O ports is on the other hand dedicated to the communication from the routers to the terminal IP cores [18,64]. RASoC [65], Xpipe [66] are the examples of two common routers used in constructing an NoC-based interconnection network. Figure 2.3 illustrates a general architecture of the NoC router with its main components [38]. Each router as seen, has four set of I/O ports in the north (N), south (S), east (E), west (W) directions to connect four neighbor routers. These I/O ports are called global ports. Apart from that, the set of I/O ports between a router and its terminal IP Core is called local (L) ports. In the straightforward implementation of a router, it is connected to one or multiple IP cores via direct unidirectional or bidirectional connections. In this case, a multiplexer is used to select an IP core to bring the flexibility in the local reconfiguration of the cores [8]. In addition to design as well as implementation issues, a router defines a set of control policies. These policies are required to deal with routing, packet collision, and overall strategy for transporting data packets on a network. The logic block in NoC routers generally implements these policies.

2.2.2.1 Flow Control

The goal of a flow control policy is to allow network resources for transmission of packets from source to destination. Therefore, a flow control is defined as the policy that characterizes the movement of packets. STALL/GO, ACK/NAK, T-Error [67] are the common flow control schemes used in NoCs. Alternatively, the flow control policy can be considered as a packet contention resolving the problem during the packet traversal [17]. The resolving issues are included at both NoC and router levels. The NoC-level issues are treated as the global issues while the router-level issues are addressed at the local level [64]. For instance, when transmission errors occur, packets in spite of the errors must be recovered. However, the support provided by the underlying flow control mechanism decides packet recovery from the error. For example, the flow control mechanism stops the flow of packets from the sender whence retransmission of a corrupted packet is needed. Subsequently, the mechanism performs request signaling in order to reallocate bandwidth, buffer, and other resources [17]. As a disadvantage, most of the flow control techniques can not reallocate all resources for the packet retransmission. Therefore, one must implement either a scheme that can handle reliable packet transfers or an error correction mode. Oppositely, as an advantage, a flow control policy can manage channel congestion by taking some measures in the policy in order to ensure a *deadlock-free* routing for guaranteeing communication performance and quality of service (QoS) [10, 68]. A measurement may be done by avoiding certain routing paths in the NoC. Flow control can be of two types: *centralized* or *distributed*. In the first case, decisions on the routing with a strategy that guarantees no traffic contention are taken globally i.e.,

throughout the NoC and applied to all nodes. One advantage of this method is that it avoids the need for an arbitration block. But, all nodes have to share a common sense of time. The well known time division multiplexing (TDM) is a possible implementation of the centralized flow control mechanism that associates a time frame to each and every packet [24, 69, 70]. NoCs in the second case, however, generally use a distributed control that allows each router to take decision locally.

Definition 2.1. *A flit is a basic and smallest unit over which flow control is performed during routing a packet.*

Definition 2.2. *A phit is a data unit transfered on a channel in single clock.*

2.2.2.2 Switching Strategies

A switching strategy in the NoC architectures determines how the message flows from a source to destination by setting the routers. The switching techniques define granularity [8]. The IP cores are the basic message generation centers in the NoC and are supported with buffers. A message is considered to be a single packet or broken into multiple packets. Each packet is further broken into several *flits*. These flits are classified into three sets- header, payload, and trailer. Some designers consider each flit as a collection of *phits*. A general packet organization is shown in Figure 2.4. Different NoC architectures use different switching techniques that correspondingly handle varying sizes of phits, flits, and packets. Two classes of switching strategies are practiced. One class is *circuit switching* and another class includes *packet switching* [8, 17, 24].

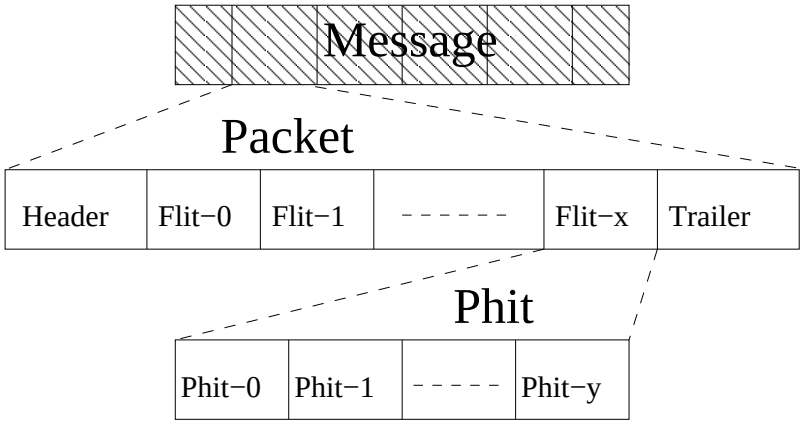


Figure 2.4: An NoC message organization.

In the circuit switching mode, a physical path as the routing path between the source and destination nodes is established by the header of a packet prior to transmission of data from the packet. The physical path is a series of alternate interconnection of routers and channels. If the packet header from the source reaches without any conflicts to the destination, the whole

routing path is reserved for the transportation of data in terms of payload flits and that path is available till the packet trailer has torn down it. The SoCBus NoC architecture [71], for example, implements this circuit switching technique. The main advantage of this switching technique is that full channel bandwidth is available through the path setup. In spite of a dedicated routing path, this technique does not scale while the NoC size grows because many channels are occupied during packet transportation. As a result, it produces excessive blocking that leads to high packet latency. But the technique guarantees throughput due to the reserved paths [18, 72].

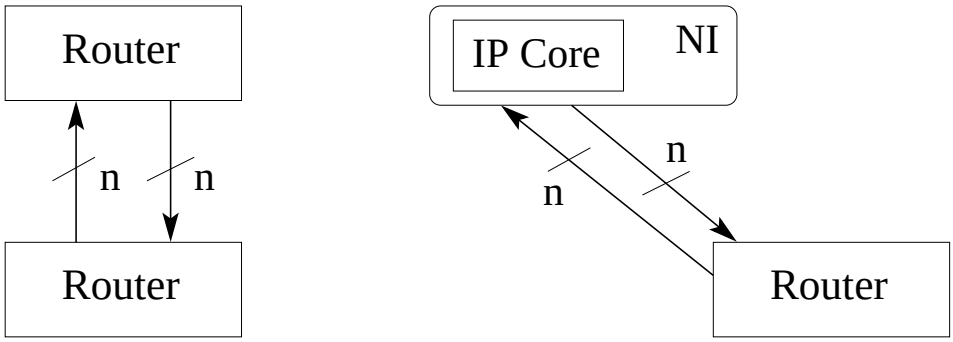
Instead of establishing a reserved path before transmitting any data packet, all packet flits are transmitted once the packet header establishes the connection between routers. In this packet switching mode, the packets from the source possibly follow different routes independent of the receiver but with different delays. Unlike the circuit switching, multiple packets from different sources reach to a router. One or many of these packets must wait until an intended output channel of the router is available. Thus, the case of packet contention is possible and subsequently, QoS guarantees become harder to make. Despite this fact, designers choose this switching technique since no channel reservation in advance is needed that saves startup time as needed in the former switching method. Further, QoS guarantees may be achieved to some level by applying any of the following packet switching techniques- *store and forward (SAF)*, *virtual cut through (VCT)*, and *wormhole (WH)* switching. The SAF switching is a simple packet switching technique. In the SAF strategy, a router stores the whole packet before forwarding it to the next router in the routing path. Therefore, one must ensure in the SAF strategy that buffer size at the routers must at least be equal to the packet size or the received packet can be stalled. The NoC architecture discussed in [73] implements the SAF strategy during packet transmission. The VCT-based packet switching aims at reducing the router latency over the SAF scheme. In the VCT switching, the first packet flit from a router is forwarded to the next router in the routing path as soon as a space for an entire packet is available at this receiver router. Thus, the scheme eliminates the waiting time for the entire packet to be received. Furthermore, no channel except the current router in the routing path is affected in case of stalling. Other packet flits follow the first flit without any delay. The VCT scheme is implemented in [74]. The WH switching is widely used for sending packets in an NoC architecture. A router makes the routing decision as well as forwards all flits of a packet. Thus, buffer requirements in the WH switching are reduced to the size of a single flit. However, this switching is more susceptible to packet deadlocks than other switchings due to blocking of channels. Because, if the receiver router is unable to store a whole packet in case of insufficient space, parts of the packet are distributed to many routers. Such a packet distribution may result in channel blocking [8, 17]. Most of the architectures, such as SPIN [75] use this WH packet switching. Also, other architectures, such as Æthereal [70], and MANGO [76] use a combination of VCT and WH switchings.

2.2.2.3 Routing Algorithms

The routing algorithm is the logic that takes the responsibility of correctly and efficiently transporting data packets from source to destination. When a router has received a packet at one of its input ports, the routing algorithm forwards the packet to the selected output port. This port selection depends on the routing information available in the packet. There are many routing algorithms available for the NoCs. The choice of a routing algorithm however, depends on the tradeoff between performance and cost that includes minimum routing power, high performance with low delay and maximum traffic utilization, minimum logic and routing table, improved robustness [17]. Routing schemes are broadly classified into *deterministic* and *adaptive* routings that are further categorized into static/dynamic, source/distributed, and minimal/non-minimal routing [17, 18, 24]. In the deterministic routing algorithm, a packet is always routed on the same routing path between source and destination nodes. The source routing and XY routing [77] are the common deterministic routing techniques. The source IP core in source routing determines the route to the destination while a packet first moves in the X direction and then in the Y direction before reaching towards the destination. On the other hand, an alternative path between nodes is used when local channel or original routing path becomes congested. In this case, channel load (traffic) is evaluated dynamically with a dynamic load balancing strategy. Negative first (NF) and west first (WF) [78], minimal adaptive and fully adaptive [64], turnaround turnback [79], odd-even [80], congestion look ahead [81], etc. are example of common adaptive routing strategies.

2.2.3 Communication Channels

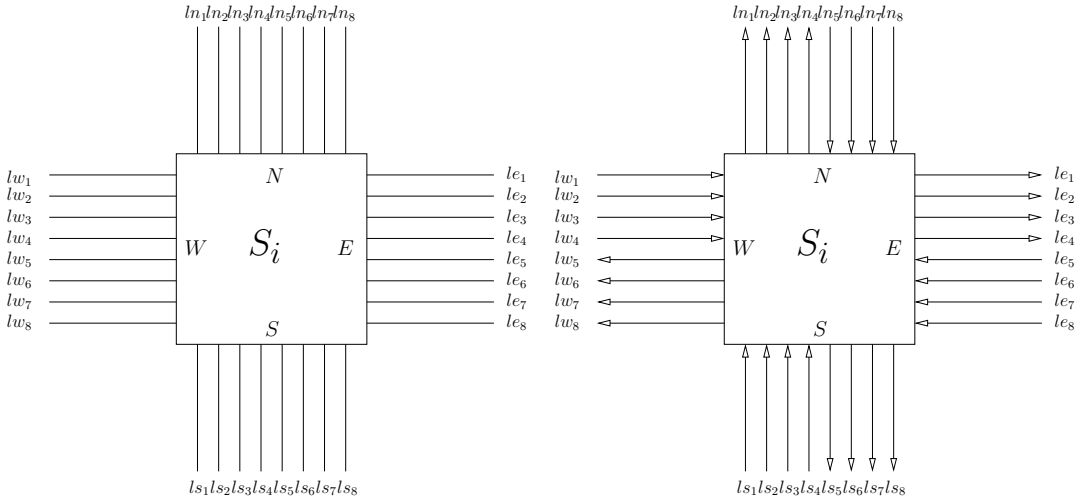
The third and final basic building block of the NoCs is the high speed communication channels. A set of metallic wires is shaped to a communication channel that acts as a basic means of packet transmission medium in the NoCs. Each wire is synonymously well known as a net, line by many designers. Also, the communication channel is meant to a link, an interconnect. A communication channel that connects other two basic building blocks (router and IP core) via their I/O ports is classified into two categories: interswitch channel, and local channel. The interconnection channel shared by two adjacent routers is called as the interswitch channel. Similarly, the interconnection channel that is shared by a pair of router and its dedicated IP core is known as the local channel. Figure 2.5 illustrates a channel in an NoC. Typically, a channel whether interswitch or local is unidirectional or bidirectional (Figure 2.6). In case of a bidirectional channel, terminal entities (router, IP core) have an opportunity to send and receive data packets at two different time instants on the channel in either direction. So they establish a *half-duplex* mode of communication. That means, the terminal entities use two *simplex* mode at non overlapping time instants. For example, the router S_i (Figure 2.6a) can only transmit/receive a packet on its channels at a time. On the unidirectional channels, the



(a) An interswitch channel with *n-bit* width.

(b) A local channel with *n-bit* width.

Figure 2.5: Abstract representation of a communication channel in an NoC.



(a) Bidirectional interswitch channel wires.

(b) Unidirectional interswitch channel wires.

Figure 2.6: Bidirectional and unidirectional channel wires from the output ports of a router S_i.

terminal entities have an opportunity to send and receive data packets at the same time on the direction of the channels. In this case the entities use a *full-duplex* mode of communication which is implemented by two separate simplex modes on the channels. A simplex mode of communication is established when the terminal entities on a channel send and receive packets unidirectionally [82]. For example, the router S_i (Figure 2.6b) can transmit/receive packets simultaneously on its output/input channels, respectively.

Definition 2.3. *The number of n wires which are shaped in a channel defines the bitwidth or width n of the channel.*

As mentioned, each channel consists of multiple uniform wires throughout the network and is characterized by its *width*. The packet flits defined earlier are considered at this channel level. Each flit in most cases corresponds to a phit and is taken as the minimum packet data to be transmitted on a channel. The flit in this case matches with the n of the channel.

But the case is invalidated on the highly serialized channels where the flit is composed of multiple phits. Implementation of channels is done synchronously or asynchronously between source and target nodes. In the former case, a synchronization protocol is implemented on the dedicated channel wires or other approaches, such as FIFOs are brought into play [83]. In the latter case, a globally asynchronous local synchronous (GALS) scheme is the option where a local handshake protocol is used [10]. Anyhow, the channels ultimately define the network performance (throughput, latency) and power consumption. Designers are therefore supposed to support high bandwidth, reliable, and low power communication channels in NoCs.

2.3 NoC Topology

An NoC architecture as the on-chip communication network is partitioned into three basic building blocks which are the processing elements commonly known as IP cores, data forwarding routers, and data transmission medium known as on-chip communication channels or links. These components are interconnected in various ways resulting in a topology. The topology plays a major role in the performance evaluation and design cost determination for an NoC. Therefore, selecting an NoC topology is an important task of NoC design that could optimize these factors in addition to dealing with many physical characteristics, such as the length of channel wires, degree of a node, and routing schemes. Designers prefer an NoC topology as an interconnection network that has smaller diameter, smaller node degree, lower average distance between source and destination nodes, more number of communication channels, and so on [64].

Definition 2.4. *An NoC topology defines a physical organization of interconnected basic components that would construct a network infrastructure, e.g., mesh, torus, octagon.*

Definition 2.5. *The term diameter in an NoC topology is the number of hops that returns the maximum shorted distance between two nodes.*

Definition 2.6. *The average of distances of possible node pairs in an NoC topology defines an average distance on the NoC.*

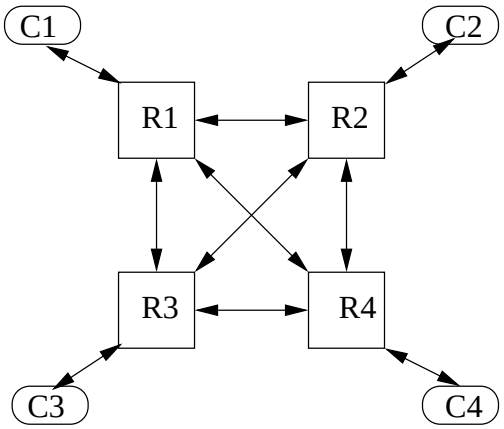
Definition 2.7. *The degree of a node in an NoC topology is defined as the number of neighbor nodes connected to the former node.*

The performance characteristics of an NoC are often influenced by these parameters. For example, a larger diameter stands for a packet to travel more hops while it intends to reach the destination at the farthest. Similarly, a large average distance incurs higher overall latency while smaller node degree makes designers easy to build a network, i.e., NoC topology [8]. Designers and researchers have come up with a variety of NoC archetype that is broadly classified into two networks: regular and irregular NoCs. The regular category is further

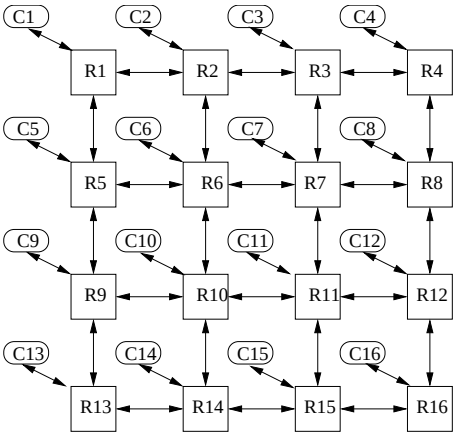
subdivided into direct and indirect classes. Note that each type of NoC interconnection has its own pros and cons.

2.3.1 Regular Topology

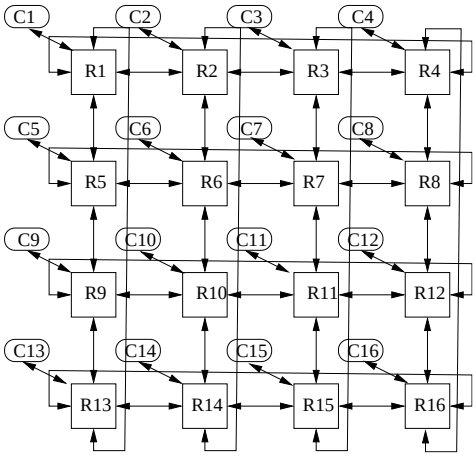
In a *regular* NoC interconnection network, the basic communicating elements (IP core, router, and channel) have a regular architectural configurations. For instance, all routers in a regular topology are identical in terms of the number of I/O ports to neighbor routers and the local IP cores. On every regular topology, there is a predefined pattern that defines the way how communication channels from a router are interconnected with another router and/or IP cores. Regular NoC topologies are usually the typical choice for high-performance chip multiprocessors (CMPs). Many reasons, such as better scalability with network size, topology re-usability whenever needed, and reduced design time include the fact. Regular NoCs are sub-classified into direct and indirect networks.



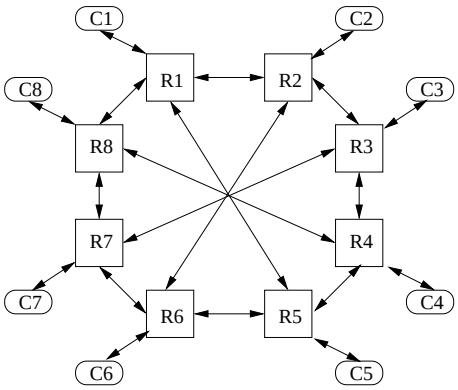
(a) A P2P Topology



(b) A 4×4 Mesh Topology



(c) A 4×4 Torus Topology



(d) An Octagon Topology

Figure 2.7: High level view of NoCs with direct topologies.

2.3.1.1 Direct Topology

In the direct topologies, each node has direct connections via channels to a fixed number of neighbor nodes. The direct topology provides the advantage that the total communication bandwidth increases in the network while the number of nodes increases. However, the connectivity for the large direct network that results in high performance, takes higher energy and area costs. Correspondingly, a tradeoff between the connectivity and cost on the design of a direct network is established. Thus, a fully connected direct network where every node is connected point-to-point (P2P)(Figure 2.7a) to all other nodes in the network, is completely prohibitive. Therefore, most direct networks practiced in general include mesh (Figure 2.7b), ring [84], torus (Figure 2.7c) and folded torus [85], octagon (Figure 2.7d) [86,87], spidergon [88, 89], etc. topologies.

Most direct topologies are implemented on the orthogonal arrangement of the NoC basic components, i.e., nodes can be interconnected on a δ-dimensional orthogonal space in the manner that every channel whether local or interswitch shows a displacement in a single direction from a router to another router or IP core or vice versa. Conventionally, $\delta = 2$ is used for an NoC topology. A message in one/multiple packets exchanged between nodes distributed in the δ-dimensional space is done using a routing scheme that moves the packets in one direction at a time. A routing in a direct network is fairly simple to implement.

As seen in the literature, a mesh with $\delta = 2$, i.e., a 2D mesh network is the most commonly used NoC architecture among the direct topologies. The network is also the common preference in commercial and industrial NoC products, e.g., Tilera multicore tiled processor family [90], Intel 80-core Polaris chip [49]. Figure 2.7b shows a 4×4 mesh NoC. A 2D mesh-based NoC architecture is called chip-level integration of communicating heterogeneous elements (CLICHE) [73]. Every router is connected to the single processor or IP core via the local channel. Furthermore, every internal router is connected to four neighbor routers in four directions- north, east, south, and west via interswitch channels while border and corner routers are similarly connected to three and two neighbor routers, respectively. The area of a mesh network linearly grows during its design with the number of nodes increases. Additionally, a 2D mesh shows scalability limitations after the certain size. Therefore, the current trend of using a mesh-based interconnection network is expected to change in the near future. Alternatively, the topologies may be the concentrated mesh (C-Mesh) [91] designed on the modifications or optimizations of 2D meshes, the WK-recursive network [92] designed on the basis of radically different connectivity patterns.

2.3.1.2 Indirect Topology

The indirect topologies known as multistage networks are the regular NoCs where identical routers are organized in stages. It means, routers in an indirect topology may or may not

be connected to IP cores. The routers connected to IP cores are called *external* routers while others that do not have any IP core are called *internal* routers. The IP cores access the network indirectly via the local channels shared with their base routers. Each external router may be connected with one or more IP cores and the corresponding nodes act as the source/destination of packets and information processing centers. On the other hand, internal routers can only use to receive and forward the packets through the network. A simplest indirect topology is crossbar where two neighbor IP cores are separated by a single router. One can realize other multistage indirect topologies on just cascading together these small crossbars [17]. Several well known indirect topologies are fat tree [93], butterfly, butterfly fat tree (BFT) [11], extended BFT interconnection (EBFTI) [94], flattened BFT (FBFT) [95], and mesh-of-tree (MoT) [96] networks.

2.3.2 Irregular Topology

The regular NoCs are most suited for general purpose designs. However, there may be a specific application that needs a customized NoC for the communication load or set of applications identified at the design time [97]. In these cases, a mixture of direct and indirect networks with a shared bus is designed. The newly designed topology results in an irregular or ad hoc topology. Shared buses provide low bandwidth while a distance between the source and target nodes in direct/indirect topologies increases with their size. One goal of this new topology is then to increase bandwidth availability as compared to a shared bus. Another goal is the reduction of the distance between nodes as compared to regular networks. Routers in the irregular topologies may show different connection patterns depending on the application specification. Additionally, unnecessary routers and channels are removed during designing an irregular network, for example, reduced mesh. Other noted irregular topologies include Clos, Benes, cluster-based hybrid network [17, 98].

Although, regular NoCs are widely and always the first commonplace of selection than an irregular NoC to meet the performance requirements and minimum design cost in most modern CMPs and MPSoCs, there are other factors that may break the regularity assumption at runtime. Two main dominant factors are power management events, and faults that include both permanent and transient faults. Therefore, an NoC must be designed in such a way that the architecture is capable of preserving certain performance and the correct operation in presence of the above runtime events [18].

2.4 Fault Modeling

This section first searches the sources of different faults that arise in VLSI components and corresponding fault models. Next, the types of testing followed by basic conventional test techniques are discussed.

Definition 2.8. *A fault is a representation of a defect that reflects a physical condition and causes a circuit to fail to perform its normal operation in the desired manner. Alternatively, a fault is a logic level abstraction of a physical defect that is used to describe the change in the logic function of a device caused by the defect.*

2.4.1 Sources of Faults

A fault, in general, is described by its value, nature, extent, and duration. These parameters classify the faults into two broad sets: logical or nonlogical faults. Traditional *manufacturing/permanent faults*, such as stuck-at, short, and open belong to the logical category. Non-logical faults include rest of the faults and commonly known as *transient faults*, such as crosstalk, timing (delay) faults, and synchronization failures [8, 99, 100].

2.4.1.1 Permanent Faults

Permanent faults, as the term suggests, are the logical faults that remain in the system permanently until repaired. Since the permanent faults are unrecoverable, they are also known as *hard* faults [101]. These faults originate from an incorrect manufacturing mechanism, a wear effect, or permanent rupture in a circuit. Consequently, they make the physical changes in the circuit/system behavior. However, this behavior remains unchanged with time. Increasing wiring density with shrinking feature size of a die causes to many permanent faults in input/output interconnects and logic blocks. On the other hand, electromagnetic interference (EMI) often appears in long-interconnects of customized, irregular NoC topologies. Electromigration is another cause of permanents faults. For instance, electromigration highly affects the aluminum interconnect-based ICs. Whereas, the same ICs with copper interconnects rarely fail due to this effect [8]. Various functional and structural test approaches are used to address these permanent faults.

2.4.1.2 Transient Faults

Transient faults also known as *soft* faults oppositely are the nonrecurring faults and occur during the system run time. So, these faults that can be observed for a finite moment, are commonly termed as temporary faults [26]. Transient faults primarily affect the interconnects for various reasons, such as an external perturbation that includes radiation and fluctuation in power supply, scaling of supply voltages, and faster clock rates. These are treated as the external disturbances from the fault site and generally appear intermittently. Many fault-tolerant schemes [46, 48, 102] are employed to handle transient faults.

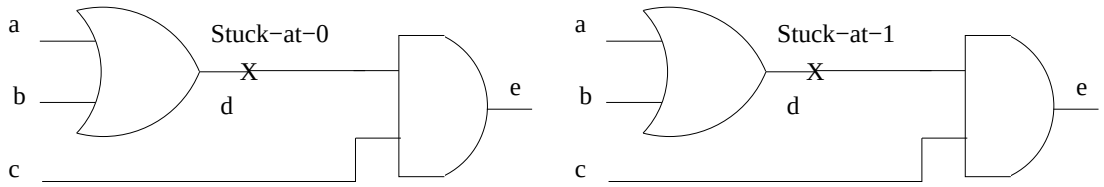
Designers and researchers assume another class of faults called as *aging* faults which arise due to aging effects. An aging effect causes typical switching frequency degradation, transistor parameter degradation, sufficient energy gain by carriers, etc. The faults, such as hot carrier

injection (HCI), negative bias temperature instability (NBTI) due to an aging effect in the circuits, devices fail to maintain performance level [8]. In present days' digital design, these faults consequently become a significant reliability concern.

Definition 2.9. *A fault model is an abstraction of real defects in a circuit or system such that (a) faults in the model are easy to represent, and (b) it should ensure on verification that no fault exists in the circuit, quality of test solution is maintained.*

2.4.2 Basic Fault Models

The faults in system designs produce defective parts, breaks in signal wires, short-circuit between signal wires, wires unintentionally connected to the power supply or ground. Subsequently, behaviors of these faults are characterized into fault models [100]. Generally, a good fault model should satisfy two criteria: it should (1) accurately reflect behaviors of a fault, and (2) be computationally efficient in terms of fault simulation and test pattern generation [103]. At the physical level, an enormous number of different faults could be present and it is quite complex to analyze them at that level. Analysis can be made simple if similar faults are categorized. Thus, physical level faults in a circuit are grouped together with regard to their logical faulty effect on the functionality of the circuit and presented with efficient fault models [99, 104]. Here, we go through a wide range of fault models that are designed for stuck-at, open, short, crosstalk, and delay faults.



(a) An SSA circuit where wire d is tied to logic-0 ($d/0$). (b) An SSA circuit where wire d is tied to logic-1 ($d/1$).

Figure 2.8: An example of the SSA fault model.

2.4.2.1 Stuck-at Fault Model

A stuck-at fault is a hard fault and most fundamental to digital testing. A stuck-at fault affects different components of a circuit. The components may be a state of signals (represented by logic 0/1) on lines (nets) including primary inputs and outputs (I/Os), internal gate, flip-flop I/Os, fanout branches, fanout stems (sources). A correct value on a faulty signal line is modified by a stuck-at fault and results in a constant (stuck) logic value, either a logic-0 or a logic-1. Former fault is commonly known as the *stuck-at-0* (SA0) fault while *stuck-at-1* (SA1) fault is labeled to the later fault. If it is assumed that only one net in a circuit can have a fault at a time, it is called *single stuck-at* (SSA) fault model. Without this assumption, it is

called *multiple stuck-at* (MSA) fault model, otherwise [105]. The SSA fault model is one of the popular fault models and eases the test set generation for a system. Many efficient SSA fault models are discussed in [106–108]. Figure 2.8 for example, represents a simple stuck-at fault model. Therefore, one must apply logic-1 and logic-0 in the primary input wires of the circuits shown in Figures 2.8a and 2.8b, respectively to see whether the wire d suffers from an SA0 and SA1 fault. A single stuck-at fault model is characterized by three assumptions. First, only one wire is faulty at a time. Second, the faulty wire is permanently set to either 0 or 1. Finally, the branches of a fanout wire are independent with respect to locations and effect of a stuck-at fault [99]. Beside the wires, various parts of logic blocks, such as FIFOs, arbiters, RLBs, MUXes, and intra-wires can be affected by the stuck-at faults.

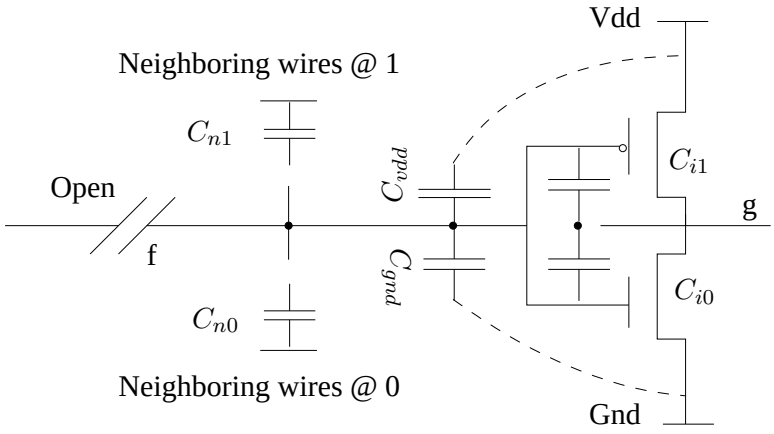
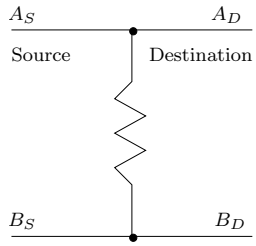


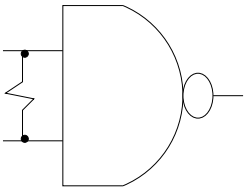
Figure 2.9: Transistor level fault model for open defects.

2.4.2.2 Open Fault Model

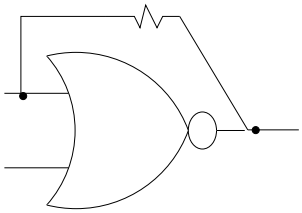
An open fault is another hard fault that breaks a signal wire in a circuit into two or more segments. Therefore, wire segments of the terminal components associated vertically or horizontally make them always disconnected. The component terminals, in this case, may not be in contact with the rest of the circuit. Even at the location of the fault, they create a high resistance in the circuit. Breaks may occur on the wires between gates, input wires to gates, or on internal wires in a gate. It is believed that open faults at first are modeled at the transistor level fault by Wadsack [109] in order to know the structure of the circuit. Later these basic transistor level fault model is reconfigured as per the requirement by many works [110–115] to detect an open defect in the system. Figure 2.9 represents a fault model for open defects. The wire with a cut/ break at the location f is regarded as the open fault floating side wire segment. At the logic level when the faulty wire interconnects the gates, open faults then tend to behave like stuck-at faults. Subsequently, open faults in circuits may be detected by the test sets used for stuck-at faults in these circuits. For the reason, an open is often referred as the stuck-open fault.



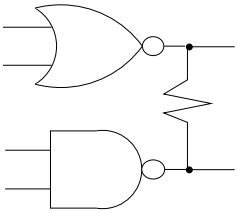
(a) Basic short fault representation.



(b) Input to Input (I/I) Short.



(c) Input to Output (I/O) Short.



(d) Output to Output (O/O) Short.

Figure 2.10: An example of the basic short fault model and its variations.

2.4.2.3 Short Fault Model

A short fault is the next hard fault that makes an unintended connection between two or more elements. These elements may be a group of signal wires, transistor terminals or connections between transistors and gates. When a short fault is experienced between two wires, then the fault is commonly known as bridge fault. Figure 2.10a represents a short fault between two wires. A short fault at the gate level may be classified into three classes: input to input (I/I), input to output (I/O), and output to output (O/O) short, based on the input and output wires of a gate. These three classes of shorts are shown in Figure 2.10b, Figure 2.10c, and Figure 2.10d, respectively. The logic value of a shorted wire is modeled as *1-dominant* known as *OR-bridging* or *0-dominant* known as *AND-bridging* [116]. In other words, the shorted wires in those dominant states are supposed to form *OR* and *AND* logic operations, respectively. Some well known short fault models have been discussed in [116–119].

2.4.2.4 Crosstalk Fault Model

Use of deep submicron (DSM) technology and high range clock frequencies, increasing cross-coupling capacitance and mutual inductance disproportionately cause signal integrity problems in the interconnects. Such problems leading to severe crosstalk have an adverse effect on the proper functioning of the system and consequently rivet significant impact on its performance. Several factors, such as clock speed, interconnect length, impedance matching, etc. contribute to the degree of crosstalk and are included during the design phase in order to avoid the faults. Besides, various technology-related factors, such as increased wire thickness, shrinking of space between wires, increased number of metal layers, the density of amalgamation, etc. may cause the increase in crosstalk errors. The first crosstalk fault model

is proposed by the Cuveillo [120] and named as the maximum aggressor fault (MAF) model. The model is discussed in detail later in this chapter. Generally, a crosstalk error is classified into six types: positive/ negative glitches, rising/falling delays, and rising/falling speed-ups.

2.4.2.5 Delay Fault Model

A delay fault is related to the timing specifications of a circuit and is considered as a soft error that causes an extra delay in the circuit. For example, finite rise/fall of signals in gates and/or transmission delay on wires between gates may result in a delay fault. Minor timing problems in a circuit may not affect its functionality at all. However, longer delays cause major timing problems in gates of the circuit resulting in failing to match its timing specification. In this case, one should prudently assess a circuit's timing if the functionality of the circuit from a delay fault needs to be prevented [121]. A delay fault is modeled in terms of *gate delay* and *path delay* faults [1, 105]. By the name, former type suggests the fault within a certain gate while the later type considers the fault on a path between gates in a circuit. A circuit or system may consist of a huge number of gates that in turn results in a large set of paths. As a remedy, one way is to consider the reduced number of paths and the selected paths are called critical paths.

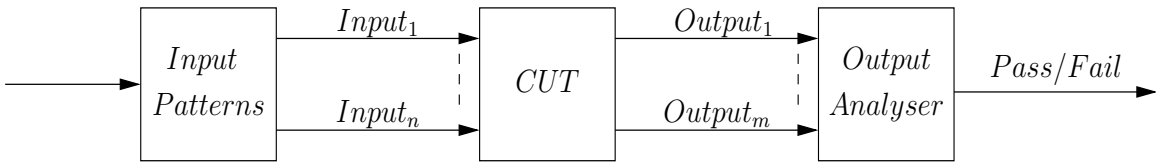


Figure 2.11: Basic Testing Approach of a VLSI Circuit.

2.4.3 Types of Testing

Continuous reduction of the feature size of chips on a die has led to both permanent and transient faults in the systems. At the same time, the necessity of testing of these faults on the systems become essential to verify their correctness as well as to ensure other issues like performance, quality of service, etc. Testing is typically defined as the application of a set of test stimuli (patterns) to the inputs of a circuit under test (CUT) and analyzes the output responses [105]. Figure 2.11 illustrates the basic test mechanism for a circuit. If the CUT produces correct output responses for all input patterns, it is considered to be fault-free. Otherwise, a fault is considered to be present at any component (gate, wire) in the CUT. Two classes of test solutions are normally followed: functional testing and structural testing.

2.4.3.1 Functional Testing

The basic idea of *functional testing* or *specification-oriented testing* on a circuit considers the test of every entry in the truth table of combinational logic blocks of the circuit and determines whether every block produces the correct response. As a comprehensive test solution, the functional testing addresses the issue that the CUT behaves as it should be. For the reason, this testing needs to generate and apply all possible test patterns on a circuit. Consequently, at-speed functional testing on the CUT using an automatic test equipment (ATE) becomes far more expensive than its (circuit) overall design [5]. Despite its large test, extremely high testing time, and low fault coverage, many designers and test engineers in practice, consider functional testing on the circuits to be applied as thoroughly as possible in a system-like mode of operation. Functional testing is carried out in two varied ways using: functional test vectors (FTVs) and hardware redundancies (HRs) [122]. In the first method of conducting the functional testing, functional patterns are applied to the CUT and the output responses are compared with golden (expected) outputs. As a matter of fact, the operation requires an ATE to generate and supply the functional patterns. In addition, the quality of test in terms of fault coverage can be predicted by this technique while a set of patterns is exercised on the CUT. For the reason, the size of the functional test set is less for a CUT in non-critical systems as compared to the CUT in critical systems. The second method of conducting the functional testing is to employ an additional hardware component and compare the outputs of a CUT and this redundant module during testing. An advantage of this approach is that there is no ATE related cost. But, a specific cost incurred by this redundant module is added.

2.4.3.2 Structural Testing

More practical and economical approach to test a circuit is to select a good and recognized solution that uses structural knowledge of the circuit and is, however, able to reduce test complexity to a great extent. This method is well known as the *structural testing* or *defect-oriented testing* [105]. Structural testing is introduced by Eldred [123] and verifies correctnesses of the specific structure of a circuit in terms of gates and interconnects. In other words, structural testing depends on the specific structure of the circuit involving gate types, interconnects, fault models etc. rather than checking a functionality of an entire circuit. The structural testing takes place in many folds unlike the functional testing. Yet it improves the test efficiency in terms of the test time and the quality of test. Because, this test technique is conducted for a specific fault that requires a finite set of test patterns. Therefore, a structural testing on a circuit cannot guarantee detection of all permanent faults at a time. A structural test can be classified into two groups: static and at-speed structural testing [124]. The static structural testing addresses only the static faults like permanent faults while the transient faults are addressed by an at-speed structural testing.

2.5 Testing of NoC Faults

The faults whether permanent, transient, or aging may appear at any time in the system. However, a majority of the faults as observed are introduced during the manufacturing process of the system. Hence, focus on the testing of faults related to manufacturing, production is of high importance than transient and aging faults [100]. Testing of faults are basically done in two phases: fault detection and fault diagnosis. Detection of a permanent fault in a chip component is carried out by generating multiple test sets followed by applying each test set to a target fault model. In other words, one can define the fault detection as the phenomena in a circuit when an output value is different from the good/ expected value. After the detection of a fault in a component, one must extend this current method to identify the specific location of the fault in the component. This location finding is done by the diagnosis procedure. For example, NoC channels must be diagnosed to search for a channel wire affected by a fault. In addition to locating the place of the specific fault, the diagnosis mechanism categorizes the identified fault as permanent/transient. The diagnosis is therefore absolutely essential for proper countermeasures, such as a variety of redundancy techniques, replacement techniques to be taken for reliability and yield improvement issues [121].

In spite of the fact as mentioned above, post-manufacturing assessment of the components in SoCs, NoCs, or any other systems for permanent faults only, does not suffice for full detection and diagnosis of all sources of faults. This is because of the fact that many transient faults may not be noticed directly after manufacturing of the product. For instance, a possible defect free NoC channel may experience transient faults in the course of lifespan of the chip [34, 125]. Even after conducting several manufacturing tests, these transient faults act as some latent faults and emerge at a later moment under certain physical or thermal conditions on the execution of several operating cycles. Therefore, detection and diagnosis of transient faults like permanent faults in on-chip architectures are similarly indispensable. Both spatial and temporal redundancy is used as the tools that greatly help for analyzing all fault models [47]. In general, spatial redundancy is appropriately used for dealing with the permanent faults while tackling of transient and other intermittent faults, and information redundancy is accomplished by the temporal redundancy.

2.5.1 Types of Test Modes

As seen in the literature, two classes of test methods are practiced for the test of the basic architectural components of NoCs, i.e., IP cores, routers, and high speed communication channels. The first method is the functional testing which applies the test on the NoCs in the normal operation mode. Sometimes additional test structures complying with NoCs' functional modes are employed for the test. The second method is the structural testing

which is practiced for specific faults in the NoC components. Both these test approaches are executed either in the *off-line* or in the *on-line* mode to the NoC structures.

2.5.1.1 Off-Line Testing

The off-line testing assesses the faults present in NoC components while it is not in use, e.g., the NoC is still on the wafer, the NoC has been packaged but not currently in a system. In the lifespan of the NoC, in design (pre-manufacturing) and on design (post-manufacturing) phases, the NoC architecture undergoes the off-line testing. Therefore, whole NoC is put in the test mode in which no application packets can be communicated. Typical ATE-based test solutions, such as scan chain, partial scan chain, boundary scan [1, 24] are used to detect faults in NoC components. In the post-manufacturing of the NoC, if the NoC is in use in a system, then the ATE-based techniques as in-filed test methods simply stop the on-going application and then bring the whole NoC in the test mode for detection of possible failures [44]. However, this is impermissible for the mission-critical NoC-based systems, such as navigating system of an aircraft, nuclear reactor safety system where current applications must not be interrupted [126]. Additionally, it is difficult to implement the at-speed ATE-based testing since the operational speed of the NoCs are very high and adds high test cost for the setup.

2.5.1.2 On-Line Testing

The detection and diagnosis operations of a test technique are traditionally used in systems to protect their communication elements from the effects of possible failures lodged by permanent as well as transient faults. These faults appear at any moment in the system including the NoC. As the off-line testing is prohibited, it is desirable that the mission-critical on-chip systems should possess some form of self-checking mechanism as the alternate solution. The self-checking property makes possible on-line testing of the faults to the systems without the need of external test sets for the faults. The well known built-in-self-test (BIST) mechanisms [1, 24] are realized in the on-line testing of NoC components. In contrast to the off-line testing, BIST-based on-line testing reduces the need of an external ATE, thus enables the at-speed test execution with the NoCs, and permits the test execution whenever the NoCs are powered ON. A typical BIST test architecture or logic BIST controller consists of a pair of *test pattern generator* (TPG) and *test or output response analyzer* (TRA or ORA). Figure 2.12 shows a BIST architecture for an NoC under test (NoCUT). Although, the online testing eliminates the demand of the ATE, but needs TPG and TRA blocks that add hardware cost of implementing such mechanism. Therefore, designers must carefully counterbalance the area overhead of these blocks over an ATE during the on-line test implementation on the on-chip data communication infrastructures. Next to the hardware area overhead, designers and test researchers should consider the energy dissipation during the test of NoCs when they are

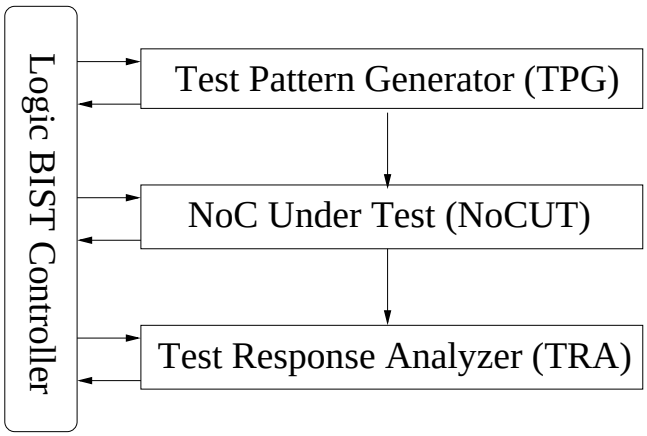


Figure 2.12: Basic built-in-self-test scheme for an NoC under test.

in the application. Unacceptably high area overhead and energy dissipation can adversely affect the NoC performance against the potential benefits of the NoC. The on-line testing is implemented in two ways. In the first approach, the NoC is allowed either to continue its normal operation for handling its usual duty or to stop the current application followed by turning the NoC into the test mode. Therefore, according to this approach, a binary decision is made on the NoC. As the disadvantage, unaccepted energy dissipation may be incurred since the whole NoC infrastructure is just under close scrutiny. In contrast to the first approach, the second approach allow both operational and testing modes together at the same time. It means the NoC is tested concurrently with handling its usual functions. In this case, a part of the NoC is put in the test mode while the rest part of the NoC is allowed an application. When application packets enter into the test zone, they either must wait in the border routers in the zone till it is switched to the operational mode or bypass this test zone and divert to another route by a suitable adaptive routing strategy [81, 127]. Abstractly, the on-line testing is preferred to the off-line testing to prevent the NoC architectures from the performance degeneration, e.g., latency [121] since the test time overhead in the former test technique is much low. Furthermore, the on-line structural testing is considered as an optimal test solution in the NoCs due to their critical responsibilities, predominant potential benefits, and special features over SoCs [122].

2.5.2 Test of NoC Building Blocks

Testing of the NoCs is classified into three parts on the basis of three basic building elements. Testing of the first part is related to the IP cores while so for the second and third parts are routers, and communication channels, respectively. Test approaches for these blocks are based on the functional or structural level. A functional-based approach is considered to reduce the NoC redesign costs while a structural approach is usually preferred for reducing the test application time, system performance degradation, and improve fault coverage.

2.5.2.1 Testing of IP Cores

Testing of IP cores in NoC-based systems, i.e., SoCs is very challenging because of their large scale integration into system design. Also, the detailed knowledge about how individual IP core is implemented in the system is unavailable. In parallel, it is difficult to access the IP cores embedded deep inside the chip. In SoC paradigms, the challenge is resolved by having a dedicated *test access mechanism* (TAM) [100] for the chip.

2.5.2.1.1 Test Access Mechanism

The TAM is a basic way of testing IP cores in NoCs and SoCs, that communicates test I/O data (test stimuli and test responses). It connects test sources that supply the test stimuli with an IP core under test (IP-CUT) and the IP-CUT to the test sinks where test responses are verified. Both test sources and sinks are implemented off-chip using an external ATE or on-chip via a BIST module. For small systems, an off-chip TAM is implemented whereas the test of IP cores in large systems prefers the on-chip implementation. One can also use a combination of both off-chip and on-chip modes when the combination of deterministic and pseudo-random test stimuli is exercised on the IP-CUT. In this case, one can use off-chip test source and on-chip test sink, or vice-versa [24]. Conceptual architecture of the TAM-based IP-CUT is provided in Figure 2.13. To ease the connection between the TAM and the IP-CUT, a test wrapper called network interface (NI) is designed. A wrapper may be simple adapter around an IP core and connects to a TAM [128]. This wrapper should be implemented in such a way that it possesses some type of bypass mode. The reason is that an IP core requires being isolated from the system when other IP cores are being tested. The TAM can be dedicated for testing purpose only. However, one can assume it as an existing structure. The advantage provided on the use of an existing structure as the TAM is that it minimizes additional routing of wires.

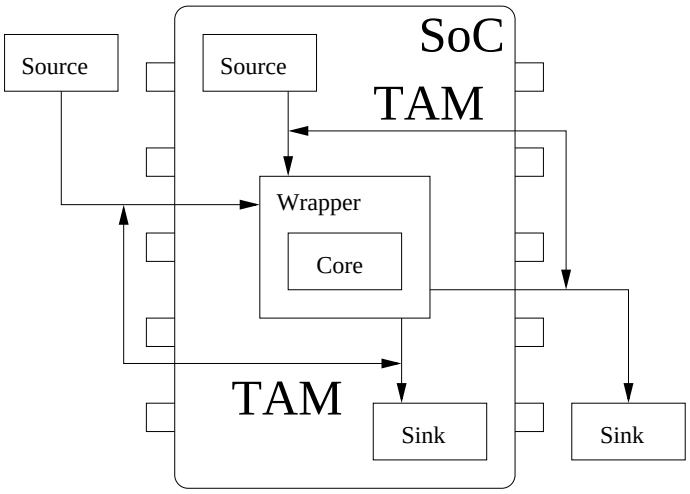


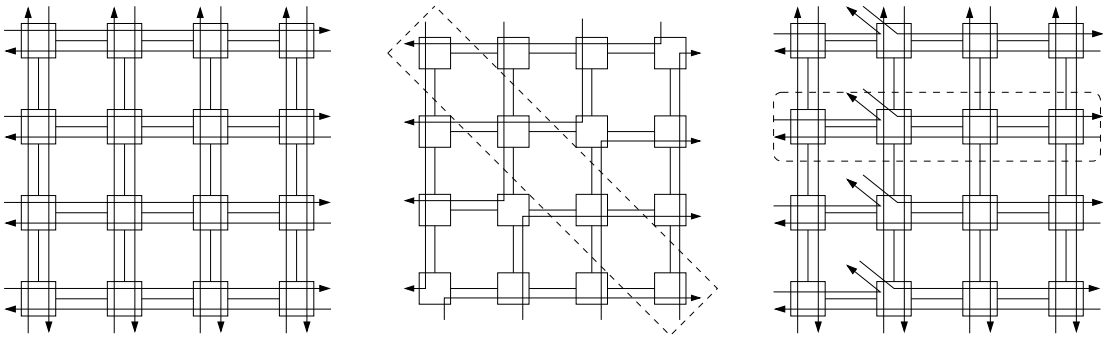
Figure 2.13: Conceptual architecture for testing of cores.

Most successful schemes for designing the TAMs combine the access mechanism and the heterogeneous IP core architectures. A TAM design consists of two parts that always search for the best trade-off between the test data transportation capacity of the access mechanism and its application cost. First, part is dedicated to the transport capacity which is determined by actual delivery of test data. This transport capacity is limited by the capacity to transport data by the test source and receive so by the test sink, and the subsystem that can be reused as the TAM. The second part is responsible for test data transportation that depends on the length and number of TAM wires termed as the TAM bandwidth. Consequently, this bandwidth infrastructure determines the test application cost in terms of TAM area overhead, pin area overhead, and the test time needed for an IP core. In a fully BISTed NoC-based system, each IP core owns a dedicated source and sink that has minimum TAM requirement. Whenever needed, these source and sink are added to the IP-CUT. Only a controlling module that would start and terminate the TAM is required. On the other hand, if existing subsystems of functional units are reused, then additional TAM cost becomes negligible.

Testing of IP cores basically focuses on speeding up the test development for every core-based designs as they aim at increasing design efficiency. In the form of dedicated DFT hardware, a TAM guarantees test access from embedded core to IC pins and vice versa, alleviating the test expansion task. Further, the mechanism supports several advantages, such as scalability, structuring, etc. By scalability, it does the trade-off between silicon area overhead and bandwidth for timely core test execution. On the other hand, by structuring it serves as a basis for the IEEE P1500 standard [129]. The TAM-based testing of IP cores is introduced for the first time by Marinissen *et al.* [130] and named the proposed scheme is named *Testrail*. Later different versions of this approach are followed by several researchers including the works discussed in [30–32, 131–134].

2.5.2.2 Testing of Routers

The reuse of an NoC as a TAM has been seen to be a cost-effective strategy for the test of IP cores embedded in the NoC. The cost-effectiveness is achieved in terms of silicon area overhead, test pin count, and test time. Simultaneously, one may claim that the network operations are tested when the NoC transmits test data. However, it is necessary for the NoC to define a specific test scheme before it is used as a TAM. Every test strategy, as a matter of fact, assumes that network infrastructures (routers and channels) have already been tested before being reused by the TAM [24]. It is then essential for the designers that a router must be tested for its various components. An NoC router consists of combinational as well as sequential logic blocks that include wide range of elements, such as arbiter, I/O ports, routing, error control, FIFO (I/O) buffers (as register banks or dedicated static random access memory (RAM) arrays), multiplexers, and so on. Two classes of test methodologies



(a) Straight paths. (b) Direction changes. (c) Resource connections.

Figure 2.14: Test configurations in first functional-based router testing.

for the test of NoC routers are basically studied in the literature. The first method is the functional-based test solutions. The second method is the structural-based test solutions.

Two representatives of the functional test solution are normally selected. The representatives are direct I/O access to the network [135], and NoC accessed from I/O through IP cores [136]. Direct I/O access to the network as the first functional-based router test scheme assumes that router ports at NoC boundary are accessible from I/O pins. Test patterns are applied at this external I/O as well local router ports. This approach provides two basic advantages due to extensive access. First, at-speed testing is made possible. Second, in comparison to a scan test scheme, test data volume is greatly reduced. On the contrary, this functional testing takes very high I/O pin overhead. Three basic test configurations on the basis of routing paths are used to define this functional router testing. These configurations are namely (a) straight paths, (b) direction changes, and (c) resource connections. Figure 2.14 illustrates these configurations. In the straight path configuration (Figure 2.14a), four routing paths viz. east-west (E-W), west-east (W-E), north-south (N-S), and south-north (S-N) are exercised. The direction change configuration covers routing paths for routers located on the central diagonal in an NoC. Figure 2.14b covers north-west (N-W) and south-east (S-E) routing paths. The resource connection configuration covers the local routing paths. Figure 2.14c illustrates the covering of north-local (N-L) and local-south (L-S) routing paths in the indicated NoC row. Similarly, other resource configurations, such as local-north (L-N), east-local (E-L), local-east (L-E), and so on are possible in the remaining NoC rows. In the NoC access from I/O through IP cores as the second functional-based router testing, it is assumed that the ATE access the NoC by means of a core's I/O pins. Getting test access to the NoC via core's I/O pins is shown in Figure 2.15. As seen, test patterns are applied at the first IP core and transported in the network. Test responses are collected at the sixth IP core via the functional I/O interfaces. An advantage provided by this functional test approach is that it does not incur any pin overhead. Additionally, the test of NIs is inherently included.

Several works add specific modes to activate scan chain and BIST modules for structural-

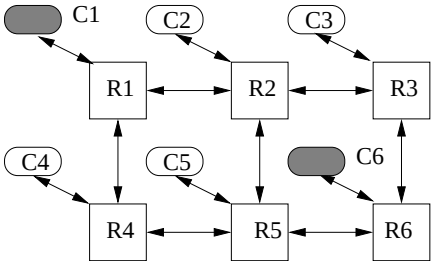
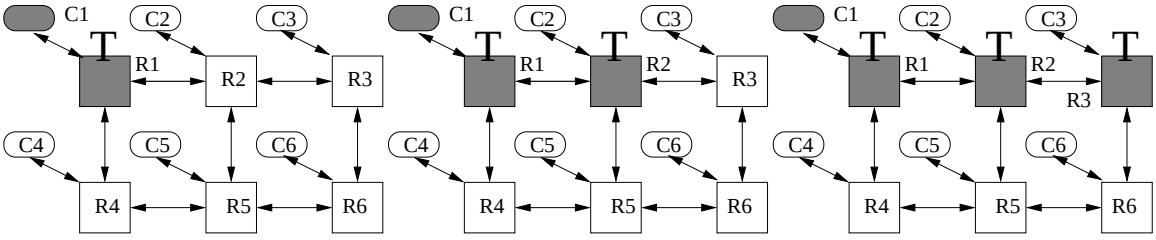


Figure 2.15: Second functional-based router testing.

based router testing. These approaches are grouped into three representatives of router testing schemes. These representatives are progressive use of already tested resources [137], partial scan with NoC test wrapper [12], and BIST for deflective switches [138]. In the progressive reuse of already tested NoC resources, the ATE access to the NoC is particularly ensured through a dedicated NI that acts as the unique test source enabling the enhancement of NoC access and the restriction on I/O overhead. Each router is connected to a dedicated IP core via the NI. Therefore, test patterns applied by the ATE at the NI are at first exercised for the router as the first network resource to be tested. Subsequently, the already tested routers (resources) are used to access the neighbor routers to be tested next. Thus, at each step, at least one router undergoes the testing. Note that the testing path carrying the test data is assumed fault-free. This path status has been proven in previous testing steps. Figure 2.16 illustrates the router testing method on the basis of progressive reuse of already test resources.



(a) Test step 1. (b) Test step 2. (c) Test step 3.

Figure 2.16: Progressive router testing.

In the partial scan with NoC wrapper method, the ATE access to the NoC via dedicated I/O pins is ensured for serial-in (SI), serial-out (SO), and control operations. The main idea behind this method is to insert partial scan chains in NoC routers. Since all routers are considered identical, same test sets are applied to all routers under test. If the routers are okay, i.e., fault-free, their responses are same with expected outputs. Test paths are created to broadcast the test sets computed by an automatic test pattern generation (ATPG) tool to NoC routers and internal comparators are implemented in the test wrappers in order to check the test responses. Figure 2.17 illustrates partial scan based router testing approach.

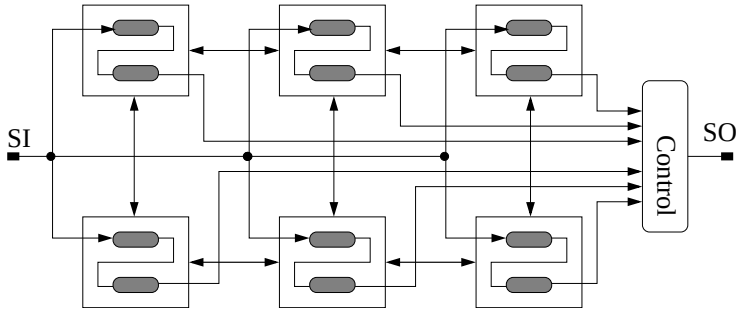
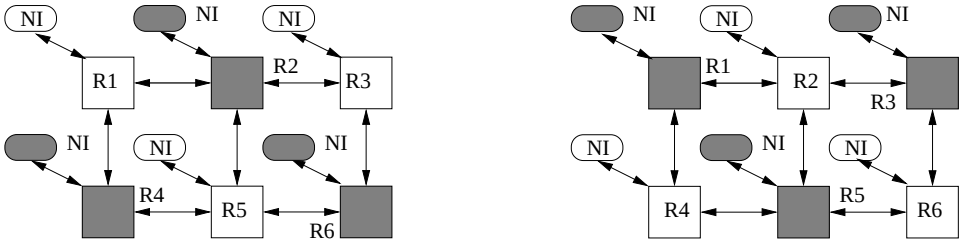


Figure 2.17: Partial scan-based router testing.

In the BISTed deflective router testing scheme, NoC routers echo or deflect the communication function. It means the data received back to the sender router are simply echoed by the deflection function. The test data then can be applied from NIs where test responses can be collected and processed as well in a BIST fashion. This third structural-based router testing is implemented in two phases as shown in Figure 2.18. In the first test phase (Figure 2.18a), one router set has tested individual data paths and channels, while another router set has tested control paths and echo. In the second test phase (Figure 2.18b), the role of the routers sets is reversed. For instance, first router set now has the control parts and echo functions for the testing since it had already tested data paths and channels.



(a) Test phase I.

(b) Test phase II.

Figure 2.18: BISTed deflective router testing phases.

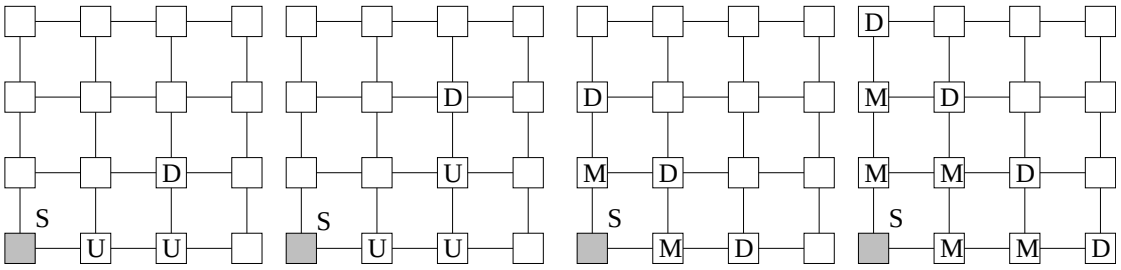
The fault models used in the above two methods (functional and structural testing) may be purely a functional model or a specific building block model. In the literature, the stuck-at fault is modeled to address the router's combinational logic. An ATPG tool is used to generate suitable test pattern set. Since the flow of data packets on the communication channels by nature is unidirectional, for the FIFO part, each FIFO possesses two-port memories for modeling its stuck-at faults. One port is labeled as *write-only* port while the another is labeled as *read-only* port. Additionally, transition faults or short faults as the dual-port coupling faults (DPCFs) are also modeled to test. In this case, FIFO's functionality is divided into the memory cell array, FIFO-specific functions, and the fault addressing mechanism [99]. For example, consider a *b-bit* wide FIFO that has p locations. An ATPG then generates p test patterns where individual test length has b-bits. On the other hand, if one wishes to

test for the DPCFs between bits b_i and $b_j; i \neq j$, four test sets in the following sequences: 0000...., 0101...., 1010...., 1111.... are needed. Each of these patterns can be generalized as $w\{\Uparrow_1^{p-1}(wr)\}r$ [3,139]. Here, w, r stands for write and read operations, respectively. Several works including [35,140–143] have been discussed in the past to include testing of NoC routers.

2.5.2.3 Testing of Communication Channels

The communication channels are one of the basic parts of an NoC fabric and occupy significant portion on a chip. This fabric, in turn, is constituted by routers and communication channels. If the IP cores on the NoC are connected in the point-to-point (P2P) way (Figure 2.7a), the message passing between the nodes is done directly. In absence of P2P connections, the communication between IP cores is fully dependent on correct functionalities of the NoC fabric, i.e., both routers and channels. The correctness of the NoC fabric as the communication infrastructure is the first and foremost thing to be ensured in order to guarantee NoC's correct operation. Similarly to the test of routers, testing of NoC channels is done either by employing the functional-based test strategies or by employing the structural-based test strategies. Both types of test strategies use the fault models termed as the link/channel fault models. The channel fault models may include different transient and manufacturing faults, such crosstalk, delay, stuck-at, short faults. Crosstalk and shorts faults in the models are considered in the same or different channels. Tailoring the test sets for detection and diagnosis of channel faults, one may wish to integrate the test of channels with the test of routers. But, testing of the NoC fabric in an integrated manner is a difficult task because routers and channels necessitate different fault models. For instance, memory fault models as router fault are used for FIFOs while logic fault models are preferred for faults in router's logic circuitry. On the other hand, channel faults models are used for channels which may be entirely different from the router fault models. Additionally, if the channels are not tested beforehand or do not show operation correctness, reuse of the NoC resources to transport test data becomes fallacious. In the literature only a few works [3] have been attempted to integrate the test of routers and channels in NoCs.

The test of NoC fabrics is generally performed in two ways. In a first way, test strategies for routers are applied while the testing of channel wires is performed in a second way. Testing of NoC routers, as well as communication channels, are accomplished with test data transportation on the NoC itself. Therefore, communication infrastructures available in the NoC can be essentially utilized for the purpose. Now routers and communication channels can be considered to be okay up to the certain testing stage and beyond it, these routers and channels need to be reused to deliver the test set for the test of next set of routers and channels. As a result, the test of routers and channels in NoCs must be done in phases. The routing scheme implemented at the routers transports the test data. It is expected that similar routers and channels are used in the NoC and at the same time same routing policy



(a) Step i. (b) Step i+1. (d) Step i. (e) Step i+1.

(A) Unicast data transport. (B) Multicast data transport.

Figure 2.19: Unicast and multicast based test schemes for router blocks in a NoC. S: source; D: destination; U: routers in unicast mode; M: routers in multicast mode.

may be considered in the test mode and normal functioning mode of the NoC. Basic NoC routers support two types of transportation modes. First one is the *unicast* mode. This is commonly used packet transmission mode in the NoC. In this mode, packets received at an input port of router are decoded and forwarded to one of the output ports. The output port is selected on the basis of the routing algorithm and the address of the destination node noted in the packet header. Figure 2.19A represents unicast-based packet transportation. The second transportation mode is the *multicast* mode. In this mode, each packet header contains the address of multiple destinations. Received packets at a router are decoded and forwarded to multiple output ports on the basis of packet's header. Figure 2.19B represents the multicast-based packet transportation. One advantage of this transport mode can be noted that packets can reach to the destinations in a faster and more efficient manner than the unicast mode. Generally, the communication channels in NoC architectures undergo the testing either functionally or structurally. It can be noted that these functional-based and structural-based test strategies include these two data transmission modes.

2.6 Literature Review: Test of Communication Channels

This section reviews the functional-based and structural-based test strategies to detect and diagnose faults in the communication channels of NoC architectures. The communication channels like the routers in an NoC usually have the regular structure. However, poor observability and controllability are shown by the channels due to high wire density and deeply embedded positions in on-chip layout [24]. Therefore, efficient use of suitable test sequences by a test method to ensure its application for the detection and diagnosis of faults on all channel wires is a great challenge from different perspectives, such as quality of service, network/system performance, reliability, and yield. To ensure the NoC functions correctly, its communication channels should be tested first prior to the testing of routers and their dedicated IP cores.

2.6.1 Quality of Service

The quality of service (QoS) is often discussed with the overall service of a network. According to Bjerregaard *et al.* [10], the term QoS is defined as "*a service quantification that is provided by the network to the demanding core*". In other words, the QoS in NoCs refers to the predictability of the communication behavior that defines the level of commitment for packet delivery. Such commitments can be the correctness or completion of on-chip packet transfers. Correctness is concerned with the integrity (corruption-less) of packets in addition to in-order delivery of the packets from source to destination. It is ensured by employing a test strategy, correcting errors, or retransmission of the packets. Completion, on the other hand, is ensured by the packets which are not dropped or lost in the network when being transmitted from source to destination. Additionally, it also ensures no deadlock or livelock conditions in the NoCs. Generally, a flow control mechanism for communicating the packets between source and destination is employed to ensure no packet loss and freedom from deadlock/livelock conditions. The QoS requirements have classified NoCs into three basic categories: *best effort* (BE), *guaranteed service* (GS), *differentiated service* (DS) NoCs. In BE NoCs, correctness and completion are guaranteed on NoC communication behavior. No other commitments, such as bounds on network performance can be made. Packet switched NoCs belong to this category. In GS NoCs, network throughput in addition to correctness and completion is guaranteed. As a result, the NoCs make a tangible guarantee on the performance. Virtual circuit switched NoCs are the GS NoCs. In DS NoCs, priority-based communication takes place. NoC routers employ priority-based scheduling and allocation strategies. Unlike GS NoCs, such priority-based approaches do not provide strong guarantees rather than enabling higher resource utilization.

2.6.2 Network Performance

The QoS, in most cases, refers to bounds on network performance and is a quantified measure of various performance metrics, such as bandwidth, throughput, latency, and power consumption. Throughput and latency, however, are the most important metrics considered to be evaluated in on-chip interconnection networks [98].

Bandwidth is defined as the maximum rate at which a message block in terms of packets is transferred in on-chip networks. This bandwidth is measured in the unit of bits per second (bps). Usually, the whole packet that consists of the header, payload, and trailer flits is considered as the channel bandwidth.

Throughput is defined as the maximum traffic accepted by the network. Alternatively, accepted traffic and throughput sometimes are considered as the fraction of network capacity. Here, the traffic is an application data, information, or a message delivered in terms of packets. It is measured in message per clock cycle, message per second. In case of NoCs, each packet

consists of multiple flits and accepted by IP cores over a specific time period. This time period may be absolute or relative time, e.g., simulation time used in the network to deliver the packets. Further, this throughput is normalized by dividing the accepted traffic with network size and the time duration. One can thus have the throughput unit in NoCs as flits/cycle/IP.

Latency is defined as the time elapsed between transmission of a message from the sender and receipt of the message at a destination. This definition is interpreted in different perspectives. For instance, the latency of a packet in NoCs is the time elapsed from when the packet header is injected from the source to the receipt of the packet trailer by a target. Note that a routing path between source and target nodes may consist of the interconnection of routers and channels alternatively. In that case, the packet transported on the path may wait at buffers of intermediate routers of the path. So, this waiting or queuing time as transport latency is added to the packet latency. Latency of a packet is measured in time units, e.g., sec, μs, clock cycle. Latency of individual packet is not so important because applications depend on the traffic size (data volume), different source-destination pair, and routing algorithm. Then, each packet may have an unequal latency.

Power consumption is the energy consumed by packets while traveling in networks. Each packet from source needs to travel a routing path of multiple hops (a hop is single channel length) before reaching the destination. During its traveling on the channels and forwarded by intermediate routers, both channel wires and router logic gates toggle which result in energy dissipation. This energy dissipation is static and dynamic. But, NoC designers are more concerned with the second category of energy dissipation by packet flits and caused due to communication process [5]. The dissipated energy is normally measured in $\mu J/flit$. One can measure the power consumed by a packet simply by dividing its dissipated energy with the latency.

2.6.3 Reliability and Yield Improvement Issues

In the current scenario, it has been found that the NoCs have moved over time from the computation-centric to communication-centric designs for the ease of their implementation as the scalable communication architectures [10] and to meet the increasing demand of high-performance computation and communication for many applications in the multicore systems. However, high wire density has increased the number of various transient and manufacturing faults in the communication channels. Therefore, these faults must be tackled prior to forwarding of any data on the channels as they are often susceptible to these faults. Subsequently, the basic need for reliable communication as well as yield in an NoC system becomes major concerns despite the fact that IP cores and routers are protected from faults in the system. The reliability and yield issues due to channel faults are primarily addressed

by following four practices: replacement policy, recovery scheme, fault-tolerant routing, and test mechanism.

2.6.3.1 Replacement Policy

By the first intuitive approach, the designers replace the faulty wires on the immediate identification of the wires in a channel while these wires are affected by manufacturing faults. The replacement of a faulty wire takes place by a spare non-faulty wire. As many as channel wires are identified as faulty, the same number of additional non-faulty wires must be embedded in the on-chip architecture [144, 145]. Although this replacement policy maintains the manageable network performance it becomes costlier as well as adds high silicon area overhead. The approach as anticipated cannot be a choice for the circuit designer.

2.6.3.2 Recovery Scheme

Manufacturing faults on channel wires are permanent and are not recoverable. On the other hand, if the channel wires are detected as faulty and affected by a transient fault, then any of the error control schemes discussed in [27, 146–149] can be used as the error recovery method that corrects the erroneous received packets due to a transient fault in control and data paths. These schemes normally use existing automatic repeat request (ARQ), forward error control (FEC), and hybrid ARQ/FEC techniques [26, 150] which are in general used during data communication in the traditional computer network system. Note that the selection of replacement and recovery strategies leads to the objective of "*not to allow any fault*" in channels.

2.6.3.3 Fault Tolerant Routing

Since the replacement schemes incur hardware area overhead and additional manufacturing cost for an NoC architecture, while the recovery schemes are able to correct only the soft errors, the most intuitive approach to tackle both permanent and transient faults in channels is to employ a fault-tolerant routing algorithm [43, 48, 151–154]. The adaptive feature of the routing algorithms provides the potential in order to route application packets on the non-faulty channels by avoiding the faulty channels or transport the packets on the faulty channels by their partial utilization (non-faulty part). A fault-tolerant routing then deals with a faulty channel so that the network may still be operational. Either of these packet transmissions achieves the fault-tolerance. However, prior knowledge about the health status of a channel in a routing path must be available to an adaptive fault-tolerant routing method. Note that unlike the replacement/recovery scheme, the fault tolerant approach "*allows to accommodate faults*" in the communication channels of the NoC architectures.

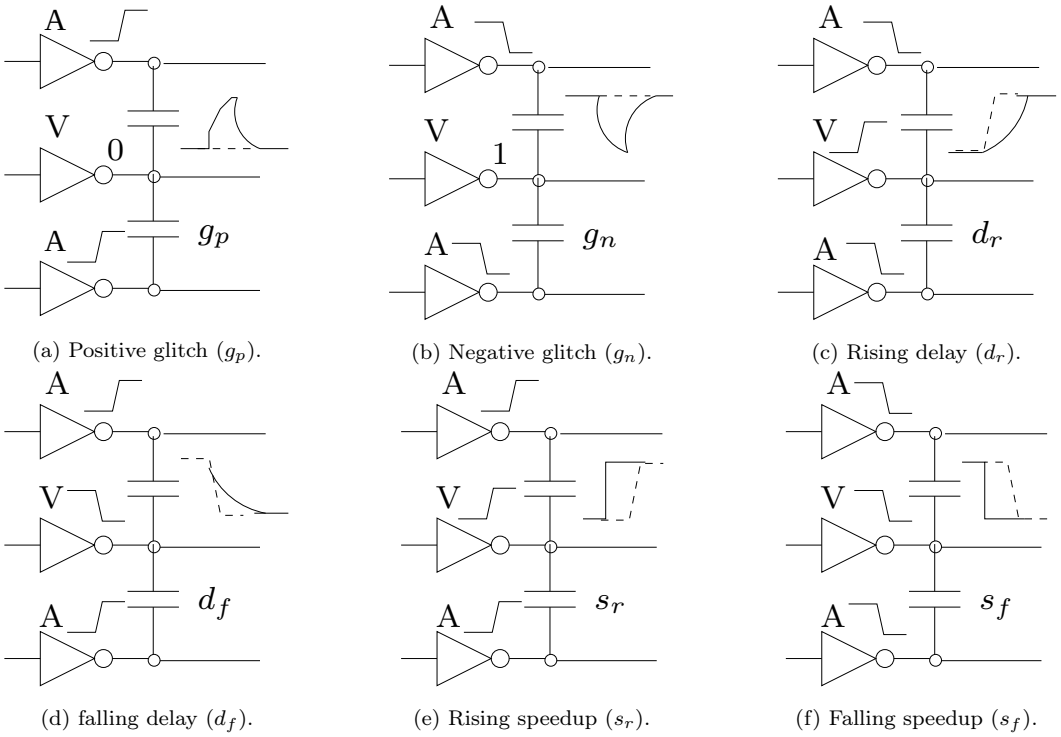
2.6.3.4 Test Strategy

The next and perhaps final approach to enhance reliability and improve the yield of an NoC infrastructure is a test strategy that has the capability in detecting channel faults and identifying faulty channel-wires in the NoC [33, 38, 155]. Various test mechanisms for transient and manufacturing faults in channels are discussed immediately after this subsection, i.e., in Subsections 2.6.4 and 2.6.5, respectively. The diagnostic information from the test mechanism can be reused by a fault tolerant routing that may reconfigure routing path for the better utilization of fault-tolerant resources [28] and keeping the system operational. It can be remembered that the sequence of testing NoC basic components matters and must be prioritized in the testing cycle. Consequently, testing communication channels in the priority sequence must be done before the routers in the NoCs because, as it is seen in the literature, already tested network resources are reused in the testing of the next set of routers. Similarly, reuse of the NoC infrastructures as the TAM has been followed in testing NoC's processing elements, i.e., IP cores. In both cases, communication channels carry both test data as well as test responses. In such circumstances, communication channels are a fundamental element for transferring testing data in NoCs. So, we must make ensure that communication channel wires show proper functionality before using them as the test instruments in testing routers and IP cores. As we will see, most of the works have considered interconnect (channel) faults in interswitch channels only. Furthermore, many contributions have shown interests to these faults affecting the data wires of the channels. It has been also seen that some works [3, 6] try to integrate the test of communication channels with the test of routers in an NoC. In the subsequent subsections, we go through the state-of-the-art on testing channels for both transient and manufacturing faults. The fault models practiced in both classes include crosstalk, stuck-at, open, and short faults.

2.6.4 Testing of Transient Faults in Channels

The signal integrity and related issues usually lead to transient faults, such as crosstalk faults in NoC channels. The test mechanism involved in the detection of transient faults in the channels should distinguish these faults from the manufacturing faults. In this subsection, different methods for testing of transient faults are focused.

Cuviello *et al.* [120] have first introduced the crosstalk fault model for the DSM SoC interconnects (channels). The model is named as maximal aggressor fault (MAF). The effect of a crosstalk fault is realized in terms of three errors namely, delay, glitch, and speedup (oscillation) at either positive/negative or rising/falling state. Figure 2.20 illustrates the MAF model for the crosstalk effects. In this model, a fault is assumed to affect only one wire at a time. This wire is termed as victim (V) wire. The remaining wires are designated as aggressor (A) wires that collectively act to generate an error condition on the V wire. For a



(a) Positive glitch (g_p). (b) Negative glitch (g_n). (c) Rising delay (d_r).

(d) falling delay (d_f). (e) Rising speedup (s_r). (f) Falling speedup (s_f).

Figure 2.20: The maximal aggressor fault model for different states of a crosstalk fault. A and V represent a aggressor and victim wire, respectively.

channel (also called bus) of W wires, it is observed that different combinations of aggressors have enumerated $W2^{W-1}$ as the total number of possible crosstalk faults. The fault size is prohibitively high from a test-coverage point of view. The MAF model on consideration of the worst case coupling impedance combinations among A wires has reduced the fault size. In this case, transitions on all $W-1$ aggressors in a channel are assumed in the same direction as a fault. As a result, the fault size is reduced to $6W$ faults on the W-line wide channel that requires same number ($6W$) of two-pattern tests.

The MAF model stands as an abstract representation of the set of all transient defects that can lead to one of the six crosstalk errors: positive/negative glitch, rising/falling delay, and rising/falling speedup by the aggressors on the victim. This model is later adopted by several researchers [3, 29, 139, 156–159] in order to consider channel's crosstalk effects. For example, Grecu *et al.* [3,139] have proposed a test set reduction scheme to account crosstalk faults in interswitch channels in addition to the test of routers of an NoC-based system. In this scheme, two consecutive test patterns are applied on the aggressor wires of a channel that provoke the signal transitions for the clear observation of the fault effect to appear on the victim. These two test patterns are combined such that the whole test set is reduced to only eight test patterns. As shown in Figure 2.21, the stimuli pairs sensitize each crosstalk error in the MAF model.

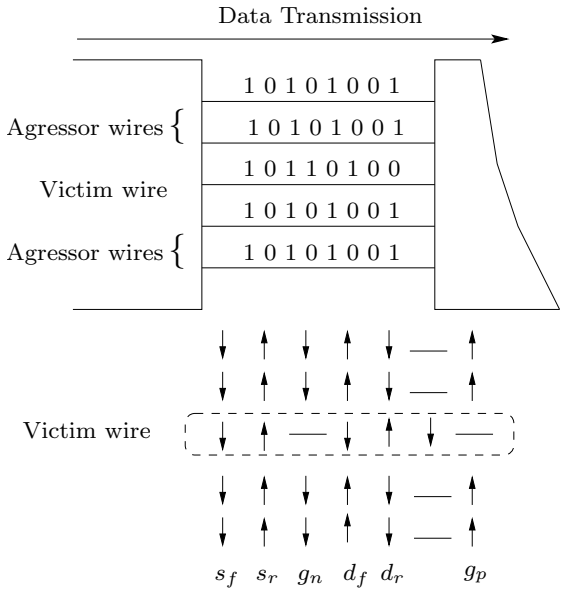


Figure 2.21: Optimize test sequence for the MAF model.

Beside the cross-coupling capacitance and impedance speculation, on-chip channels are often vulnerable to transient faults due to miscellaneous temporary conditions, such as process variations, external radiation, spurious voltage spikes, reduced noise margins, connectivity issues or service unavailability between nodes [160]. Because of the temporary behavior, the transient faults are generally managed by several error control coding (ECC) schemes, such as automatic repeat request (ARQ) transmission and forward error correction (FEC), and hybrid ARQ/FEC which are discussed time to time in [27, 53, 149, 161, 162]. For example, Pande *et al.* [53] have proposed a crosstalk avoiding double error correction code (CADEC) scheme to enhance system reliability. The scheme incorporates two types of coding techniques in the NoC data stream. One is crosstalk avoidance coding. Another is forward error correction coding. Qiaoyan *et al.* [149] have proposed a co-management method for transient errors in a channel. In order to manage variable transient errors, the reconfigurable error control coding is used.

2.6.5 Testing of Manufacturing Faults in Channels

It is impossible to anticipate the signal integrity problems in advance that cause transient faults in channels. Further, these are temporary faults which can be recovered by many error recovery schemes as mentioned earlier. Design errors, design rule violations, and wrong manufacturing process on the contrary encompass to three basic faults: stuck-at, short, and open in the channels which are commonly known as manufacturing defects. These faults remain active in channels. Also some of these faults may affect other parts of the NoCs, such as router components. The faults exist in channels and other parts in long-term basis until

the faulty components are separated from the NoC system. For the reason, these faults are referred to as permanent faults. Additionally, these faults may significantly aggravate the cross-coupling effects. Hence, the need to test for permanent defects leading to a transient fault is mandatory because permanent faults act as the agents to various system-level failures. For example, stuck-at faults bring the network into packet corruption, packet misrouting, and packet dropping (loss) failure modes. Similarly, short faults are the agent to packet duplication (overloading) in addition to packet misrouting and packet dropping, while the open faults have shown to be the agent to partial and full packet loss. Therefore, permanent faults in channels are crucial for the NoC performance in terms of performance characteristics: throughput, latency and power minimization, quality of services (QoS), fault tolerance policies, and traffic characterization and modeling [44].

Efficient fault models [106–108, 113–118] are designed in the past in VLSI circuits for manufacturing faults. One basic disadvantage of these wiring fault models is that they are designed without any routing constraints in the system. Subsequently, these faults models have been reused by a group of researchers for the analysis of manufacturing faults in channels in order to maintain reliability and yield issues in the NoC-based communication systems. On-chip communication channels are two types- interswitch and local. Each channel irrespective of its type further consists of three set of wires: control, data, and handshake wires. Manufacturing faults can be experienced on these channel wires. Researchers, like as the transient channel faults have shown their interests in the test of permanent faults only in the interswitch channels or both in interswitch and local channels with or without the test of routers in the NoCs.

Cota *et al.* [37] as found in the literature have first proposed an off-line functional-based test mechanism that addresses pairwise short faults in 2×2 network neighborhood of commonly used mesh NoCs. This method named here 2×2-Model reuses the wiring fault model in the neighborhood to include short faults on interswitch and local channel wires. This approach also extends to include short faults that may affect wires in distinct channels of the neighborhood. As in most wiring fault models, AND-type short faults are modeled by the 2×2-Model that includes a built-in-self-test scheme. The scheme implements test data generators (TDGs) and test error detectors (TEDs) which are placed inside the IP cores of the neighborhood. Considering the XY routing strategy, test data are transmitted by a TDG from an IP core to the TED in another IP core which is located over four hops for the analysis of test responses. A four hop routing path consists of two interswitch channels and two local channels. The 2×2 neighborhood has four IP cores, each is wrapped with a network interface (NI). So, four TDGs and TEDs are placed in these IP cores. It means four routing paths treated as the test paths during testing are simultaneously activated for test data transmission. The walking-one sequence [118] as the required test set is applied to each routing path. Figure 2.22 illustrates these test paths using four different arrows: one with

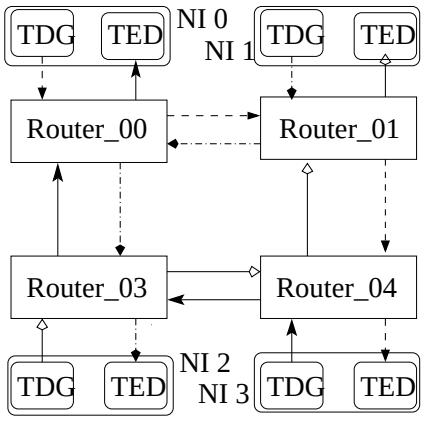
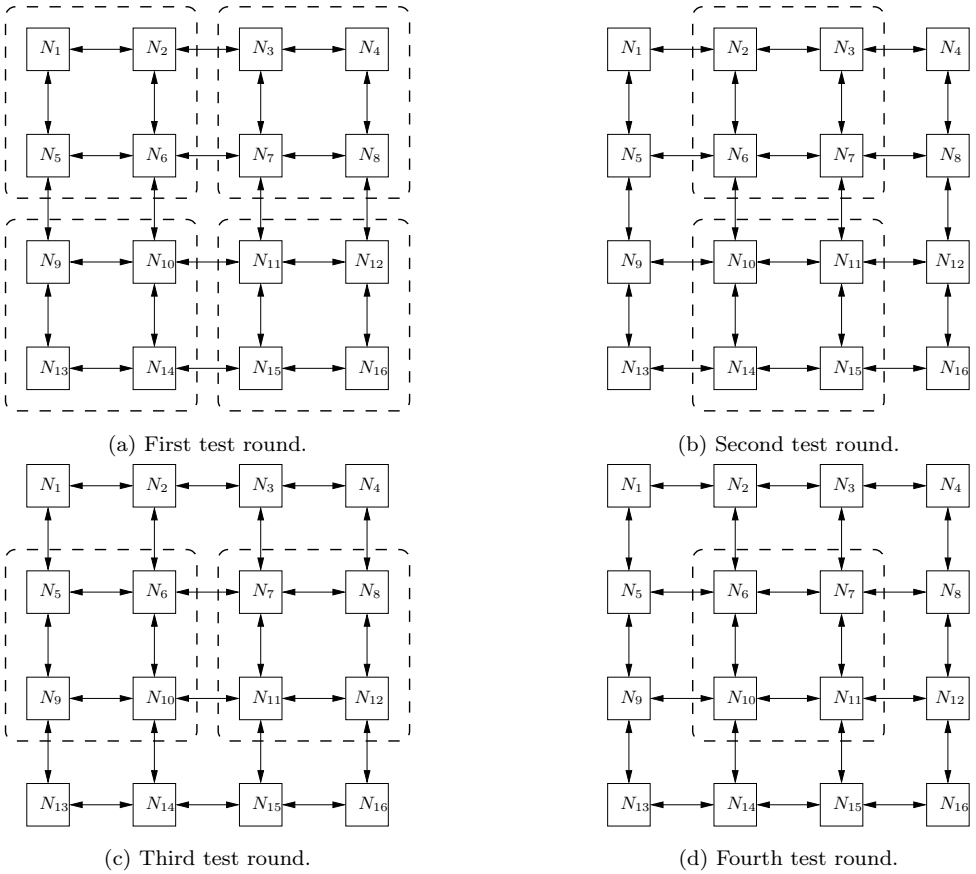


Figure 2.22: The 2×2 neighborhood-based test configuration.

the solid line, one with the dashed line, one with the dash-dotted line, and one with the solid line and open arrow. One can consider the 2×2 neighborhood as shown in Figure 2.22 as a 2×2 sub-NoC. For the detection of channel short faults in larger $M \times N$ mesh networks, authors have proposed possible 2×2 sub-NoCs to be concurrently applied in a test round on an $M \times N$ mesh NoC. Whatever may be the size of this network, multiple applications of the 2×2 neighborhood-based test configuration can cover the channel shorts in four test rounds. Figure 2.23 illustrates these four test rounds on a 4×4 mesh NoC where the same instance of the test configuration is simultaneously applied.

The current approach has considered only the inter-channel shorts involving the data wires, although control, as well as handshake wires, can be affected by these shorts. Authors have extended this test scheme in [38] to include inter-channel short faults in the 2×2 neighborhood. In addition, a simple algorithm is proposed to decide the test configurations when they need to be applied to larger mesh networks. Considering the fault diagnosis capability of the test solution discussed in [38], it shows poor diagnosability in contrary to the test application time while mesh networks become larger in size. The reason is due to the concurrent application of the basic test configurations. This poor diagnosability has been overcome in the test scheme proposed by Herve *et al.* [163]. Instead of single test cycle (iteration or round) used in detecting pairwise inter-channel shorts in the 2×2 neighborhood, these faults are now detected in five test cycles using additional test sequences on the different test paths. Furthermore, the basic 2×2 test configuration is now applied serially instead of applying it in parallel on a larger mesh network. For example, inter-channel shorts on a 4×4 mesh network now can be detected in nine test rounds instead of four as shown in Figure 2.23. The price to pay in terms of the test application time for enhancing this fault diagnosis capability increases significantly. Though, the 2×2-Model scales with the mesh NoCs irrespective of their size, however, this 2×2 neighborhood is usually not possible on other networks such as torus, octagon, butterfly tree, etc. because of the typical architecture



(a) First test round. (b) Second test round.

(c) Third test round. (d) Fourth test round.

Figure 2.23: Concurrent application of 2×2 test configurations on a 4×4 mesh network.

of the respective networks. For example, every border router in a torus network has point-to-point (P2P) interconnection with another border router in the opposite direction. Similarly, every corner router has shared a communication channel with its opposite corner router.

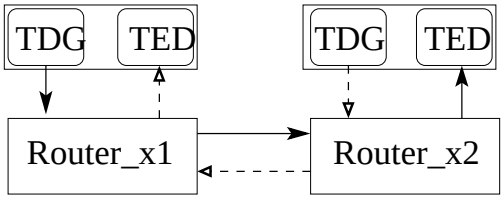
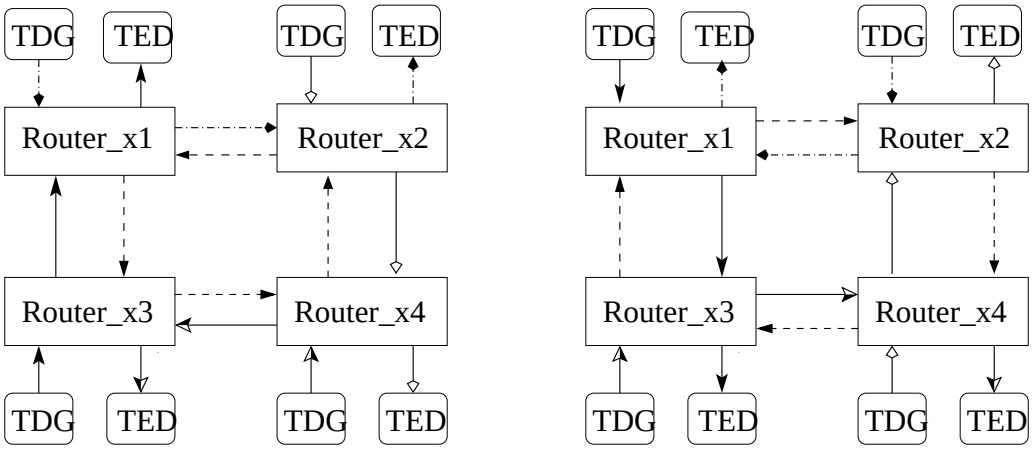


Figure 2.24: The 2×1 neighborhood-based test configuration.

Concatto *et al.* [40] have proposed an alternative BIST method to test pairwise channel short faults in order to increase the yield of a torus network. The proposed approach partially uses the 2×2 model and the extended test model named here partial 2×2 ($P-2\times2$) model that assumes a 2×1 neighborhood (Figure 2.24). The test configuration in this neighborhood is applied simultaneously on the activated alternative paths that take the advantage of inherent redundancy of the torus topology. As the method is executed in the off-line mode and the

test data are analyzed after traversing three channel lengths, the $P-2\times2$ model is supposed to incur comparatively low area overhead but high test time than the 2×2-Model. However, this test application time can be reduced if a 2×2 neighborhood on internal routers in torus network undergoes the testing in two test cycles. This approach is followed in [41] where the multiple 2×1 test configurations are concurrently applied in a test cycle on the 2×2 neighborhood. The concurrent application of the basic 2×1 test configuration is shown in Figure 2.25.



(a) Test applications in the first test cycle.

(b) Test applications in the second test cycle.

Figure 2.25: Concurrent test application of 2×1 test configurations on the 2×2 neighborhood. Channels denoted by gray color remain unused in a test cycle.

The 2×2-Model and its extended versions have been basically proposed for the detection and diagnosis of pairwise short faults only in the channels of a 2×2 neighborhood of a mesh network but no other manufacturing faults in the channels have been considered. Later this 2×2-Model is exploited by Herve *et al.* [39] where authors have extended the fault model that comprises stuck-at faults in addition to pairwise shorts in channels of the neighborhood. The test applications for the detection of shorts and stuck-at faults in larger mesh networks are similar as followed in [37, 38].

The previous test strategies in all the cases are practiced in the off-line mode where a test strategy simply stops the current application in a network and puts the whole network into the test mode for the detection of a fault in the network components, such as channels. However, there are many mission-critical systems [44, 102, 164] that do not allow any interruption in the ongoing applications in these systems. The ideal solution is the application of a test mechanism in the on-line mode where a part of a network is kept in the test mode for possible channel failure while rest part of the network is in the operational mode. If an application packet is received by a router involved in testing its channels and needs to forward the packet, then it has to wait at the router till the testing is completed or is diverted to another route depending on the routing strategy.

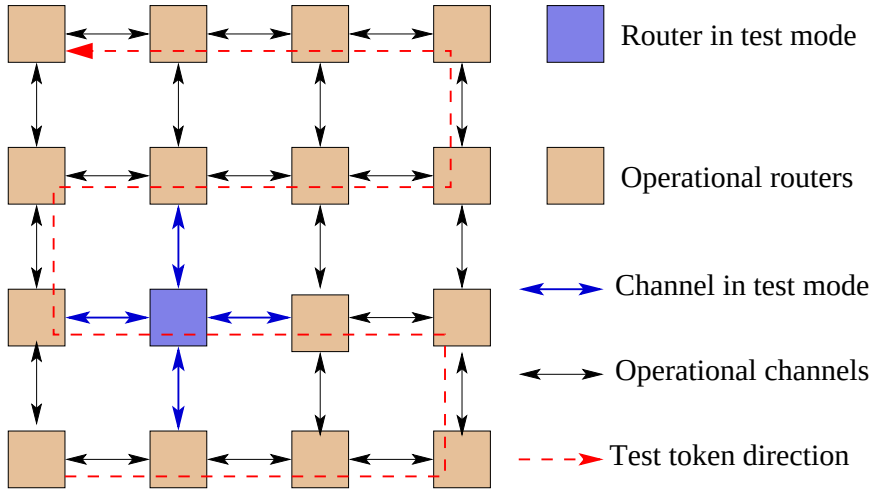
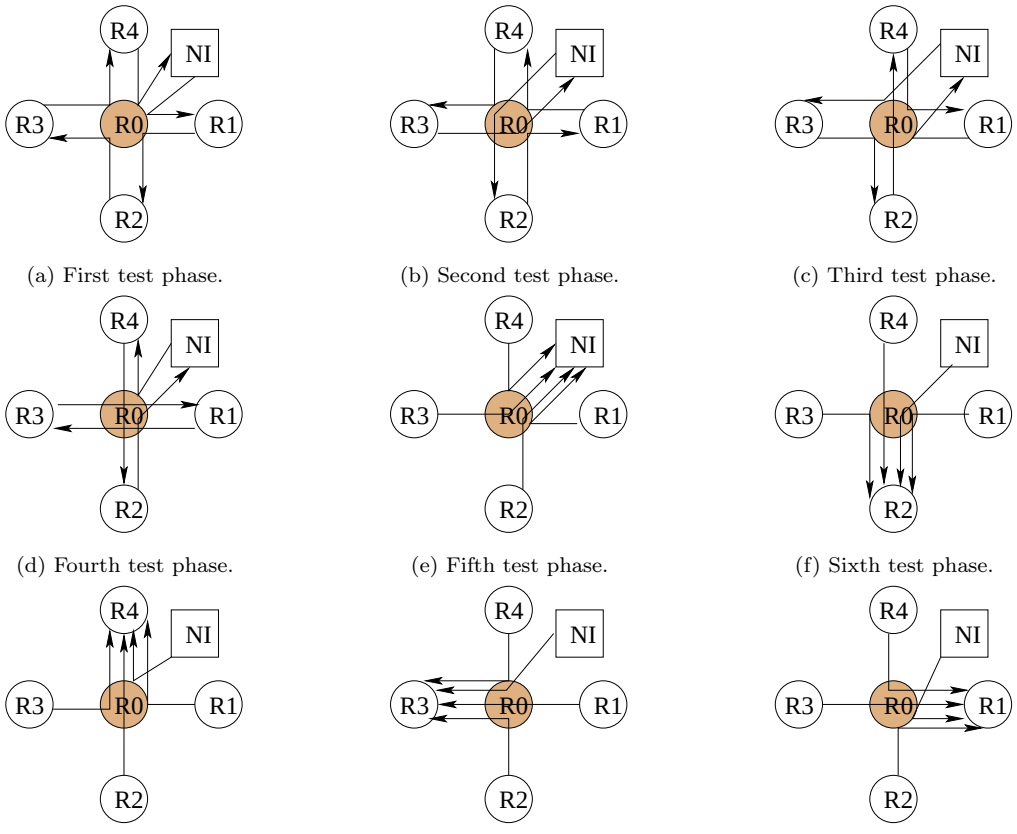


Figure 2.26: Sequentially one router selection method for the on-line testing of channels on a 4×4 mesh network. Each router which gets a token can start testing itself with the help of its neighbors. Only one router and its links are in the test mode and others are in the operational mode.



(a) First test phase. (b) Second test phase. (c) Third test phase.

(d) Fourth test phase. (e) Fifth test phase. (f) Sixth test phase.

(g) Seventh test phase. (h) Eighth test phase. (i) Ninth test phase.

Figure 2.27: Different test phases for a router having four neighbors and one dedicated network interface (NI). First four phases cover data path and all routings without any arbitration while rest of the phases cover control logic in arbiter and FIFOs.

Kakoee *et al.* [6,7] have proposed a token-based on-line test mechanism to address stuck-at faults in interswitch channels. The mechanism is integrated with the test of router FIFOs for stuck-at faults. The method selects only one router at a time from a network. Figure 2.26 illustrates sequential router selection method (named here S-Model). Once a router is selected, the neighborhood that consists of the router itself, its channels and neighbor routers are put in the test mode for stuck-at faults in the I/O buffers, router logic as well as in the interswitch channels. The test mechanism works in two steps. In the first step, interswitch channels of the router under test (RUT) undergo stuck-at fault testing. In the next step, the RUT is considered. These two steps of the test method detect the faults on these network components (channels and routers) by passing the neighborhood through different test phases as shown in Figure 2.27. There are nine test phases that the test method iterates. First, four test phases are used in testing faults in channels of the neighborhood. Anyone from these four phases is executed in two test cycles. In the first test cycle, test data set is transmitted from the router, say P to its neighbor routers, say Q where the received test set is analyzed. The test set on the other hand in second test cycle is transmitted from the Q to the P where the received test set is analyzed. Thus, outgoing and incoming interswitch channels undergo testing for stuck-at faults in the first and second test cycle, respectively. In addition to the stuck-at faults, the S-Model is supposed to address pairwise short faults in these channels. These first four test phases are also used for covering faults in router logics while rest of the phases are dedicated to testing the functionality of the arbiters. One advantage of this test methodology over the previous schemes is that it scales with large-scale general NoC architectures. But the great disadvantage is that overall test time is very high due to sequential selection of routers and linearly increases with network size assuming that channel width is same. Furthermore, the scheme assumes these short and stuck-at faults in interswitch channels only but does not include them in local channels which can likewise be affected.

2.7 Issues Related to Prior Works

With the test of the communication channels of an NoC architecture discussed in Section 2.6, the literature survey has thus covered altogether the test of transient and permanent faults in channels of the NoCs. It is also seen that permanent faults are more critical than the transient faults on such communication architectures. It is noted that an NoC as the emerging communication infrastructure is continuously replacing the SoC-based systems and delightedly meets the high bandwidth requirement of many complex and voluminous applications. Subsequently, more and more metallic wires in support of these applications, are shaped to an NoC channel which is much inclined to basic logic level manufacturing faults due to several reasons, such as high wire density. As a result, severe impact on the system performance may be observed along the reliability and yield concerns. In order to guarantee

the outgoing quality of service and minimum performance degradation while not sacrificing yield and reliability, these communication channels of NoC-based systems must undergo a test for permanent faults both at the pre- and post-manufacturing stages. Different issues of the existing test schemes have been noticed, which account for a large portion of the overall manufacturing cost in order to address the reliability and related challenges, such as yield, and QoS [121, 165]. Here, the testing issues (TIs) are classified into the following ways: (a) maintaining system reliability and yield, (b) test size, (c) hardware area overhead, (d) test time, (e) fault coverage, (f) on-line performance overhead, (g) method scalability, (h) fault efficacy, and (i) test energy. First, these issues are briefly discussed and then the way of handling them is mentioned with respect to the contributions of the thesis.

TI-A. Maintaining System Reliability and Yield: In Subsection 2.6.3, it has been seen that the improvement on the issues of reliability and yield of the NoC-based systems is preserved by four intuitive approaches that include replacement policy, recovery scheme, fault-tolerant routing algorithm, and fault detection and diagnosis strategy. One basic prerequisite or precondition of the aforesaid former three approaches [27, 144, 154] is the identification of faulty wires in a channel prior to their implementation for taking counter measurements against the channel's faults [26]. On the contrary, only the last approach has the ability to detect a fault on a channel and locate the faulty channel wires. It is seen from the literature that most of the existing fault recovery, as well as fault-tolerant schemes in NoCs, do not come up with a testing mechanism [6, 7].

TI-B. Test Size Reduction: Every test mechanism exercises a suitable test data set for the detection and finding the location of temporary and permanent channel faults. The size of the selected test data set (test size) generally varies with various parameters, such as channel width, channel type, faulty category, the way of implementation of the test mechanism, network topology, and routing.

TI-C. Silicon Area Overhead: Every BIST-based test mechanism is executed by TDGs and TEDs that handle the test set for a fault. Besides the test size for a fault, the area taken by these hardware units in an NoC architecture depends on the route of the test set. If the routing path length between a TDG and a TED pair is more than a hop, the TDG has to deliver large test set on the path and the TED analyzes the corresponding test responses. Therefore, for the generation, delivery, and analysis of large test data set, these hardware blocks may take a significant portion of an NoC node resulting in higher area overhead. In addition to routing path length, the area of TDGs and TEDs depends on the consideration of faults whether to include them in interswitch channels only or both in interswitch and local channels.

TI-D. Test Time: The amount of test time which is needed to detect and diagnose the faults in a channel depends on the time required at each phase from the generation to the analysis phase of the test set. Additionally, the test mode whether off-line or on-line influences the test cost in terms of the test time. In a test mode, the overall test time depends on various factors, such as the size of the test set to be handled, number of test iterations taken to cover the faults in channels, and the amount of time incurred on a test iteration.

TI-E. Fault Coverage Metric: Every test method intends to maximize the detection of faults in the channels on exercising the test sets. A fault coverage metric defines the number detected faults over the injected faults. Therefore, a test mechanism may provide the fault coverage up to 100%. However, it becomes possible when this mechanism uses single fault model, i.e., assumes only one kind of faults in channels and single hop test data transmission. The achievement can be considered as an advantage of using the approach for a fault. Routing distance between TDGs and TEDs has the important role on the fault coverage achievement. Multi-hop transmission of test data may prevent a test method to reach this highest fault coverage even on the consideration of single fault. In this case (multi-hop transmission), the fault coverage can, however, be achieved up to 100% but at the cost of additional test data which in other words results in high area overhead. Faults in channels can be coexistent, e.g., short and stuck-at faults can appear simultaneously on the channels. The corresponding fault model is referred as multiple fault model. In case of coexistent faults in channels, it is natural that a fraction of the faults remains undetected resulting in the issue of fault non-diagnosability.

TI-F. Performance Overhead: The number of test iterations, as well as the number of test clocks per iteration, have a direct influence on the network performance behavior. It is more prominent when a test scheme is applied in the on-line mode where application packets wait at the nodes involved in the testing of its channels. If the number of test iterations is high, then a test packet may pass through multiple test iterations. Correspondingly, more network resources are accessed in transporting the packet as many times as it passes the test zones. Subsequently, network performance metrics, such as packet latency, power consumption of a packet are notably raised. Thus, overall system performance is seen to be degraded. Prior test schemes show degraded performance due to their higher number of test iterations.

TI-G. Method Scalability: The scalability is one of the important issues for a test approach. The scalability issues can be demonstrated with respect to different aspects, such as application scope of the test approach. The scalability feature with respect

to application scope considers the fact that whether the test approach is applicable to an NoC with respect to its size, channel width, and type without or little compromise in one or multiple basic testing related issues: TI-B to TI-F.

TI-H. Fault Efficacy: Manufacturing faults at the logic level is categorized into short, stuck-at, and open faults. A test method for the testing of basic logic level manufacturing faults does not include short and stuck-at faults only but should also consider open (cut) faults. If a test strategy is designed on the assumption that communication channels cannot experience any open fault in channels despite the fact that an open fault is as natural as short and stuck-at faults and may be similarly responsible for performance degradation in the NoC systems, the strategy is then said to represent a *fault efficacy* property. The term "fault efficacy" defines the deficiency in covering the manufacturing faults in NoC channels by a test scheme. Without considering the open faults in channels, a test method thus cannot be considered to be efficient for permanent faults in NoC channels.

TI-I. Test Energy: During testing of a fault whether permanent or temporary in the channels of on-chip networks, different network resources are utilized by an underlying test method. One of the important network resources is the dissipated energy needed for the completion of the on-going channel testing [5,53]. The amount of the dissipated test energy primarily depends on the working policy of a test scheme. When the hardware area overhead due to the handling of large test data for a fault, the size of a subnet put in the test mode, and the high test time due to a large number of test iterations are observed for a test scheme, they directly affect this test energy metric. For example, many channels during testing may enter the test mode more than once, although they are already tested. Subsequently, BIST blocks unnecessarily need to process the test data resulting in the increase of dissipated energy.

Existing test schemes discussed in the previous section (Section 2.6) although may compliment each other but contribute many limitations with respect to the above-mentioned issues. These issues are tackled in the work and described below with respect to the contributions in the thesis.

CN-1: Addressing Stuck-at Faults

The first contribution (CN-1) is dedicated to the testing of stuck-at faults in NoC channels. In this work, the issues TI-A to TI-G exhibited by the prior works have been handled. For example, the S-Model [6] detects stuck-at faults only in the interswitch channels. As a result, the scheme partially meets the TI-A and TI-E. These issues are overcome by considering the testing of stuck-at faults in both interswitch and local channels of an NoC. When this test model is extended to include the faults in local

channels, the additional test set is needed. Correspondingly, the scheme raises the issues TI-B and TI-C which are overcome by placing the test modules at the core and router of a node. In the S-Model, every time only one router is selected to test the channel faults. Application of the test method in this way although allows the scheme to be applicable with general NoCs, however, the number of test iterations grows linearly with the network size resulting in the issues TI-D, TI-F, and TI-G. These are overcome by proposing an effective test scheduling technique that allows concurrent execution of an instance of the proposed test algorithm at multiple nodes in an NoC. As a result, the proposed scheduling method does not only lower the number of test iterations satisfactorily on the NoC but also makes the proposed test scheme applicable to all NoCs in general.

CN-2: Addressing Open Faults

An open fault also known as a cut fault is one of the important logic level manufacturing fault that breaks a channel-wire into one or multiple segments. As a result, the connectivity problem arises in the network. It is observed in the literature that no work has considered the testing of open faults in NoC channels. Since the prior works have considered only stuck-at and short faults, it can be inferred that these test schemes fail to detect open faults due to the assumption mentioned earlier. As a result, along with the issues TI-A to TI-G raised in the prior works, they raise another issue namely TI-H. The second contribution (CN-2) is exclusively devoted to the testing of the open fault in NoC channels. The proposed work in this contribution overcomes the issues TI-A to TI-G of the prior works similarly as done in the first contributed work CN-1 and eliminates the issue TI-H on the consideration of channel-open faults. Although, the test scheduling scheme proposed in the CN-1 reduces the overall test time and related performance overhead remarkably than the prior works by lowering the number of test iterations on an NoC, yet, this number can be lowered on the NoC. This limitation and related overheads of the first contributed work CN-1 are also overcome by proposing another new test scheduling scheme.

CN-3: Addressing Short Faults

In the existing previous works, it has been found that the number of test iterations of a test scheme varies with the size and type of the networks. Despite the fact of lower test time per iteration, the overall test time gets higher for the test scheme which in turn implies the issues TI-D, TI-F, and TI-G. These issues are particularly significant with reference to the contributed works CN-1 and CN-2 because the proposed scheduling schemes in these works, though are able to lower the number of test iterations noticeably as seen in prior works they vary with the network size and type. Therefore, the issues TI-D, TI-F, and TI-G due to higher number of test

iterations of the works CN-1 and CN-2 must be overcome in order to improve the efficiency of the proposed test scheme. In this regard, the third contribution (CN-3) is discussed with the testing of short faults in NoC channels. In this work, the issues except TI-D, TI-F, and TI-G are tackled in the way as followed in the contributed works CN-1 and CN-2 with the proposed test algorithm. In order to eliminate the issue of varying number of iterations with respect to network size and type, a new test scheduling scheme is presented. The proposed new scheduling scheme provides same and fixed number of test iterations on the NoCs irrespective of their size and type. This scheduling scheme does not only overcomes the issues TI-D and TI-F but also addresses the issue TI-G.

CN-4: Addressing Short and Transient Faults

In the literature, it is found that many test strategies [7, 40] have considered a permanent fault, such as short fault only in the channels of on-chip networks but do not include the testing of a temporary fault, such as a transient fault. For example, the test algorithm in the $P - 2 \times 2$-Model [40] detects pairwise short faults in the 2×1 neighborhood channels where the test data set is analyzed on the traveling of three channel length. The method does not test a transient fault. Another disadvantage of both prior and contributed works CN-1 to CN-3 is that they do not provide any information regarding the dissipated test energy. The fourth contribution (CN-4) is dedicated to the testing of short and transient faults in NoC channels and presenting a test energy estimation model. The work in this contribution also presents a new test scheduling scheme which makes the proposed solution to be test-time independent of the network size and type. Like the contributed works CN-1 to CN-3, this work overcomes the issues TI-A to TI-G of the previous works. Moreover, this new scheduling technique ensures that the equal amount of test energy is dissipated on the completion of each test round in an NoC. As a result, the issue TI-I of the previous and proposed works is overcome.

CN-5: Addressing Coexistent Short and Stuck-at Faults

In addition to the testing of channel faults in the mesh networks as the conventional networks, there are many unconventional networks, such as the octagon, κ-octagon, spidergon networks [87, 89] which can also meet the performance requirements of network processor SoCs. In the literature, it is found that many testing approaches [37–39] are applicable to mesh network only, i.e., they scale with mesh networks only. This is because of their subnet selection schemes since the channels of a selected subnet undergo the testing in a test round or iteration. For example, every test iteration in the 2×2-Model [39] selects a subset of 2×2 neighborhood from an $M \times N$ mesh network in order to test short and stuck-at faults in the channels of

this neighborhood in the iteration. In this scheme, the test data are analyzed after traveling four channels, i.e., four hops in the neighborhood. Due to this typical 2×2 subnet selection in the test mode, the subnet cannot be applied to the unconventional networks. In addition to the testing issues mentioned above, two issues TI-E and TI-G become more prominent. Due to the multi-hop transmission of the test data, many faults in the neighborhood remain undetected. As a result, the corresponding test method is unable to provide the fault coverage metric up to 100%. The number of undetected faults is increased when multiple faults, such as short and stuck-at faults are coexistent in the same channel. Consequently, the issue of fault non-diagnosability becomes significant and is not addressed in the prior works. The fifth contribution (CN-5) is dedicated to the testing of short and stuck-at faults that are coexistent in NoC channels. The issues TI-A to TI-F of the prior works are overcome by proposing a general test algorithm that detects the coexistent faults on single hop transmission of test packet and exclusively addresses the issue of fault non-diagnosability. The scalability issue TI-G on the traditional and untraditional networks is overcome by proposing a new but a variant of the test scheduling scheme followed in the contributed work CN-4.

2.8 Motivation and Contribution

The system performance in an NoC can be defined by the *aggregate* performances of its building blocks. Also, process scaling is exploited by increasing the number rather than the complexity of these components across the network (NoC). In the meanwhile, the NoC may be exposed to various faults. As scaling continues, concurrent testing of the components in the architecture becomes more complex and problem size becomes increasingly larger as well. Hence, it is intelligent to divide a given problem into sub-problems which correspondingly makes it easier to realize the performance improvements. In this thesis, the on-line testing of channels primarily for manufacturing faults due to their perpetual impediments on the system performance is considered. Additionally, modeling the network resource utilization in terms of energy dissipation during testing a channel and testing of transient faults are included. This fixed problem size is treated as the smaller slice of the overall NoC testing problem.

The motivation of the current thesis is designing a distributed time optimized test solution that can be applied both in the off-line as well as the on-line mode. The proposed solution should have the capability of detecting a channel fault and diagnosing the fault to identify a faulty channel wire. The fault should not be limited to within stuck-at and short faults only but also extended to open fault in the channels. At the same time, the channel testing for these faults should not be confined to the interswitch channels only but also include local channels as well. In addition to the test of these manufacturing faults on data,

control, and handshake wires in a channel and between channels of a node, and analyzing the severe impact of these faults on the network performance in the on-line mode, the proposed test scheme must overcome the limitations by prior schemes or at least reduce them to a satisfactory level. Such demands can be fulfilled by considering a suitable test scheduling that concurrently executes the test algorithm at multiple nodes in an NoC architecture. The thesis also presents a suitable test scheduling scheme. Thus, while the proposed test mechanism addresses the manufacturing faults in NoC channels using small test set, small hardware units, and few clocks per iteration. The proposed test scheduling is targeted to lower the overall test time and subsequent performance overhead and makes the test scheme scalable to large general networks. With each contribution presented in chapters subsequently, the limitations of prior works are overcome as described earlier. The contributions to this thesis are briefly stated as below.

- **First Contribution** – A test algorithm at a node of an NoC is proposed. The proposed algorithm is dedicated to detect and diagnose the stuck-at faults appearing in channels, and report failure modes (packet corruption, packet misrouting, packet dropping) caused due to the faulty channels. Both stuck-at-0 (SA0) and stuck-at-1 (SA1) faults are considered. The BIST structures are designed to implement the test algorithm. Two test vector sets, one contains All-One (A1) and another contains All-Zero (A0) vector, are exercised to test SA0 and SA1 faults, respectively. Two new test scheduling schemes are proposed to lower the overall test time needed by the test solution for channel's stuck-at faults in the NoC. First scheduling scheme is based on the selection of nodes which are located in the diagonal positions. This scheduling scheme scales the proposed test solution with $M \times N$ NoCs, such as mesh, torus. The second scheduling scheme, on the contrary, is based on the graph coloring problem that helps to select the nodes in a test iteration. As the advantage, this scheduling scheme overcomes the limitation of the first scheduling scheme and generalizes the solution with any NoCs in general. The effectiveness of the proposed solution is illustrated with respect to a set of NoCs. On these networks four quality characteristics: silicon hardware area overhead, test time, coverage metrics, and performance metrics: throughput, latency, and energy consumption are used to evaluate the proposed solution. Further, scalability of the proposed solution is shown with respect to network size, higher channel width, and network type.

- **Second Contribution** – The test algorithm which is primarily designed for addressing the stuck-at faults, is now extended to detect and diagnose open faults in NoC channels followed by reporting the failure modes (partial and full packet loss) caused due to these open faults on the channels. The test algorithm either exercises A0 or A1 test set on the basis of fault model used for the open faults. A new test scheduling scheme is presented

in order to lower the test time and scale the current test scheme with general NoCs. This test scheduling scheme is guided by a rule which is in the work, termed as the *4-Corner Principle*. This test solution is evaluated on a set of NoCs with respect to the quality characteristics: silicon hardware area overhead, test time, coverage metrics, and network performance metrics- throughput, latency, and energy consumption. The method scalability is demonstrated with respect to three network parameters: network size, channel width, and network type.

- **Third Contribution** – A cost-effective test algorithm is proposed to address short faults in NoC channels. Both intra-channel and inter-channel short faults are considered in the NoCs. Considering the high wire density, the proposed fault model has assumed the fault group beyond pairwise occurrence of intra-channel short faults. The well known walking one (W1) sequences as the test data are exercised by the test algorithm to detect channel short faults. A cluster-based test scheduling scheme is proposed towards lowering the test time and related performance overhead. This test scheduling scheme provides the fixed number of test regions irrespective of size and type of NoCs. As a result, this scheduling scheme does not only reduce the overall test time to a constant value for the general NoCs but also scales the proposed test solution on these NoCs. The evaluation of the solution is done with respect to common evaluation parameters: silicon hardware area overhead, test time, coverage metrics, and network performance metrics-throughput, latency, and energy consumption. Application of the proposed cluster-set driven test mechanism on a traditional NoC having smaller and larger size, smaller and higher channel width, and belonging to an untraditional NoC category demonstrates the scalable behavior of the test solution with NoC type.

- **Fourth Contribution** – In addition to detection and diagnosis of short faults, the test mechanism includes transient faults in the channels. A parity checking based method is added in the test mechanism in order to detect the transient faults. Another new test scheduling scheme that partitions an NoC-based system into four subnetworks is presented. Therefore, this partition-based test scheduling brings the test solution into a test-time independent solution for NoCs. In the sequence of designing a test algorithm and test scheduling scheme, an estimation model for the network resource utilization in terms of energy dissipation during testing of NoC channels is presented. In addition to providing another test-time independent solution, the proposed partition-based test scheduling ensures (nearly) the same amount of network resource utilization by the test mechanism. The proposed test solution is evaluated with respect to general efficiency measurement elements: silicon hardware area overhead, test time, coverage metrics, and network performance metrics- throughput, latency, and energy consumption. This test solution also scales with all NoCs in general. The scalability property is discussed with

respect to following three related basic characteristics of the networks; these are network size, channel width, and network type.

- **Fifth Contribution** – The short and stuck-at faults which are assumed to coexistent in the NoC channels are tested by proposing a suitable test method. The W1, A1, and A0 are selected as the test data and exercised by the test method to address these faults. The coexistent nature of the faults prevents the test algorithm from the detection of all short and stuck-at faults appearing in the channels. Although, this testing algorithm efficiently detects all these faults when they appear alone in the channels. Therefore, the issue of fault non-diagnosability is exclusively included. More emphasis is first given on the testing of these coexistent short and stuck-at faults in the channels of unconventional NoCs and then the testing is extended to conventional NoCs. A variant of the partition-based test scheduling is proposed and shows constant test time on these NoCs when the test method is applied. The effectiveness of the proposed test scheme is illustrated with an octagon NoC with respect to silicon hardware area overhead, test time, cover metrics, and network performance metrics- throughput, latency, and energy consumption. The scalability property of this test solution is illustrated in reference to network size, channel width, and network type.

2.9 Conclusion

This chapter has discussed the various test approaches for the basic building blocks of an NoC infrastructure. The literature survey on the previous works has been focused on the testing of IP cores, routers, and channels. The detailed survey is emphasized on the channel testing for transient and permanent faults. It is observed that only a fraction of researchers has targeted a subset of manufacturing faults in channels to improve the reliability and yield of the NoC-based systems. This chapter has also indicated the research direction of the thesis on the basis of the several drawbacks of the prior works. The research direction is represented in the form of contributed works. Contributions to the thesis thus begin with the next chapter which is dedicated to addressing the stuck-at faults in channels of an NoC and the underlying test mechanism is executed on activation of diagonal nodes in the NoC.

Chapter 3

Addressing Stuck-at Faults in Channels of Networks-on-Chip

3.1 Introduction

With the constant advancements in manufacturing yield, the Moore's Law has tremendously contributed to the development of dependability design techniques that give sufficient space for growing complexity of many computing systems. Many microprocessor manufacturers are currently migrating to chip multiprocessors (CMPs) and embed in their latest products. As Moore's Law continues to apply, CMPs allow an integration of many IP cores in systems-on-chip (SoCs). Embedded systems as a part of computing systems are then transformed into multiprocessor and many-core SoCs (MPSoCs). But, with the increase in IP cores and long interconnects, SoCs are unable to meet the expectation of a wide range of high-performance computing applications, advanced networking, high-end digital multimedia, and cloud computing. As the demand for high-performance computing and communication has increased since last decade, network-on-chip (NoC) paradigm has emerged as a holistic communication architecture for solving the issues of SoCs on a single die [3, 16, 17, 166].

Current trends have successfully inherited a multicore architecture in applications of energy-efficient NoC-based SoCs. However, aggressive CMOS scaling expedites interconnect and transistor wear out. As a result, the lifespan of MPSoCs becomes shorter. The NoC-based systems can, therefore, experience failures caused by aging to various physical defects as well as hostile attacks caused by malicious third parties. Designing safe and secure systems, in general, has taken into account of the reliability from early design stages to optimize application specific solutions [160, 167]. Use of an NoC architecture as communication infrastructure over integrated multicore SoCs has brought new challenges in terms of testability [16, 168] due to wire density and manufacturing faults, such as stuck-at faults (SAFs) which are experienced in the communication channels (interconnects) of the NoC. A

communication channel in NoCs is realized with a group of metallic wires. The performance benefit earned in adopting an NoC is constrained by these metallic communication channels while affected by faults because the faulty channels quickly become performance impediment and forcibly put the NoC into various system level failure modes. The failure modes may include corruption, misrouting, and dropping of application data packets while transported on faulty channels. Therefore, these faults are of special concern not only in normal mode but also in test mode to improve yield and reliability in NoC-based systems.

The remainder of this chapter is organized as follows. Motivation and various contributions to this work are stated in Section 3.2. Modeling SAFs in channels and various system level failures are described in Section 3.3. Proposed test module and the test algorithm are described in Section 3.4. A new test scheduling is proposed in Section 3.5. Results are provided in Section 3.6. The solution scalability and portability of the proposed scheme are consecutively shown in Sections 3.7 and 3.8. Different benefits arising out of the proposed model over a set of existing approaches are appraised in Section 3.9. Basic limitations of the proposed scheme are discussed in Section 3.10. The chapter concludes with Section 3.11.

3.2 Motivation and Contributions

Nowadays, NoC communication architectures are often used in many MPSoC designs. On the contrary, manufacturing an NoC architecture without any fault is almost impossible. For example, NoC channels are much inclined to logic level stuck-at faults. This trend directs towards a relevant challenge viz. the post-manufacturing test costs associated with basic NoC components (here channels). The test costs may account significant part of the total budget allocated for testing [3]. Indeed, the main reason includes long test time. Additionally, an NoC communication architecture needs to be furnished with a capability of fault detection and diagnosis in parallel so as to observe early reliability hazards. In order to guarantee reliable data transportation in such a system, one must apply avoidance actions by a fault tolerant mechanism [44, 46, 48] that exercises a faulty channel identified by a test method. The literature, however, reveals that most of the fault tolerant mechanisms do not comprise a test method. Therefore, the demand of devising a suitable test algorithm for detecting and diagnosing channel faults is enhanced so as to prevent the NoCs from various system level failures during routing.

When an efficient test mechanism compliments any new design technique, it will then only be adopted. In an NoC architecture, two main requirements must be addressed while a test method is referred. First, how to detect and diagnose faults in NoC communication channels with little hardware area overhead. Second, how to reduce overall test time that incurs low-performance overhead in terms of packet latency and energy consumption. The work in this chapter aims to develop a distributed, low-cost and fast on-line test solution that

addresses SAFs in NoC channels and analyzes their severe effects on network performance. The proposed test algorithm named here "*1-step*" algorithm detects both stuck-at-0 (SA0) and stuck-at-1 (SA1) faults on wires of the interswitch and local channels from a node (router and its core). First, a test module (TM) is implemented in the nodes, that incurs little hardware area overhead on an NoC architecture. Second, the advantage of the inherent parallelism of data transmission in NoC structures is extended to apply the test algorithm simultaneously at multiple nodes. Next, a suitable test scheduling is proposed to guarantee reduced overall test time and makes the proposed solution scalable with a network of arbitrary size, channel width, and topology. Synthesis results show that the TM takes 5–7% hardware area to cover SAFs. Experimental results show that the proposed on-line test method can detect all modeled SAFs in both interswitch and local channels resulting 100% coverage metrics. Simulation results reveal deep insights on the impact of SAFs in NoC performances in order to demonstrate the on-line evaluation of the proposed solution. As compared to prior approaches, the proposed solution reduces 23.84–78.01% hardware area overhead and speeds up to $4 - 16\times$ for a series of networks considered for evaluation. Furthermore, the performance overhead is significantly reduced. The proposed solution improves 13.21–40.40% packet latency degradation and reduces 10.90–46.52% energy consumption.

Thus, multiple contributions to the work presented here are listed below.

C1. Detection and diagnosis of SAFs on wires of interswitch and local channels from a node in an NoC architecture. The detection mechanism ensures the state of faultiness or non-faultiness of a wire in a channel. The diagnosis mechanism on SA0 or SA1 fault identifies a faulty channel-wire. A set of faulty channel-wires thus identified may be exercised by a fault tolerant mechanism to redirect the traffic during transmission.

C2. Test scheduling that selects multiple nodes to simultaneously execute the test algorithm for reducing the test time.

C3. Evaluation of the proposed scheme in terms of various quality characteristics, such as test time, coverage and performance metrics to show the effectiveness of the scheme.

C4. Establish the scalability of the proposed solution irrespective of network size and channel width.

C5. Portability of the proposed solution with different topologies.

C6. Benefits of the proposed solution over a set of prior approaches.

3.3 SAFs and System Level Failures

It is assumed that NoC channels are exposed to SAFs that may put the NoC into various system level failures, even make it useless. In this section, the SAF model used during testing

Table 3.1: Notations used for SAF model in this chapter.

Acronym	Meaning
n	Channel width.
l_k	A channel-wire of single bit width. $1 \leq k \leq n$.
f_{sa0}	An SA0 fault on an l_k.
f_{sa1}	An SA1 fault on an l_k.
x_k	Single/multiple occurrences of an SA0 fault on the l_k.
#SA0	Size of SA0 faults in a channel.
y_k	Single/multiple occurrences of an SA1 fault on the l_k.
#SA1	Size of SA1 faults in a channel.
#SA	Size of SAFs in a channel.
#Ch	Number of unidirectional channels in an NoC.
#SAF	Size of SAFs in an NoC.

of the channels is first presented. Then, various system level failures caused by the SAFs and the respective effects propagated to NoC performances are discussed.

Definition 3.1. ***Stuck-at-0 Fault:*** *It is a manufacturing fault on a channel wire l_k that forces the logic value of the wire to be stuck at logic-0 regardless of the actual logic level of the wire. The fault is labeled as f_{sa0} and its occurrence is termed as an instance x_k.*

Definition 3.2. ***Stuck-at-1 Fault:*** *It is a manufacturing fault on a channel wire l_k that forces the logic value of the wire to be stuck at logic-1 regardless of the actual logic level of the wire. The fault is labeled as f_{sa1} and its occurrence is termed as an instance y_k.*

3.3.1 SAF Model

Devising a test mechanism for NoC channels needs to start from a model that must realistically serve the faults. These faults may be specific to the nature of the channels which act as a highway for data transportation. An SAF whether SA0 or SA1 is a most common manufacturing fault at the logic level. When channel-wires experience these faults, data being transported get modified into logic-0 or logic-1, respectively. As a result, packet misrouting, corruption, and timeout treated as channel errors may be experienced by the network. In order to keep the reliable transmission of data on the network, these faults must be detected and diagnosed prior to execution of a fault-tolerant routing approach that may update its routing table if necessary to avoid faulty channel-wires in a routing path. Any channel can be a part of a routing path between terminal nodes (sender and receiver) in an NoC architecture. Therefore, the testing region is actually spread to all channels in the NoC.

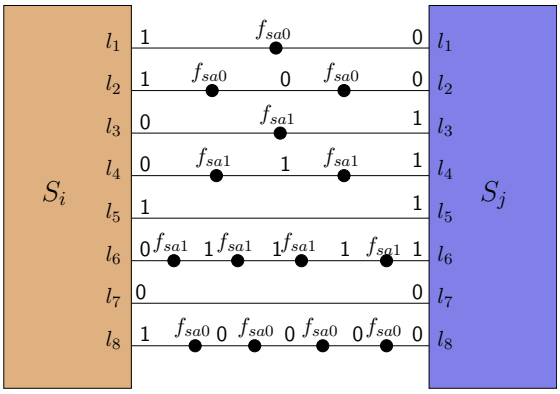


Figure 3.1: Single/multiple existences of stuck-at faults in the channel (S_i, S_j).

Figure 3.1 demonstrates SA0 and SA1 faults on wires of the interswitch channel (S_i, S_j). A channel-wire l_k can experience either SA0 or SA1 fault but not both simultaneously. Further, an SA0 or SA1 fault on the l_k can occur at single or multiple locations. For example, the l_1 and l_8 has experienced single ($x_1 = 1$) and four ($x_8 = 4$) instances of the SA0 fault, respectively. Similarly, l_3, l_4 has experienced single ($y_3 = 1$) and double ($y_4 = 2$) instances of the SA1 fault, respectively. Accordingly, the #SA0, #SA1, and #SA (Table 3.1) on an n-bit channel can be defined in Equations 3.1, 3.2, and 3.3, respectively. Therefore, #SAF in an NoC that has #Ch channels, can be defined in Equation 3.4. It is noted that the effect of single/multiple existences of either of the fault over the l_k is same because one instance supersedes another instance on the channel wire. The condition is known as *fault masking* [169]. The proposed test mechanism, therefore, needs to address both SA0 and SA1 faults, in turn, on a channel.

$$\#SA0 = \sum_{k=1}^{n} l_k \times x_k \qquad (3.1)$$

$$\#SA1 = \sum_{k=1}^{n} l_k \times y_k \qquad (3.2)$$

$$\#SA = \#SA0 + \#SA1 \qquad (3.3)$$

$$\#SAF = \#SA \times \#Ch \qquad (3.4)$$

Application of the proposed test method begins on a 2×2 NoC with unidirectional channel configurations, each has bit-width $n = 16$. Then, the network consists of 256 channel-wires. One may think an *n-bit* channel as a single *n-bit* data bus. In view of the effect of single and multiple occurrences of SA0 or SA1 on a wire, it suffices to assume single existence, i.e., $x_k = y_k = 1$ of these faults on a l_k in turn for analysis. Therefore, the test mechanism will engage itself in testing 256 SA0 and 256 SA1 faults, in turn, i.e., 512 SAFs on the 2×2 network of 16-bit channels.

3.3.2 System Level Failures

The proposed test solution is applied in the on-line mode in which a subset of NoC channels undergoes the testing while rest of the NoC is operational. However, an application packet received at a router that needs to forward the packet on its channel under test must wait at the router till the channel is no longer in the test mode. This decision is taken by the arbiter of the router. Besides, logic level SAFs in channels may have effects on the functional behavior of NoC system resulting in various system-level failures. A packet consequently gets influenced while being transported on a faulty channel. Various system-level failures caused by channel-SAFs are now defined. The system-level failures caused by the SAFs are classified into three main categories- (a) packet corruption, (b) packet misrouting, and (c) packet dropping. Also, the last category is extended to timeout failure.

(a) *Packet Corruption-* This failure mode is assumed when the data and handshake wires of a channel in an NoC suffer from stuck-at faults. Application data in packets (each has multiple flits) on the channel while gets corrupted due to SAFs in data-wires, the router over the channel receives corrupted packets and forwards them to its intended output port if no preventive measurement is taken at the router. Naturally, a destination node receives a corrupted packet even if no channel later in the routing path has an SAF. Note that the packet data may be modified several times depending upon the faulty data-wires on channels appearing in a routing path. The SAFs on handshake-wires modify the "ack" and "val" signals resulting wrong information about the receipt of a packet sent to the sender and received at the receiver, respectively. As the application data are placed only in payload field, the failure mode is realized as the payload error during performance evaluation of the NoC.

(b) *Packet Misrouting-* This failure mode results out of the faulty behavior of a control channel-wire that carries the packet header. An NoC is a packet switched network. Every node communicates with each other by transporting a packet via one or multiple channels in a routing path of the node pair in the communication. When the control wire suffers from an SAF, the packet header may be modified. The router over a channel whose control-wire is affected with an SAF forwards the packet to an unintended destination node. Thus, a packet misrouting occurs. From security viewpoint in some systems, it is unexpected as the information is leaked. This packet misrouting is realized as the misrouting error during performance evaluation of the NoC.

(c) *Packet Dropping-* This failure mode is experienced in a network when the packet trailer is not received by the intended node in a predefined time unit. Several factors, for example, traffic size, channel status, I/O buffer status in a router are responsible for this failure. Besides these factors, an SAF may enhance this failure mode when a control channel-

wire carrying the packet trailer suffers from the fault. Like the header, this trailer gets modified and the intended destination node never receives the end of packet (eop) signal. Thus, the packet is dropped in the network resulting in packet dropping failure mode.

3.4 Proposed Test Model

This section presents a low cost and fast test mechanism that addresses both SA0 and SA1 faults in channels from an NoC node. First, the test infrastructure is discussed, that is mandatory to execute the test algorithm for the SAFs. Next, the 1-step test algorithm is presented to ensure whether the channels of the node are affected by the SAFs.

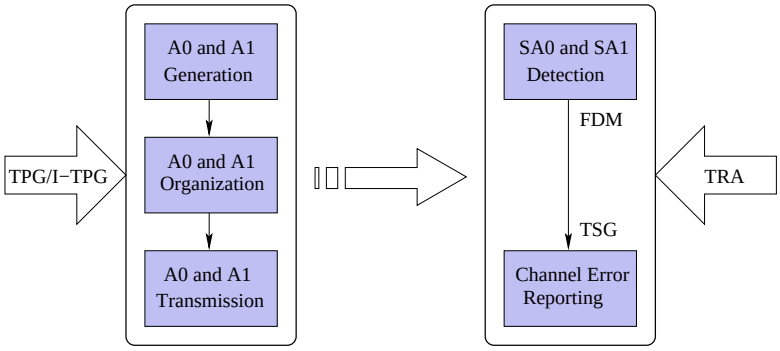


Figure 3.2: A general view of test module for addressing SAFs in channels.

3.4.1 Test Infrastructure for SAFs

The detection and diagnosis of stuck-at faults in channels of an NoC communication architecture are done here by proposing a test algorithm. In order to implement the proposed algorithm in the on-line mode, a pair of test pattern generator (TPG) and test response analyzer (TRA) must be included. Figure 3.2 represents a general view of a TPG and TRA unit. The common functionalities of a TPG unit are to derive test sequences including header and trailer information, organize these raw data into test packets using wormhole switching technique, and transmit them. Two test sequences- All-One (A1) and All-Zero (A0) are sufficient to detect all SA0 and SA1 faults in channels. The two test vectors as flits are enclosed with header and trailer flits into two test packets. Typical test packet organization to address SA0 and SA1 faults is provided in Table 3.2 and 3.3, respectively. The header and trailer flits contain routing and control information like the beginning of packet (bop), end of packet (eop), and so on. The test bits are put in these flits when control wires (CWs) of a channel undergo the testing. The test bits put in the payload flit are mandatory to test the data wires (DWs) of the channel. Beside these CWs and DWs, the channel subsumes handshake wires (HWs) which are tested on considering the test bits in handshake information that carry “ack” and “val” signals of a test packet.

Table 3.2: A test packet organization with A1 sequence for SA0 fault detection in 8-bit data channel.

	Payload Flit (A1)	
H	1	T
e	1	r
a	1	a
d	1	i
e	1	l
e	1	e
	1	r
	1	

Table 3.3: A test packet organization with A0 sequence for SA1 fault detection in 8-bit data channel.

	Payload Flit (A0)	
H	0	T
e	0	r
a	0	a
d	0	i
e	0	l
r	0	e
	0	r
	0	

The TRA unit has two parts. One, fault diagnosis module (FDM) that verifies the test responses (received test sequences) with the local test sequences. The module basically checks the response patterns to diagnose the SAFs on the channels over which they are received. Consequently, the faulty wires are identified. Second, signal generator called TRA's signal generator (TSG) that takes diagnosed results from the FDM. Then, the generator checks and reports various system level failures depending on the type of faulty wires or caused by SAFs. For instance, if DWs/HWs are affected by SA0 (or SA1) faults, the TSG detects payload error which is treated as the packet corruption.

An NoC architecture possesses inherent parallelism where a packet can be transmitted to single or multiple destinations. This feature is exploited to lower the test time and consequent performance hazards. The $< TPG, TRA >$ pair is placed at both core and router of each node and the pair is called a test module (TM). Further, each test packet from a core-TPG has only one destination, i.e., connected router. The core-TPG thus unicasts test packets on the outgoing local channel. On the other hand, a test packet from a router may have multiple destinations. The router-TPG thus multicasts the packet on both outgoing local and interswitch channels to its linked core and neighbor routers. The TPG in a router is integrated with an additional circuit block such that the integrated TPG (I-TPG) can multicast the test packets. One may consider this I-TPG as the multicast wrapper unit (MWU) [3].

3.4.2 Testing SAFs in Channels of a Node

The test of an NoC-based communication architecture is divided into the test of three basic components- channels, routers, and cores. Therefore, the test of local and interswitch channels is one part of the whole NoC architecture. This work exclusively presents a test technique which is designed in view of detecting both SA0 and SA1 faults in channels of the NoC

architecture and diagnosing the faults to report channel errors that as seen later, have the significant effect on the network performance. The proposed test technique accounts these faults on the DWs, CWs, and HWs of channels in the NoCs. The test technique is executed in the on-line mode where a subset of channels is considered under test. Additionally, rest of the underlying network is kept functional. In the functional mode, any node can transmit application packets to any other nodes in the network. However, each incoming application packet must wait at a node in a routing path, which is currently busy in testing its channels. Since the arbiter in the node does not allow it to forward the packet to its output port on the channel in the routing path.

The basic mechanism of the test algorithm for SAFs in channels from a node is to transport test packets from the node and analyze the test responses at a neighbor node. The TMs in sender and receiver of the test packets take control of execution of the algorithm. Both SA0 and SA1 faults are assumed in channels. Two test packets, one containing A1 test vector and another containing A0 test vector are transmitted in turn from the sender. The core-TPG and router-TPG (I-TPG) in sender node generate the test packets. The test vectors for the faults affecting DWs, CWs, and HWs of the channels are placed in the payload, control, and handshake information of the packets. The units generate similar test packets. But, the header of a packet generated by router-TPG differs with that generated by the core-TPG. Because the former packet may have multiple destinations. During transmission, the core-TPG inside the sender node unicasts the packets on the channels to its linked router. At the same time, the router-TPG multicasts the test packets on the channels shared with its linked core and the routers of the neighbor nodes. Each packet sent by core-TPG or router-TPG is capable of detecting either SA0 or SA1 fault on all wires of the channels. Therefore, the wires of the channels are kept filled with a test vector of the packet. One can say, in other words, the subset of channels under test must be filled at the same time while the test packets are sent. The test algorithm proposed here is named as "*1-step*" algorithm since the whole test vector (A0 for SA1 fault or A1 for SA0 fault) is applied on a channel at the single attempt.

$$\bigwedge = \begin{cases} 00111110 \longleftarrow \mathit{Response\,Vector} \\ 11111111 \longleftarrow \mathit{Local\ A1} \\ ---------- \\ 00111110 \longleftarrow \mathit{Count\ 0s} \end{cases} \tag{3.5}$$

$$\bigvee = \begin{cases} 00110100 \longleftarrow \mathit{Response\,Vector} \\ 00000000 \longleftarrow \mathit{Local\ A0} \\ ---------- \\ 00110100 \longleftarrow \mathit{Count\ 1s} \end{cases} \tag{3.6}$$

On receiving the test packets, the TRA at receiver core and routers analyzes the received test vectors treated as responses to detect whether a channel-wire is affected by either an SA0 or SA1 fault. If the wire experiences an SA0 or SA1 fault, the TRA at its input port receives logic-0 or logic-1 on the wire. Thus, the TRA detects an SA0/SA1 fault on the wire. Such detection ensures the state of the faultiness of the channel-wire. Otherwise, received flit over the wire is same as sent. Note that the detection is done by the FDM of the TRA. Detecting a fault on a wire is not enough, but the faulty wires in channels must be identified so that the faulty wires can be exploited by a fault tolerant routing to make reliable communication in the network. In order to identify the faulty channel-wires, the FDM in the TRA extends its functionality and does the pattern checking of the responses with respective local test vectors derived already. The local test vectors may be treated as the "*golden or expected*" test responses. The pattern checking is performed by doing logical $ANDing$ and $ORing$ on the test responses with respective local test vectors. The 0s of the first operation called *zero counts* and the 1s on the second operation called *one counts* represent the faulty wires in a channel affected by SA0 and SA1 faults, respectively. For example, if the I-TPG in the router S_i (Figure 3.1) transmits the test vector "1111 1111" on the channel (S_i, S_j) to test SA0 faults, the response vector received by the TRA of the neighbor router S_j is "0011 1110". Now, the FDM in the TRA does logical *ANDing* on the response vector with the local vector (golden response) "1111 1111" already generated by the I-TPG in the S_j. Equation 3.5 shows the operation that determines the channel-wires l_1, l_2, l_8 (taking 0-counts) are affected by SA0 fault. Similarly, the wires in the channel (S_i, S_j) that are affected by SA1 fault can be explained. For example, the faulty wires can be found (taking 1-counts) in Equation 3.6 and the l_3, l_4, l_6 are those faulty wires that are affected by the SA1 fault.

Table 3.4: The 2-bit TRA signals for the channel errors due to channels stuck-at faults.

Signal bits	Signal Type	Meaning
00	No Fault	Data received over channel is correct
01	Payload Error (PE)	Data get corrupted
10	Misrouted Error (ME)	Data received at unintended node
11	Timeout Error (TE)	Data get dropped

Now, the FDM supplies the results obtained from the logical operation to the connected TSG unit. On the basis of 0-/1-counts and the type of faulty wires whether belonging to DWs, CWs, or HWs, the TSG unit reports channel errors. It is presumed that there are 3-types of channel errors which are characterized by payload, misrouting, and dropping errors. Each error is reported by the TSG unit using a 2-bit signal. The meaning of each 2-bit signal is referred in Table 3.4. The SAFs affecting the DWs result in payload error (PE). Application data in parts are placed in terms of payload flits. If the PE is suspected on a channel, these

flits are corrupted. At the system level, the error is realized as packet corruption. Every packet is enclosed with header and trailer flits. The flits are relayed on the CWs. The SAFs may then affect these flits. A CW which carries the packet header, being affected by an SAF, brings the network into packet misrouting mode which is another system-level failure. As the packet header is modified, a router over the affected CW updates its routing table and forwards the packet to an unwanted destination. On the other hand, if the faulty CW carries the packet trailer, the network is put into third system level failure known as packet dropping mode. In this mode, the router over the faulty CW drops the packet instead of forwarding to its output port. Like another type of channel-wires, HWs may be affected by an SAF resulting in corrupted "ack" and "val" signal bits for the packet. This channel error due to faulty HWs is added into the PE that enhances the packet corruption.

Above all, the channels may undergo another error called packet deadlock. But, the advantage of the proposed test mechanism is that no packet deadlock occurs during execution of the algorithm. It becomes possible since the algorithm exploits two small size test packets which are in addition analyzed over single hop (one channel length) transmission, e.g., a test packet sent from the router S_i is transmitted on the channel (S_i, S_j), shown in Figure 3.1, is analyzed in the router S_j. Additionally, a network during transmission of application data may be intruded with a natural error called packet timeout. The error happens while there is high traffic in a channel or an intended receiver never receives the trailer flit ("eop" signal) of a packet within a predefined time period. No doubt, the packet is lost. It does not matter for the error whether a channel in a routing path is affected with an SAF. The packet timeout is enhanced while the CW of the channel carrying the trailer flit in the routing path is influenced by an SAF. Thus, the packet dropping can be realized along with the packet timeout.

3.5 Test Scheduling

In the proposed test mechanism, a TPG/I-TPG module propagates the test packets and its neighbor TRA module analyzes the responses. The approach entirely excludes the need for a TAM for channel faults. Another distinct advantage of the approach is the at-speed delivery of the test packets on an NoC independent of its size, channel width, and type. One major challenge of testing NoC channel faults is the test time that incurs the test cost of a fault-free design. The test time depends on the number of test rounds and/or iterations. With the increase in the number of test rounds/iterations, the overall test time is increased resulting in performance degradation in terms of packet latency and energy consumption. Therefore, a trade-off between the number of test rounds/iterations and performance overhead must be tackled. As mentioned earlier, an NoC-based communication architecture allows parallelism where multiple nodes use the test mechanism at the same time. Thus, an important preprocessing task that determines the test rounds/iterations is defined as test scheduling.

3.5.1 Finding Test Rounds and Iterations

The NoC nodes must be selected in such a way that the test scheduling problem should satisfy a set of constraints, e.g., test time required to complete post-manufacturing test of SAFs in NoC channels. Many research efforts have tried to lower the total test time via minimizing the number of test rounds/iterations in a network. It is seen that the number is comparatively high and network specific. At this point, the test scheduling problem is interestingly formulated as (1) matrix diagonal problem on an $M \times N$ network, and (2) graph coloring problem [170] on a general network. An NoC is an interconnection of the routers and IP cores via channels. Analysis of a system becomes easier on considering its graphical modeling. An interconnection of basic components of an NoC architecture has resulted in a topology which can be modeled as a graph $\mathbb{G} = \{\aleph, \complement\}$. The $\aleph = \{\aleph_i | 1 \leq i \leq \mathbb{N}\}$ is a set of NoC nodes. Each node $\aleph_i$ is a pair of router S_i and its dedicated core C_i. A set of NoC channels is denoted with $\complement = \{E_k | 1 \leq k \leq \mathbb{N}\}$. Each channel E_k whether local or interswitch occupies *n-bit* length, where each bit corresponds to a channel-wire. Note that first 2-bits and last 2-bits in the *n-bit* channel are exercised here as CWs and HWs, respectively. The rest of the bits is treated as DWs.

Table 3.5: Matrix representations of a 4×4 mesh/torus NoC.

$\aleph_1$	$\aleph_2$	$\aleph_3$	$\aleph_4$
$\aleph_5$	$\aleph_6$	$\aleph_7$	$\aleph_8$
$\aleph_9$	$\aleph_{10}$	$\aleph_{11}$	$\aleph_{12}$
$\aleph_{13}$	$\aleph_{14}$	$\aleph_{15}$	$\aleph_{16}$

Table 3.6: Matrix representation of an octagon NoC.

$\aleph_1$	$\aleph_2$
$\aleph_3$	$\aleph_4$
$\aleph_5$	$\aleph_6$
$\aleph_7$	$\aleph_8$

Table 3.7: Matrix representation of a hybrid NoC.

$\aleph_1$	$\aleph_2$	$\aleph_3$	$\aleph_4$
$\aleph_5$	$\aleph_6$	$\aleph_7$	$\aleph_8$
$\aleph_9$	$\aleph_{10}$	$\aleph_{11}$	$\aleph_{12}$
$\aleph_{13}$	$\aleph_{14}$	$\aleph_{15}$	–

Table 3.8: Location of nodes with respect to Table 3.5.

00	01	02	03
10	11	12	13
20	21	22	23
30	31	32	33

Table 3.9: Location of nodes with respect to Table 3.6.

00	01
10	11
20	21
30	31

Table 3.10: Location of nodes with respect to Table 3.7.

00	01	02	03
10	11	12	13
20	21	22	23
30	31	32	–

3.5.1.1 Matrix Diagonal Problem

The problem of determining the test rounds (TRs) and test iterations (Tits) in a test round in an NoC is mapped to the problem of finding diagonals in a matrix $\Im$ representation of the network. Any graphical representation $\mathbb{G}$ of an $M \times N$ NoC can be modeled to the matrix $\Im_{r \times c}; 1 \leq r \leq M, 1 \leq c \leq N$. For example, Tables 3.5, 3.6, and 3.7 present matrix representation of mesh/torus, octagon, and hybrid NoCs, respectively [171]. The diagonal and

symmetric properties of a matrix computation can help in solving the problem. Every element χ_{rc} in the $\Im$ is a node $\aleph_i$ and is placed in row major order. Then, a node is considered to be on a diagonal level D if the end elements of the diagonal are the elements χ_{rc} and χ_{cr}. The $\aleph_1, \aleph_2$ in Table 3.5 are therefore treated as odd and even nodes, respectively. Another way to find a node $\aleph_i$ as the odd or even node uses the matrix cell locations $\Im_{r\times c}$ instead of matrix elements (nodes). If the addition of the digits used for cell location of an $\aleph_i$ returns to an odd/even number, the $\aleph_i$ is termed as odd/even node. For example, cells "00=0+0=0" and "01=0+1=1" in Table 3.8 correspond nodes $\aleph_1, \aleph_2$ in Table 3.5 as even and odd nodes, respectively. Thus, routers S_1, S_2 and their dedicated cores C_1, C_2 are similarly said as even/odd routers and cores, respectively. Similarly, cells in Table 3.9 and 3.10 correspond to in Table 3.6 and 3.10, respectively.

Each diagonal D is labeled as odd $(D_\gamma; \gamma = 1, 3, ..)$ and even $(D_\gamma; \gamma = 2, 4, ..)$ alternatively. Thus, two sets of diagonals D_{odd}, D_{even} are available in the $\Im$. These sets correspond to test rounds (Equation 3.7). Subsequently, initiating a test execution on the nodes on a diagonal level D is treated as a test iteration. A test round whether odd/even is completed after sequencing a test execution on odd/even diagonal nodes. Thus, Equation 3.10 defines the number of test iterations on the $M \times N$ NoC.

$$|TR| = 2 \tag{3.7}$$

$$D_{odd} = \bigcup_{\gamma \in (2\mathbb{N}+1)} D_\gamma \tag{3.8}$$

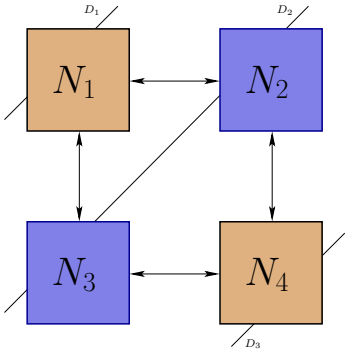
$$|D_{odd}| = \begin{cases} M, \; if \; M \geq N \\ N, \; if \; N \geq M \end{cases} \tag{3.11}$$

$$D_{even} = \bigcup_{\gamma \in (2\mathbb{N}+2)} D_\gamma \tag{3.9}$$

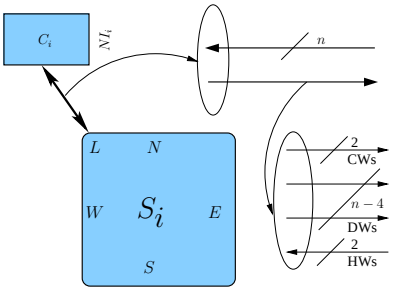
$$|D_{even}| = \begin{cases} N, \; if \; M \geq N \\ M, \; if \; N \geq M \end{cases} \tag{3.12}$$

$$|Tits| = |D_{odd}| + |D_{even}| \tag{3.10}$$

It is evident that the $M \times N$ network results to $M + N - 1$ diagonals as provided in Equations 3.11, 3.12. The proposed test solution for stuck-at faults in channels is illustrated by introducing a basic example on a 2×2 mesh NoC. Figure 3.3 shows a high-level representation of the network where each node $\aleph_i =< S_i, C_i >; 1 \leq i \leq 4$. The $\Im_{2\times 2}$ shown in Table 3.11 corresponds the matrix modeling of the 2×2 mesh network. As per the discussion, nodes $\aleph_1, \aleph_4$ are on the odd diagonal levels D_1, D_3 while the nodes $\aleph_2, \aleph_3$ are on the same even diagonal level D_2. The testing of channels of the $M \times N$ network can, therefore, be completed in just two test rounds D_{odd}, D_{even}. In first test round (TR=I), the nodes at odd diagonal levels D_1, D_3 and the nodes at even diagonal level D_2 in the second round (TR=II) are scheduled to complete testing of the channels.



(a) Abstract representation of a 2×2 NoC.



(b) Abstract view of an NoC node $\aleph_i$.

Table 3.11: Modeling of a 2×2 mesh NoC.

$\aleph_1$	$\aleph_2$
$\aleph_3$	$\aleph_4$

Figure 3.3: High level view of a 2×2 NoC architecture and its node.

3.5.1.2 Graph Color Problem

The matrix-based approach for finding the number of test rounds and iterations may not be accepted by a section of users on the networks other than the $M \times N$ networks. In that case, the problem on a general network can be solved using the graph coloring approach. In favor of searching for an optimal test scheduling, the nodes $\in \mathbb{G}$ that act as the source of test packet injection may be colored. After coloring through all the nodes, the job is to determine the distinct colors chosen. The number of distinct colors $|\Re|$ for the nodes is actually treated as the number of test rounds $|TR|$ (Equation 3.13). Subsequently, the same colored nodes at a level that execute the test mechanism in parallel are treated as a test iteration. Sequencing the test execution at a level for a color $\alpha \in \Re$, a set of test iterations β_α (Equation 3.14) for the α is derived. After sequencing through all the test rounds, the number of test iterations $|Tits|$ in an NoC system is found by the Equation 3.15. Thus, the test execution after the $|Tits|$ guarantees the completion of the undergoing test of SAFs in NoC channels.

$$|TR| = |\Re| \tag{3.13}$$

$$\beta_\alpha = \bigcup_{\upsilon=1,2,..} Tit_\upsilon \tag{3.14}$$

$$|Tits| = \sum_{\alpha=1,2,..}^{|\Re|} |\beta_\alpha| \tag{3.15}$$

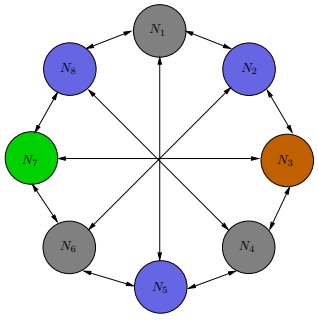


Figure 3.4: Abstract representation of an octagon NoC.

The proposed test approach is introduced with an octagon NoC (Figure 3.4) to illustrate how the TRs and Tits in a TR are computed. The nodes in TR=I are colored with black color. In this round, test execution at Tit=1 is initiated by the node N_1, while next time (Tit=2) execution is simultaneously done on nodes $\aleph_4, \aleph_6$. So, $|\beta_{\alpha=black}| = 2$. Similarly, other

nodes are colored followed by test execution on these nodes. It shows that the channels in an octagon network can be tested in four rounds and six iterations. Furthermore, it is noticed that 2-color problem can also be referred to find TRs and Tits on the $M \times N$ networks.

3.5.2 Determination of Test Time

The proposed test scheduling is applied in order to put a set of channels under test and evaluate the test time needed to complete testing of the SAFs on the NoC communication channels. The test time needed for a channel T_{ch} (Equation 3.16) is directly related to four different components. The first one is the time needed to generate the test set including header and trailer. It is called as test generation time T_{gen}. The second component is the time needed to organize the test set into a test packet. It is called as test organization time T_{org}. Next component of the T_{ch} is the time needed to transport a test packet from a source to a destination where an analysis of the received test set is done. It is called as packet transport latency T_{tpt}. The last component is the time needed to analyze the received test set. This time is called test analysis time T_{tra}.

$$T_{ch} = T_{gen} + T_{org} + T_{tpt} + T_{tra} \tag{3.16}$$

$$T_{n/w} = T_{it} \times Tits \tag{3.17}$$

The T_{ch} can effectively be reduced in two ways by sending the test packets on (1) the shortest path, and (2) non-overlapping paths between sender and receiver nodes. The first one is achieved by limiting the shortest path to the single hop. The second one is managed by two transmission modes- unicast and multicast. The core-TPG in a node unicasts test packets on the outgoing local channel while the router-TPG multicasts the test packets on both outgoing local and interswitch channels. Therefore, the test algorithm tests these outgoing channels of the node in parallel. Thus, the test time T_{ch} needed for a channel is same for all the outgoing channels of the node. The proposed test model is applied in the on-line mode where a subset of channels is kept in the test mode. The goal of minimizing the test time can further be furnished. The proposed test scheduling shows the feasible way where multiple nodes in a test iteration must involve in testing the channels from these nodes. It shows that the time needed for an iteration T_{it} is same as the T_{ch}. The test iterations govern the $T_{n/w}$, the time needed to complete testing of the channels on a network. After sequencing the iterations, the $T_{n/w}$ can be defined in Equation 3.17.

3.6 Experimental Results

In order to demonstrate the effectiveness of the proposed on-line test mechanism in detecting and diagnosing stuck-at faults in channels, it is applied to a set of various NoC communication

Table 3.12: Characteristics of $M \times N$ NoCs.

N/w Size	#R	#C	#Ch	#W	#R-R	#R-C
2×2	4	4	16	256	8	8
3×3	9	9	42	672	24	18
4×4	16	16	80	1280	48	32
5×5	25	25	130	2080	80	50
6×6	36	36	192	3072	120	72
7×7	49	49	266	4256	168	98
8×8	64	64	352	5632	224	128
9×9	81	81	450	7200	288	162
10×10	100	100	560	8960	360	200
11×11	121	121	682	10912	440	242
12×12	144	144	816	13056	528	288
13×13	169	169	962	15392	624	338
14×14	196	196	1120	17920	728	392
15×15	225	225	1290	20640	840	450

architectures. In current as well as next sections, many $M \times N$ NoCs are considered to illustrate the effectiveness including the scalable behavior of the proposed solution with respect to network size and channel-width. Further, the proposed solution is described with an octagon network in Section 3.8 where the demand for portability of the solution is fulfilled. Characteristics of the $M \times N$ networks are described in Table 3.12. The characteristic parameters include the number of routers #R, cores #C, channels #Ch that sums up the number of interswitch channels #R-R and local channels #R-C, and communication wires #W. Here, the bit-width of a channel is set to *n=16-bit*. As mentioned (refer Section 3.3), the effect of multiple instances $x_k \geq 2$ (or $y_k \geq 2$) of an SA0 (or SA1) fault on a channel-wire l_k is same as the single instance of the fault. Therefore, the #SA0 and #SA1 as provided in Figure 3.5 must be injected in turn on these networks before executing the test mechanism. The proposed test method has targeted the post-manufacturing SAFs in NoC channels, where the objective is to detect and locate an SAF on a channel-wire. So, the efficiency of this test method is evaluated in terms of various quality metrics- silicon-area overhead, test time, coverage metrics, and performance metrics.

3.6.1 Test Area Overhead

The area-overhead of the TM architecture in a node is one of the important quality metrics for a test method. The unicast and multicast test set based TM architecture at a node drives the test method. The TM architecture derives, organizes, delivers, and analyses two test

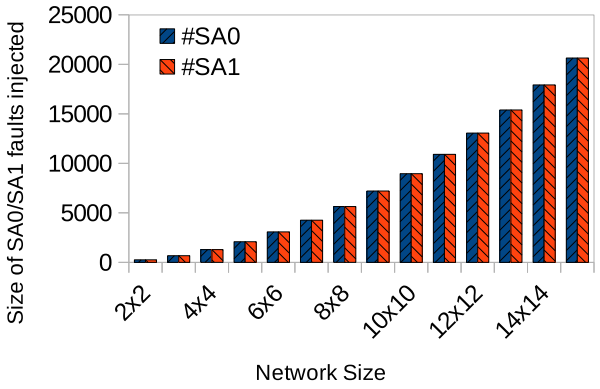


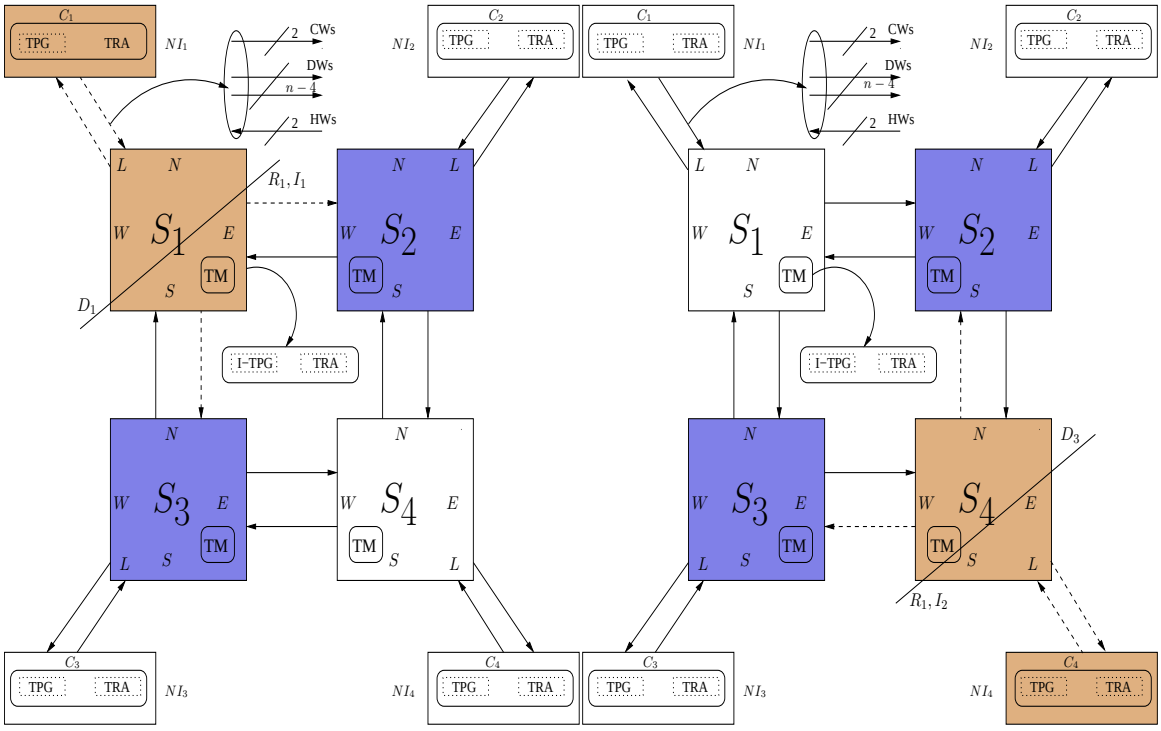
Figure 3.5: Fault injection campaign in 16-bit NoCs.

sets, one contains A1 and another contains A0 to tackle SA0 and SA1 faults, respectively. Two major components- TPG (I-TPG) and TRA contribute to the area-overhead of the TM architecture. Here, two common routers RaSoC [65] and Xpipe [66] are selected to see the area taken by the TM block in these routers. Since the test set size is dependent on n, the area taken by the block varies. The TM block is implemented at a node, i.e., at both core and router of the node. Now, the core-TPG unicasts while the router-TPG (TPG-R), i.e., I-TPG multicasts the test packets. Similarly, the core-TRA analyzes the response from its dedicated router while the router-TRA (TRA-R) does the job on the responses from its neighbors' core/routers. Thus, the area of a TM at a core is generally smaller than that of a router. Table 3.13 provides the synthesis results treated as area overhead (in terms of gate count (GC) needed to implement the TM architecture) for a 16-bit channel. The TM block is synthesized using the Xilinx 10.1 ISE Design Suite on a computer system with 2.60 GHz AMD Phenom(tm) II X3 710 Processor, 4 GB RAM, and 64-bit Widows 7 OS.

Table 3.13: Area overhead of a TM for 16-bit channel.

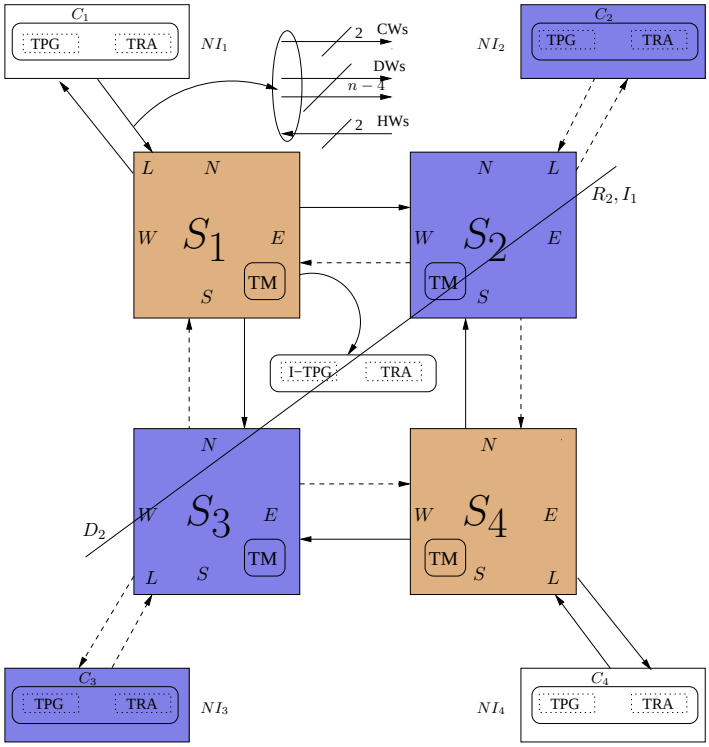
TM Unit	#GC	RASoC(%)	Xpipe(%)
TPG	45	2.67	2.95
TRA	32	1.9	2.1
Total	77	4.57	5.05
TPG-R	70	4.15	4.59
TRA-R	50	2.96	3.28
Total	120	7.11	7.87

In the implementation of the proposed test mechanism, a subset of channels under test in an iteration is selected as per the diagonal location of the corresponding nodes. During



(a) TR=I, Tit=I (R_1, I_1) at the D_1.

(b) TR=I, Tit=II (R_1, I_2) at the D_3.



(c) TR=II, Tit=I (R_2, I_1) at the D_2.

Figure 3.6: Application of the proposed test solution at different test rounds and iterations on a 2×2 mesh NoC with *n-bit* unidirectional channel configuration.

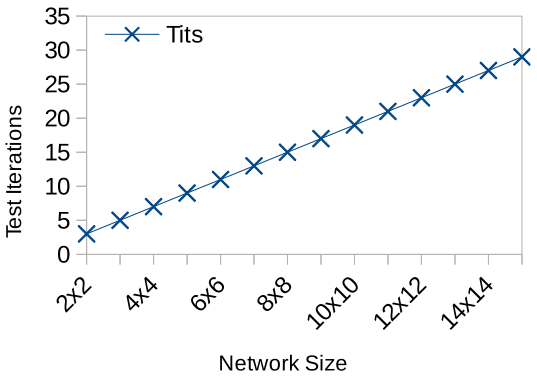


Figure 3.7: Number of test iterations vs. NoC Size.

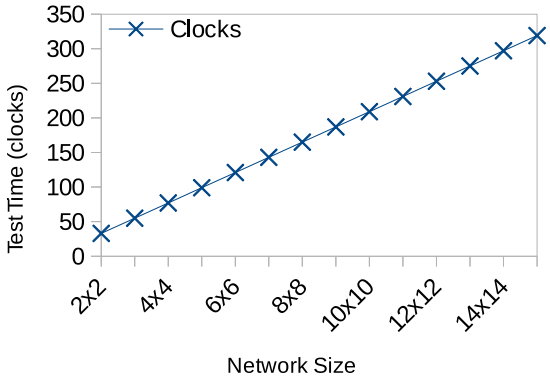


Figure 3.8: Amount of test time vs. NoC Size.

the implementation, an SA1 fault on a channel-wire is treated as a short to VDD while an SA0 fault on the wire is treated as a short to Ground. At one side, TPGs and I-TPGs are placed to derive, organize, and deliver the test sets. On the other side, corresponding TRAs are placed to analyze the responses. All the channels of a network are tested in two test rounds (TRs). However, each round is completed after finite iterations. The proposed test mechanism is illustrated on the 2×2 NoC. Figure 3.6 demonstrates the execution of the test algorithm on the 2×2 mesh NoC with a *n=16-bit* channel. The first round (R_1) test iterations I_1, I_2 are executed as shown in Figures 3.6a and 3.6b, respectively at the D_1, D_3. In R_1, I_1, the node $< S_1, C_1 >$ executes the test algorithm to detect SAFs on interswitch channels $(S_1, S_2), (S_1, S_3)$ and local channels $(S_1, C_1), (C_1, S_1)$. The responses are analyzed by the neighbor's TRA. Once the testing of these channels is done, the test iteration I_2 in the same round R_1 is shifted to the node $< S_4, C_4 >$ which initiates the test algorithm to address SAFs in its outgoing channels. It is seen that a subset of channels of the underlying network has been tested. In order to cover faults on rest of the channels, one must execute the second round R_2 test application as shown in Figure 3.6c. The round is executed for single iteration I_1 at the D_2. The nodes $< S_2, C_2 >, < S_3, C_3 >$ initiate the test algorithm. Thus, all channels in the underlying network are tested by three iterations. Figure 3.7 subsequently gives an idea of the Tits based on the Equations 3.10 and 3.15 for the selected networks.

3.6.2 Test Clock Cycles

The second important quality metric for the test mechanism is the test time (clock) required to address SAFs on the channels under test in an iteration. For 16-bit channels with specified packet format in the test environment, Figure 3.8 tells the amount of test time needed for the channels in the networks. During experimentation, it is observed that the TPG takes 1 clock for generation of A1/A0, header and trailer flit. Another clock is used to organize

Table 3.14: LCM(%) achieved on detection of stuck-at fault in channels of $M \times N$ NoCs.

TR	N/W Size	Tit=1	Tit=2	Tit=3	Tit=4	Tit=5	Tit=6	Tit=7	Tit=8	
	2 × 2	25	25	-	-	-	-	-	-	
	3 × 3	9.52	33.33	9.52	-	-	-	-	-	
	4 × 4	5	20	20	5	-	-	-	-	
	5 × 5	3.08	12.31	20	12.31	3.08	-	-	-	
	6 × 6	2.08	8.33	14.58	14.58	8.33	2.08	-	-	
	7 × 7	1.5	6.02	10.53	14.29	10.53	6.02	1.5	-	
	8 × 8	1.14	4.55	7.95	11.36	11.36	7.95	4.55	1.14	
	9 × 9	0.89	3.56	6.22	8.89	11.11	8.89	6.22	3.56	
	10 × 10	0.71	2.86	5	7.14	8.93	9.29	7.14	5	
	11 × 11	0.59	2.35	4.11	5.87	7.33	9.38	7.62	5.87	
	12 × 12	0.49	1.96	3.43	4.9	6.13	7.84	7.84	6.37	
	13 × 13	0.42	1.66	2.91	4.16	5.2	6.65	7.9	6.65	
	14 × 14	0.36	1.43	2.5	3.57	4.46	5.71	6.79	6.79	
	15 × 15	0.31	1.24	2.17	3.1	3.88	4.96	5.89	6.82	
R1		Tit=9	Tit=10	Tit=11	Tit=12	Tit=13	Tit=14	Tit=15	CLCM@R1(%)	
	2 × 2	-	-	-	-	-	-	-	50	
	3 × 3	-	-	-	-	-	-	-	52.38	
	4 × 4	-	-	-	-	-	-	-	50	
	5 × 5	-	-	-	-	-	-	-	50.77	
	6 × 6	-	-	-	-	-	-	-	50	
	7 × 7	-	-	-	-	-	-	-	50.38	
	8 × 8	-	-	-	-	-	-	-	50	
	9 × 9	0.89	-	-	-	-	-	-	50.22	
	10 × 10	2.86	0.71	-	-	-	-	-	49.64	
	11 × 11	4.11	2.35	0.59	-	-	-	-	50.15	
	12 × 12	4.9	3.43	1.96	0.49	-	-	-	49.75	
	13 × 13	5.41	4.16	2.91	1.66	0.42	-	-	50.1	
	14 × 14	5.71	4.64	3.57	2.5	1.43	0.36	-	49.82	
	15 × 15	5.89	4.96	4.03	3.1	2.17	1.24	0.31	50.08	
		Tit=1	Tit=2	Tit=3	Tit=4	Tit=5	Tit=6	Tit=7	Tit=8	
	2 × 2	50	-	-	-	-	-	-	-	
	3 × 3	23.81	23.81	-	-	-	-	-	-	
	4 × 4	12.5	25	12.5	-	-	-	-	-	
	5 × 5	7.69	16.92	16.92	7.69	-	-	-	-	
	6 × 6	5.21	11.46	16.67	11.46	5.21	-	-	-	
	7 × 7	3.76	8.27	12.78	12.78	8.27	3.76	-	-	
	8 × 8	2.84	6.25	9.66	12.5	9.66	6.25	2.84	-	
	9 × 9	2.22	4.89	7.56	10.22	10.22	7.56	4.89	2.22	
	10 × 10	1.79	3.93	6.07	8.21	10.36	8.21	6.07	3.93	
	11 × 11	1.47	3.23	4.99	6.74	8.5	8.5	6.74	4.99	
	12 × 12	1.23	2.7	4.17	5.64	7.11	8.58	7.11	5.64	
	13 × 13	1.04	2.29	3.53	4.78	6.03	7.28	7.28	6.03	
	14 × 14	0.89	1.96	3.04	4.11	5.18	6.25	7.32	6.25	
	15 × 15	0.78	1.71	2.64	3.57	4.5	5.43	6.36	6.36	
R2		Tit=9	Tit=10	Tit=11	Tit=12	Tit=13	Tit=14	Tit=15	CLCM@R2(%)	CLCM(%)
	2 × 2	-	-	-	-	-	-	-	50	100
	3 × 3	-	-	-	-	-	-	-	47.62	100
	4 × 4	-	-	-	-	-	-	-	50	100
	5 × 5	-	-	-	-	-	-	-	49.23	100
	6 × 6	-	-	-	-	-	-	-	50	100
	7 × 7	-	-	-	-	-	-	-	49.62	100
	8 × 8	-	-	-	-	-	-	-	50	100
	9 × 9	-	-	-	-	-	-	-	49.78	100
	10 × 10	1.79	-	-	-	-	-	-	50.36	100
	11 × 11	3.23	1.47	-	-	-	-	-	49.85	100
	12 × 12	4.17	2.7	1.23	-	-	-	-	50.25	100
	13 × 13	4.78	3.53	2.29	1.04	-	-	-	49.9	100
	14 × 14	5.18	4.11	3.04	1.96	0.89	-	-	50.18	100
	15 × 15	5.43	4.5	3.57	2.64	1.71	0.78	-	49.92	100

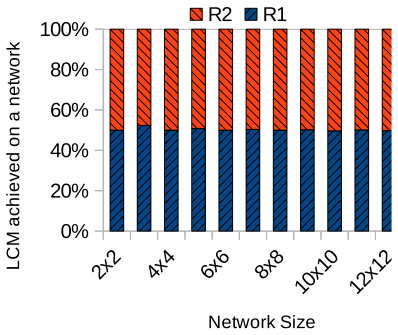


Figure 3.9: Cumulative LCM vs. NoC Size.

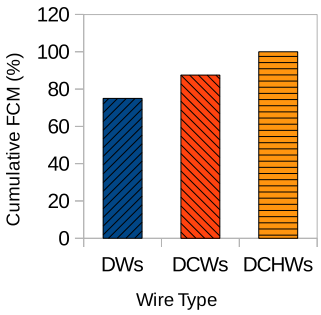


Figure 3.10: Cumulative FCM in networks with 16-bit channels.

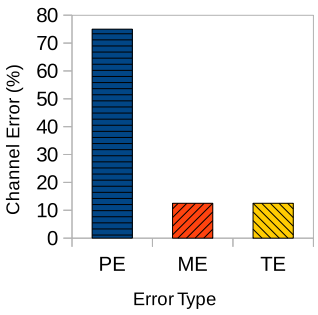


Figure 3.11: Expected channel error in networks with 16-bit channels.

A1/A0 as a test packet. Let us consider that first the test packet containing A1 is delivered. Since a channel under test carries the test packet, it takes 3 clocks (1 flit/clock) to reach the neighbor TRA. During transmission of the test packet with the A1, next packet with the A0 is generated by sender TPG side. On receiving the A1, it is put under analysis at receiver TRAs. As said earlier, a TRA has two modules- FDM and TSG. At one clock, the FDM does the pattern checking while in the next clock, the TSG announces the channel error, if any. Similarly, the received A0 is analyzed. Thus, a channel can be tested in just 11 clocks or in other words an iteration is completed at the cost of 11 clocks.

3.6.3 Link and Fault Coverage Metrics

Next quality metric of the proposed test scheme is the coverage metric. This metric is divided into two categories. One is referred as link or test coverage metric (LCM) while another is called fault coverage metric (FCM). A subset of channels in an iteration is tested. This fraction defines the link coverage metric (LCM). For example, four channels out of 16 channels from the node $N_1 =< S_1, C_1 >$ in R_1, I_1 are placed under test (Figure 3.6a). The LCM achieved is 25%. Table 3.14 provides iteration-wise LCM. Sequencing the test iterations R_1, I_2 (Figure 3.6b) and R_2, I_1(Figure 3.6c) all channels have been successfully tested. Figure 3.9 ensures the LCM achieved at R_1 and R_2, respectively. For example, after executing the test mechanism at the R_1 on a 2×2 mesh NoC, 50% channels have been tested. Rest of the channels are tested at R_2 resulting in the next 50% of the LCM. Thus, the proposed test model has achieved 100% LCM. The single instance of an SA0 and SA1 fault is assumed on wires of the 16-bit channels in which first two bits are treated as control-bits for CWs, next 12-bits as data-bits for DWs, and last two bits as handshake-bits for HWs. The channels for SAFs are put in the sequence of data, control, and handshake wires for testing. Consideration of the data wires (DWs) under test, 75% of total faults can be covered. On extending the test mechanism to control wires (CWs), another 12.5% faults are covered. Further extension of the mechanism to handshake

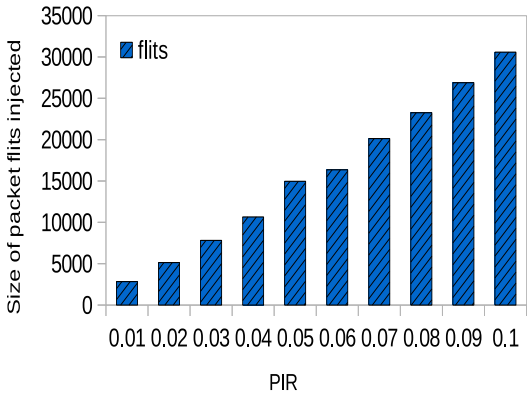
Table 3.15: Size of SA0 or SA1 detected in networks of 16-bit channels.

Network	R_1			R_2			Detected		
Size	#DWs	#CWs	#HWs	#DWs	#CWs	#HWs	#SA0	#SA1	#SAF
2×2	96	16	16	96	16	16	256	256	512
3×3	264	44	44	240	40	40	672	672	1344
4×4	480	80	80	480	80	80	1280	1280	2560
5×5	792	132	132	768	128	128	2080	2080	4160
6×6	1152	192	192	1152	192	192	3072	3072	6144
7×7	1608	268	268	1584	264	264	4256	4256	8512
8×8	2112	352	352	2112	352	352	5632	5632	11264
9×9	2712	452	452	2688	448	448	7200	7200	14400
10×10	3336	556	556	3384	564	564	8960	8960	17920
11×11	4104	684	684	4080	680	680	10912	10912	21824
12×12	4872	812	812	4920	820	820	13056	13056	26112
13×13	5784	964	964	5760	960	960	15392	15392	30784
14×14	6696	1116	1116	6744	1124	1124	17920	17920	35840
15×15	7752	1292	1292	7728	1288	1288	20640	20640	41280

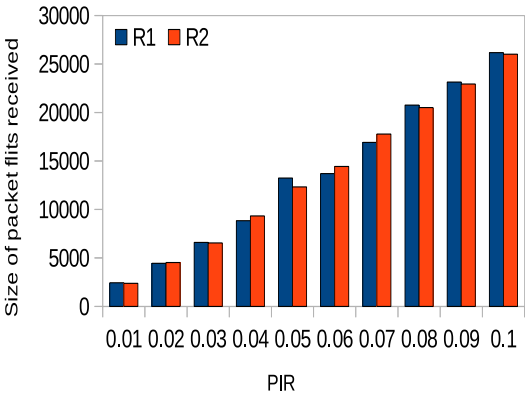
wires (HWs), rest 12.5% faults are covered. Thus, like LCM, the proposed test mechanism achieves 100% FCM and is shown in Figure 3.10. As the faults are addressed iteratively in a test round, Table 3.15 shows the size of faults detected in a test round depending on the channel-wire type. Once an SAF is detected in a channel, the fault is realized in terms of errors on the channel. The payload error (PE), misrouting error (ME), and timeout error (TE) are caused due to SAFs on data and handshake wires, control-wire carrying packet header, and control-wire carrying packet trailer, respectively. Figure 3.11 can be used to observe a channel error. For example, if all DWs are found to be faulty, then the PE is 75%.

3.6.4 Network Performance Metrics

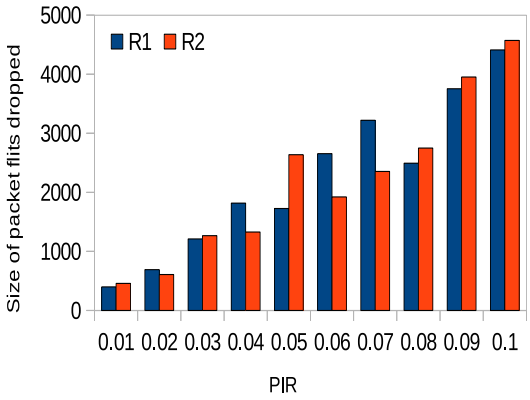
The final quality metric is the network performance that demonstrates the behavior of the network in the on-line testing of its channels. This behavior is analyzed in terms of the effect of the SAFs in channels. The stuck-at faults in channels put an NoC into various system level failures resulting in a significant impact on network performance metrics for instance throughput, latency, and power consumption. To evaluate the network performance under the test model, a cycle-accurate simulator [22] is used and the simulation is carried out on the computer system with 64-bit Ubuntu-14.04 OS. Simulations are done at both rounds, in turn, at the packet injection rate (PIR) of 0.01-0.1. The simulation at every PIR value in a round is run for 10000 cycles including 10% warm-up cycles. Each time large traffic in terms



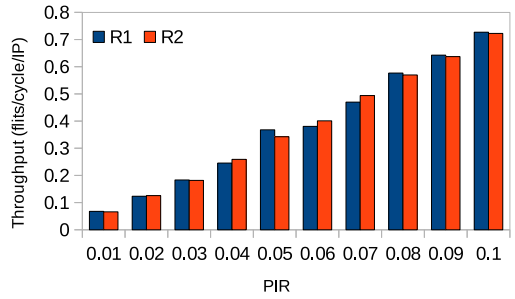
(a) Amount of packet flits injected in 2×2 NoC.



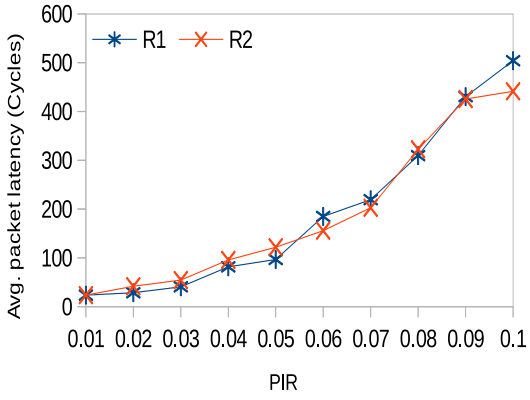
(b) Amount of packet flits received in 2×2 NoC.



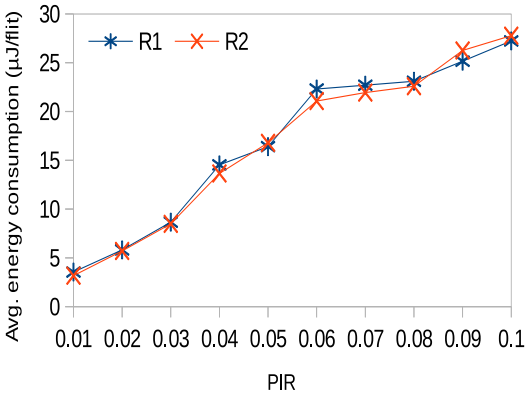
(c) Amount of packet flits lost in 2×2 NoC.



(d) Throughput behavior in 2×2 NoC.



(e) Latency behavior in 2×2 NoC.



(f) Power consumed by a flit in 2×2 NoC.

Figure 3.12: On-line evaluation of the proposed test solution applied at both test rounds on the 16-bit 2×2 NoC.

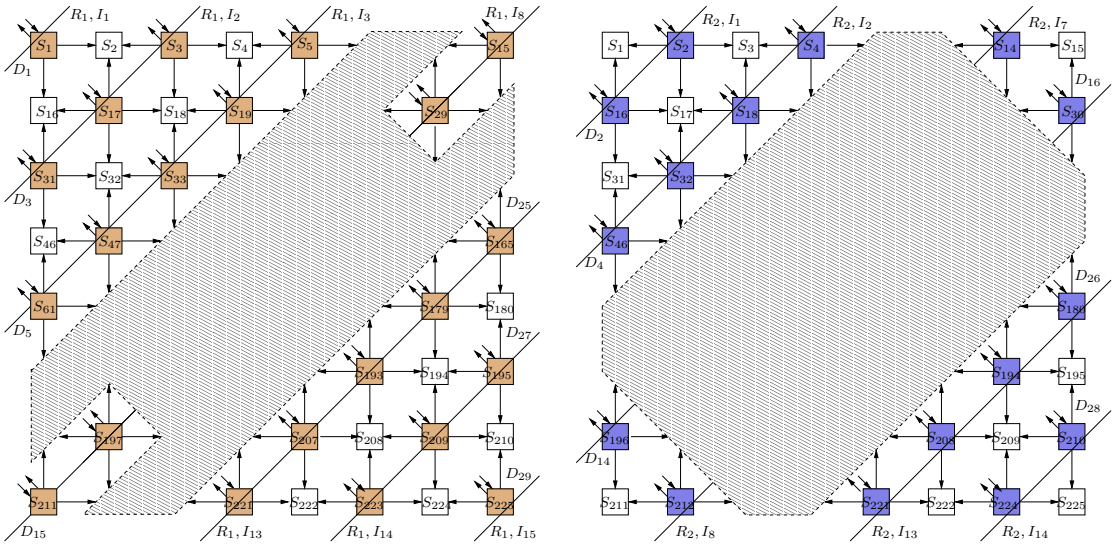
of packet flits is injected and transmitted by the XY routing. Both A1, A0 test patterns are taken as the traffic in order to observe the effect of the channel-SAFs. These test patterns are packeted using the wormhole switching technique. The performance metrics in this section are observed in the test setup for 2×2 mesh network termed as the 4-IP network (Figure 3.6) and are provided in Figure 3.12. The same size of packet flits shown in Figure 3.12a is sent on the network for both the test rounds. A network may receive all most same amount of traffic sent but cannot accept more than what is injected. However, it is an ideal case. In a practical situation, routing limitations and collisions prevent accepting all traffic. Thus some packets get dropped in the network. Additionally, faults in channels may enhance the dropping. The size of packet flits received and dropped is demonstrated in Figures 3.12b and 3.12c, respectively. Note that the amount of received flits include misrouted (due to SA0/SA1 faults on CWs that carry packet header), corrupted flits (due to SA0/SA1 faults DWs that carry application data) with normal flits. It is seen that $\approx$ 10–18% and $\approx$7–15% of the received traffic is corrupted and misrouted, respectively. On the other hand, in normal mode, it is observed that $\approx$ 5–10% of the total flits sent are dropped due to timeout fault. But, this timeout is enhanced due to SA0/SA1 faults on CWs that carry packet trailer resulting $\approx$ 8–17% dropping of the sent flits. In the case, Figures 3.12d, 3.12e, and 3.12f provide statistics for throughput, latency, and power (energy) consumption behavior, respectively in the network.

3.7 Solution Scalability

A test methodology must have an important feature called scalability. Another important property of the methodology must include portability on a network rather than its scope in traditional $M \times N$ networks. A clear advantage of the proposed test approach is that it scales with network size and channel width. In the previous section, the effectiveness of the proposed approach is discussed with a basic 4-IP network where n is confined to 16-bits. In this section, the proposed test approach is studied with a comparatively larger 225-IP network having 16-bit channels and 25-IP network with 32-bit channels to illustrate the test scalability. The portability of the proposed approach is demonstrated with an octagon network in Section 3.8.

3.7.1 Scalability with NoC Size

The scalability of the proposed test scheme for channel-SAFs with respect to network size is illustrated on a 15×15 network with 16-bit unidirectional channels. Basic characteristics of the network are provided in Table 3.12. It shows that the network consists of 225 routers, 225 IP cores, 1290 channels (840 interswitch and 450 local channels). Since each channel is considered to be 16-bits, thus the test mechanism needs to detect 41280 SAFs (Figure 3.5), i.e., 20640 as SA0s and 20640 as SA1s are injected in turn. Figure 3.13 demonstrates round wise experimental setup for the 15×15 (225-IP) network. At the first round, the test iterations



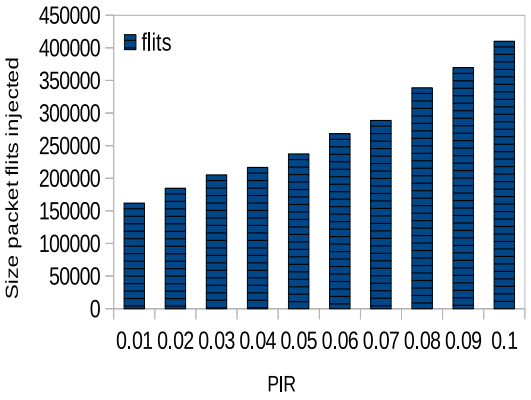
(a) Sequencing test applications at odd diagonal levels as TR=I (R_1).

(b) Sequencing test applications at even diagonal levels as TR=II (R_2).

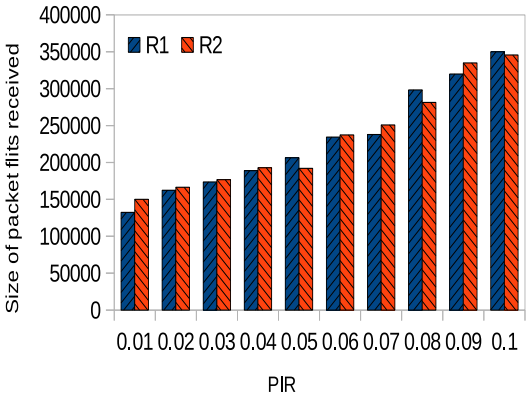
Figure 3.13: Application of the proposed test solution at different test rounds and iterations on 15×15 NoC with unidirectional channel configuration.

as seen in Figure 3.13a are applied by the Brown colored nodes (test sources). The nodes are considered to be on the odd diagonal level. The test iterations in the second round are shown in Figure 3.13b and initiated by the Blue colored nodes at the D_2. The test setup is made with Xilinx ISE Design Suite. TPGs at one side while TRAs at another side of the channels under test in an iteration are placed to detect and locate SAFs on these channels. Repeating round-wise test iterations, all the channels in the underlying 225-IP network are examined. Since the bit-width of the channels remain fixed, the quality metric- area overhead remains unchanged (Table 3.13) while the test time metric for the network linearly increases with the Tits. It is seen that the network needs 319 clocks (Figure 3.8) to address injected SAFs. Consequently, other quality metric- coverage metrics (LCM and FCM) can be viewed in Tables 3.14 and 3.15, and Figures 3.9 through 3.11. Thus, all modeled faults are detected resulting 100% achievement.

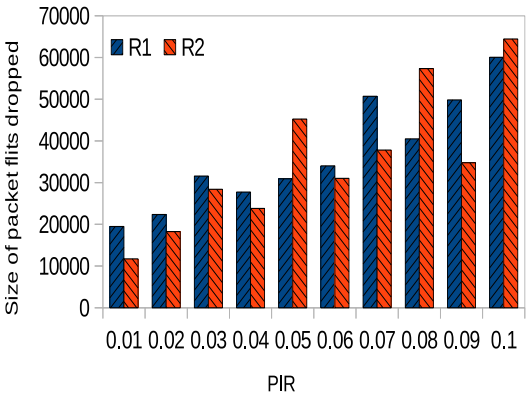
To observe the effect of SAFs on the system performance, the simulation is extended on the 16-bit 225-IP network. Figure 3.14 illustrates the on-line evaluation of the proposed diagonal node selection based test model named D-Model on the 225-IP network. Amount of packet flits injected in the network is shown Figure 3.14a while the amount of received and dropped flits are shown in Figures 3.14b and 3.14c, respectively. Among the received flits, $\approx$ 15–32% as corrupted and $\approx$ 12–23% as misrouted flits are realized. In the normal mode, 7–14% of the injected flits get dropped due to timeout error while it is boosted to 15–31% due to SAFs experienced in a channel of the routing paths. Consequently, the simulator maintains the statistics for the performance metrics as provided in Figures 3.14d, 3.14e and 3.14f.



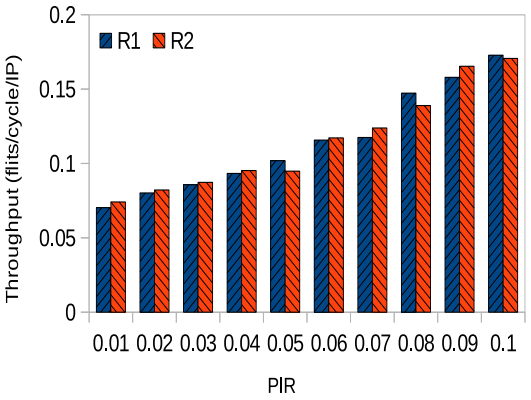
(a) Amount of packet flits injected in 15 × 15 NoC.



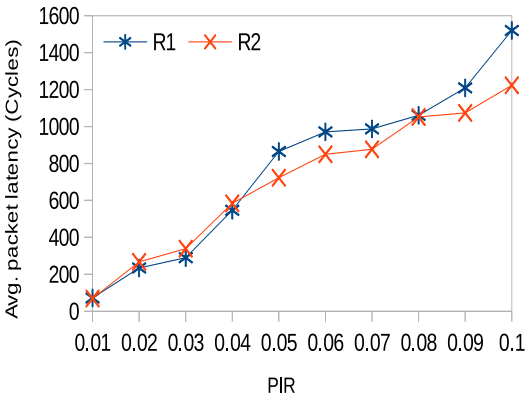
(b) Amount of packet flits received in 15 × 15 NoC.



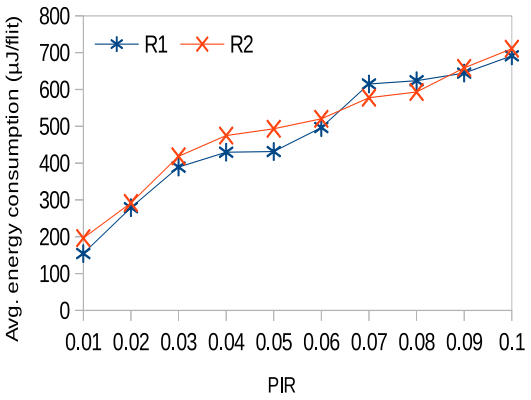
(c) Amount of packet flits lost in 15 × 15 NoC.



(d) Throughput behavior in 15 × 15 NoC.



(e) Latency behavior in 15 × 15 NoC.



(f) Power consumed by a flit in 15 × 15 NoC.

Figure 3.14: On-line evaluation of the proposed test solution applied at both test rounds on the 16-bit 15 × 15 NoC.

3.7.2 Scalability with Channel Width

Many NoC-based communication systems host various complex, heterogeneous sets of applications. Though each application has access to shared resources to meet the provision of communication services it is inadequate for many applications, e.g., multimedia applications like MPEG-2 video [172], networking applications like telecommunication, network security, and network operation domain [173] to achieve the average well-defined performance metrics. This is because such applications intend delivery of high volume traffic that consequently demands high channel bit-width and the limited power consumption. At the same time, the traffic must be transported reliably. In such a scenario, the proposed test approach must scale with higher n.

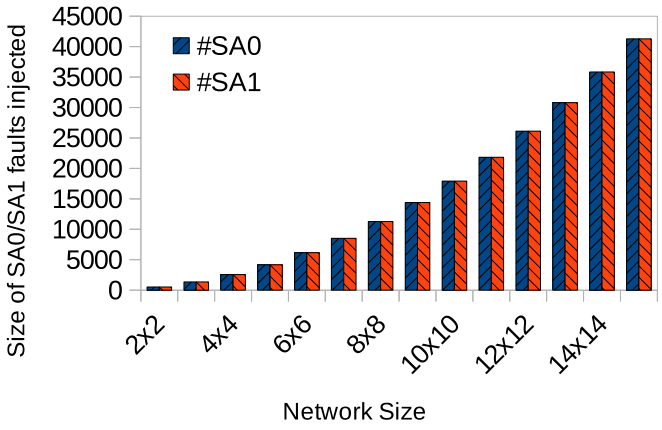


Figure 3.15: Fault injection campaign in 32-bit NoCs.

Table 3.16: Area overhead of a TM for 32-bit channel.

TM Unit	#GC	RASoC(%)	Xpipe(%)
TPG	73	3.32	3.49
TRA	51	2.23	2.37
Total	124	5.55	5.86
TPG-R	105	5.88	6.28
TRA-R	69	3.99	4.27
Total	174	9.87	10.55

The proposed D-Model is now evaluated on many $M \times N$ networks with 32-bit channels. In each 32-bit channel, first and last 2-bits, as mentioned earlier, are used for control bits (i.e., CWs) and handshake bits (i.e., HWs) while rest 28-bits are dedicated for data bits (i.e., DWs). Basic characteristics of the networks provided in Table 3.12 remain same except the wire size which becomes double now. The number of channel-SAFs that must undergo the

testing in the networks is given in Figure 3.15. Two test data sets, each containing 32-bit A1 and A0 vectors are injected from a test source (node), transported and applied to the channels under test. Then test outputs (responses) are extracted and analyzed. Therefore, TPG and TRA components of the TM for 32-bit channels contribute little (1–2%) more area overhead at both cores and routers over the TM synthesized for 16-bit channels. In this case, Table 3.16 provides the synthesis results for a core-TM and router-TM. The implementation of a TM block with the increase in n, for instance, *n=32-bits*, enhances a small burden on hardware area in cores and routers in the networks. However, such implementation makes the proposed test model be a time-independent solution for the networks. It means, the time required to test SAFs in an *n-bit* channel equals that required for an *mn-bit* channel. Therefore, Figure 3.8 provides the $T_{n/w}$ needed for detecting the channel-SAFs in the 32-bit $2 \times 2 - 15 \times 15$ networks.

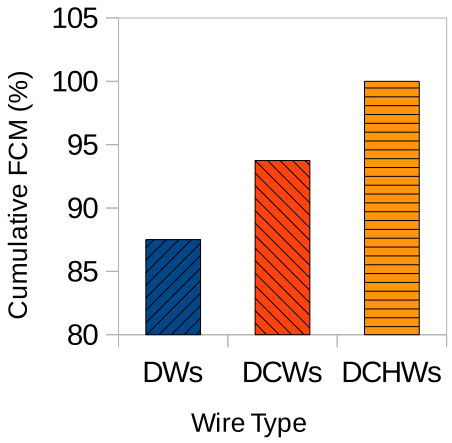


Figure 3.16: Cumulative FCM in networks with 32-bit channels.

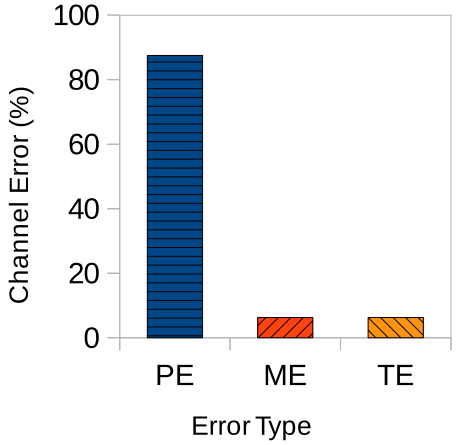


Figure 3.17: Expected channel error in networks with 32-bit channels.

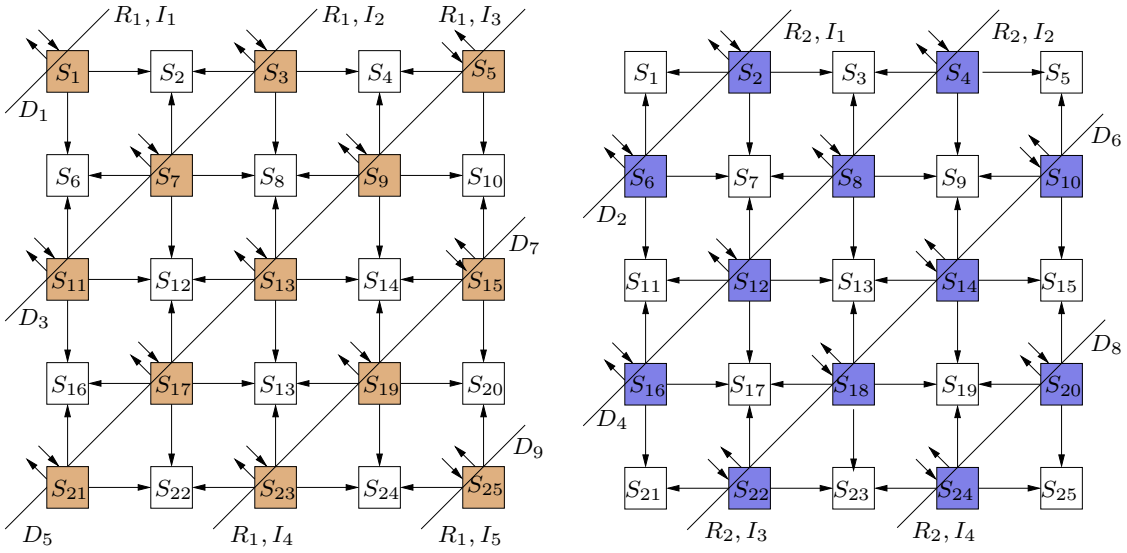
The number of test iterations is independent of the channel bit-width n. It directly ensures that the underlying LCM evaluation remains unchanged. Then, Table 3.14 that establishes the LCM achievement for the networks with 16-bit channels in a test iteration of a test round, can also be referred for this quality metric in the same iteration and round on the networks with higher channel bit-widths, for instance, 32-bit. On the other hand, the FCM varies with the channel bit-width in spite of keeping bit-widths of some wire sets (here CWs and HWs) fixed. Applying the test sets and analyzing responses on the DWs, CWs, and HWs consecutively, one can achieve the cumulative FCM as shown in Figure 3.16. Consequently, the size of SAFs detected after a test round on a subset of $M \times N$ networks is provided in Table 3.17. The effect of SAFs on these channel-wires are realized in terms of payload,

Table 3.17: Size of SA0 or SA1 detected in networks of 32-bit channels.

N/w Size	R_1			R_2			Detected		
	#DWs	#CWs	#HWs	#DWs	#CWs	#HWs	#SA0	#SA1	#SAF
2×2	224	16	16	224	16	16	512	512	1024
3×3	616	44	44	560	40	40	1344	1344	2688
4×4	1120	80	80	1120	80	80	2560	2560	5120
5×5	1848	132	132	1792	128	128	4160	4160	8320
6×6	2688	192	192	2688	192	192	6144	6144	12288
7×7	3752	268	268	3696	264	264	8512	8512	17024
8×8	4928	352	352	4928	352	352	11264	11264	22528
9×9	6328	452	452	6272	448	448	14400	14400	28800
10×10	7784	556	556	7896	564	564	17920	17920	35840
11×11	9576	684	684	9520	680	680	21824	21824	43648
12×12	11368	812	812	11480	820	820	26112	26112	52224
13×13	13496	964	964	13440	960	960	30784	30784	61568
14×14	15624	1116	1116	15736	1124	1124	35840	35840	71680
15×15	18088	1292	1292	18032	1288	1288	41280	41280	82560

misrouting, and dropping errors. Figure 3.17 shows these errors that may be experienced in 32-bit channels of these networks.

The on-line evaluation of the proposed D-Model to see the effect of channel-SAFs on basic performance metrics is done on the 5×5 network of 32-bit channels. Basic test application following the proposed scheduling approach is shown in Figure 3.18. The first round test iterations are shown by Brown colored nodes (Figure 3.18a) whereas, the Blue colored nodes (Figure 3.18b) represent the next round test iterations. The simulation setup is extended on the network. Simulations are performed for both test rounds with A1 and A0 test sets as traffic (packet flits). Figure 3.19 demonstrates the performance evaluation of the proposed solution on the 5×5 network of 32-bit channels. The amount of traffic injected on the network is provided in Figure 3.19a. Subsequently, the amount of traffic received and dropped by the network in a round are provided in Figures 3.19b and 3.19c, respectively. Since the bit-width of the channels in the network is higher, more traffic may be affected by faults. We see that $\approx$ 17–26% and $\approx$ 9–15% of received traffic are affected due to packet corruption and misrouting failure modes, respectively caused by SAFs. On the other hand, $\approx$ 12–20% of the injected traffic is lost in the network due to timeout error caused by SAFs and related routing constraints. Consequently, the network performance metrics can be observed as shown in Figures 3.19d, 3.19e, and 3.19f.



(a) Sequencing test applications at odd diagonal levels as TR=I (R_1).

(b) Sequencing test applications at even diagonal levels as TR=II (R_2).

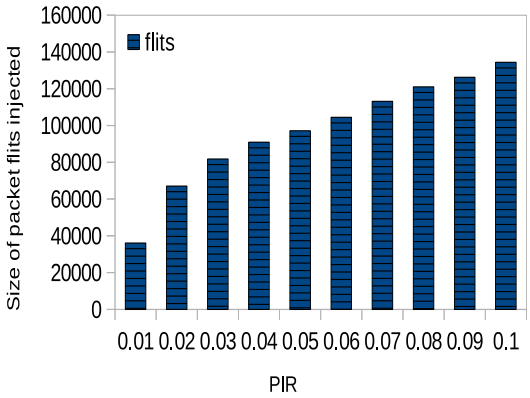
Figure 3.18: Application of the proposed test solution at different test rounds and iterations on 5×5 NoC with unidirectional channel configuration.

3.8 Solution Portability

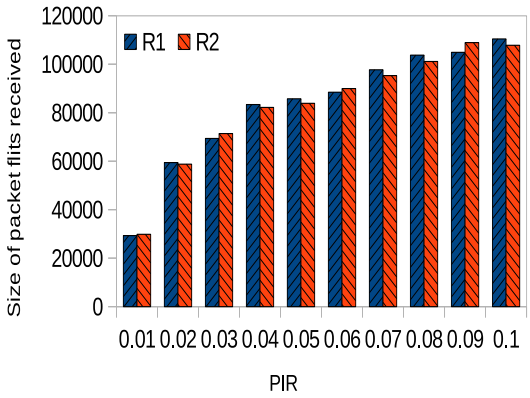
Till now, it is demonstrated that the proposed D-Model does not only scale with network size but also scales with channel bit-widths. Subsequently, the proposed D-model is evaluated with respect to many $M \times N$ mesh networks having 16 and 32-bit channels. An aspect of the test methodology must include its adaptability with network topologies. It is seen in practice that most of the existing test approaches are however applicable for mesh topology but not with other networks, such as octagon network. The basic reason is the typical network architecture, e.g., non-rectangular.

An octagon NoC is another on-chip communication architecture and is proposed to meet performance requirements of several network processor SOCs by delivering application packets at most two hops between any two nodes. Because of several advantages such as cost, performance, and scalability offered by an octagon network, it becomes suitable for supporting aggressive on-chip communication demand of networking SOCs and related domains [87]. But, manufacturing SAFs can be experienced in the octagon's channels. A basic octagon network can be characterized by 8 nodes and 40 channels (24 interswitch and 16 local). In this section, the proposed test mechanism is applied on the octagon NoC of 16-bit channels. Therefore, 1280 SAFs (640 SA0s and 640 SA1s in turn) injected into the channels of the network are needed to be addressed by the test mechanism.

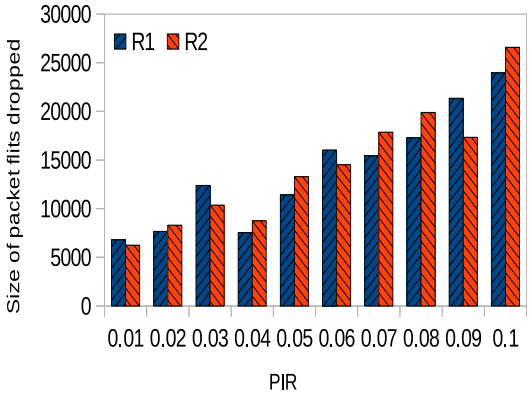
The limitation of the proposed D-Model is that it may not be applied to an octagon network directly because determining the odd and even diagonal levels on the network may



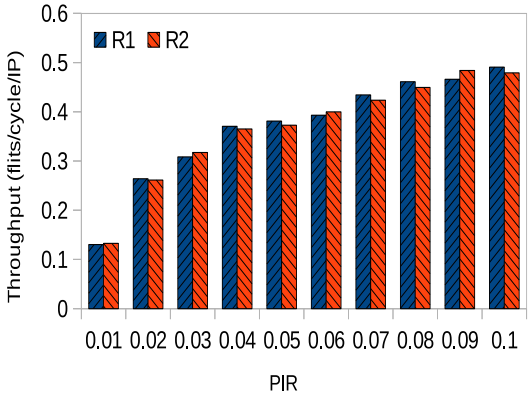
(a) Amount of packet flits injected in 5×5 NoC.



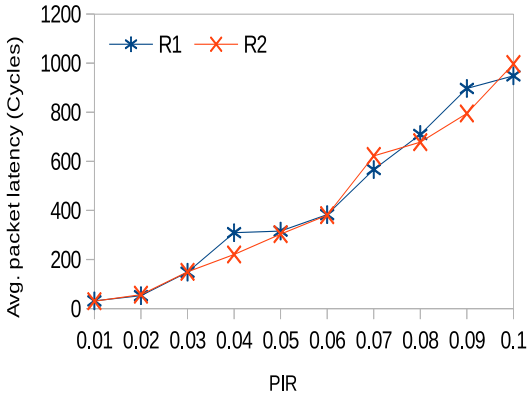
(b) Amount of packet flits received in 5×5 NoC.



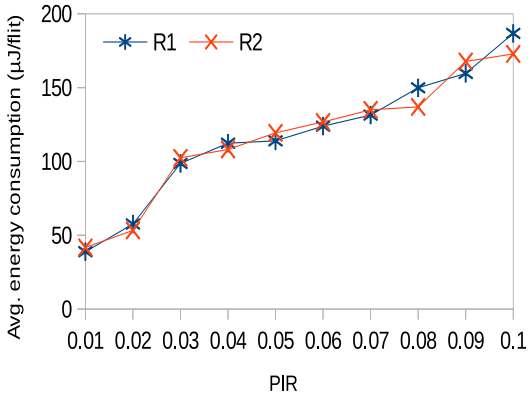
(c) Amount of packet flits lost in 5×5 NoC.



(d) Throughput behavior in 5×5 NoC.

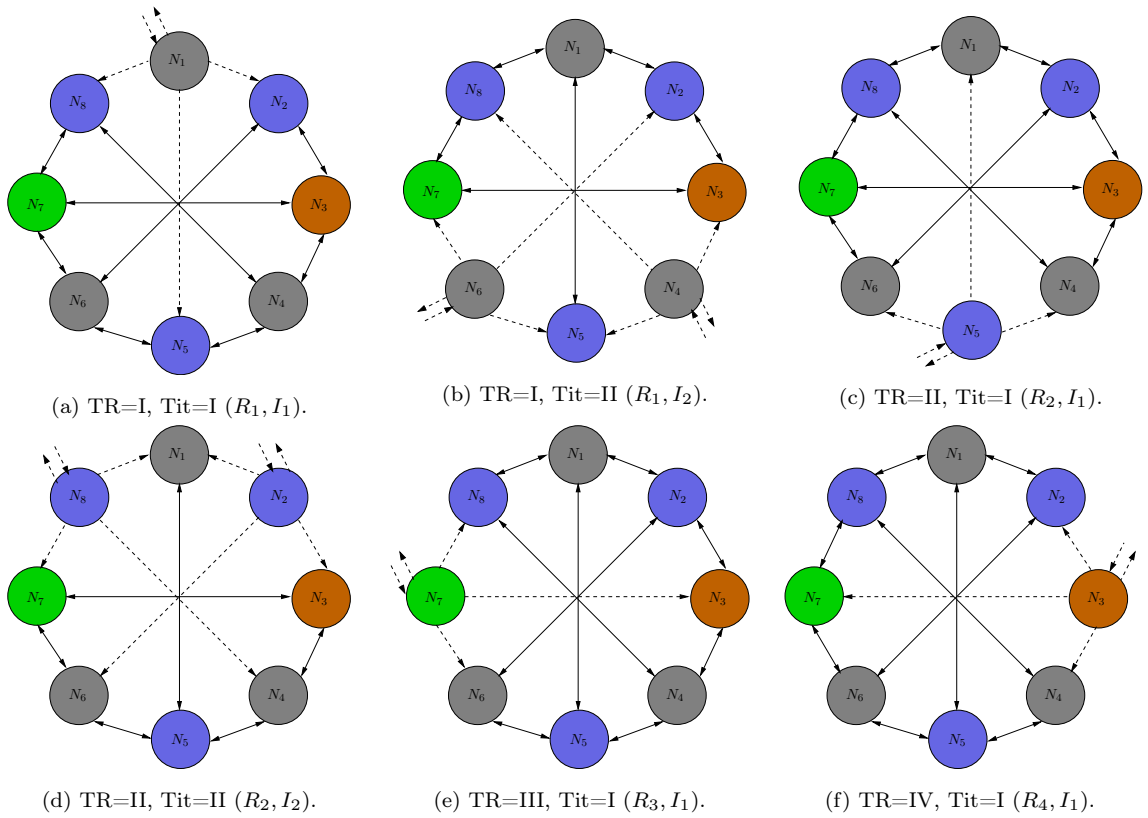


(e) Latency behavior in 5×5 NoC.



(f) Power consumed by a flit in 5×5 NoC.

Figure 3.19: On-line evaluation of the proposed test solution applied at both test rounds on the 5×5 NoC with 32-bit channels.



(a) TR=I, Tit=I (R_1, I_1). (b) TR=I, Tit=II (R_1, I_2). (c) TR=II, Tit=I (R_2, I_1).

(d) TR=II, Tit=II (R_2, I_2). (e) TR=III, Tit=I (R_3, I_1). (f) TR=IV, Tit=I (R_4, I_1).

Figure 3.20: Application of the proposed test solution at different test rounds and iterations on an octagon NoC with unidirectional channel configuration.

become complex or equivocal. But, by making a small variation in the test scheduling policy, the modified D-Model can be applied to the network. The variation is done by mapping the problem of finding odd-even diagonals into the problem of graph coloring for the test rounds. As discussed in Section 3.5, the number of test rounds in general networks including the underlying octagon NoC equals the number of distinct colors used (Equation 3.13). Subsequently, the number of test iterations per round and total test iterations are found in Equations 3.14 and 3.15, respectively.

Figure 3.20 demonstrates the application of the proposed test mechanism for channel-SAFs on an octagon network. The TPGs as test source applies the 16-bit A1 and A0 test sets on the channels and the receiver's TRA diagnoses the SAFs on the basis of responses. Now, every router in the network has three neighbor routers. A TPG/TRA at a router transmits/receives test packets (at most) to/from these neighbors. Thus, the area overhead of the TMs at all routers are the same. In Table 3.13, the size of a router-TM is synthesized with respect to the router having four neighbor routers. Therefore, the size of a router-TM in an octagon network naturally becomes smaller. But, from designer's viewpoint, the same router-TM is kept in use (as given in Table 3.13) because a designer may need to construct

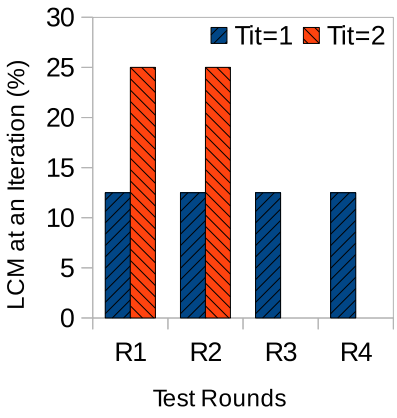


Figure 3.21: LCM achieved on a test round in octagon network with 16-bit channels.

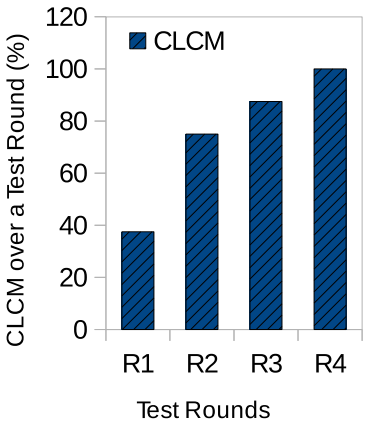


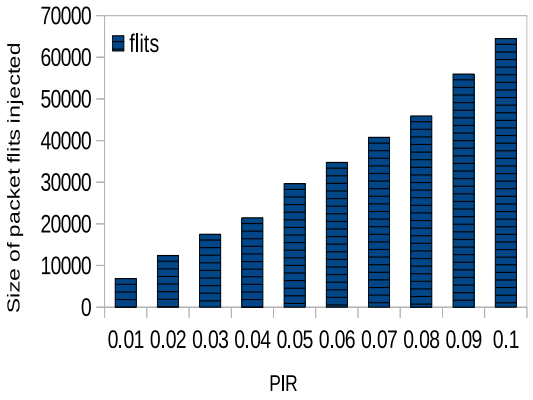
Figure 3.22: Cumulative LCM achieved on a test round in octagon network.

a large network, e.g., square-octagon NoC Architecture (SONoC) [174] by adding multiple copies of basic octagon networks.

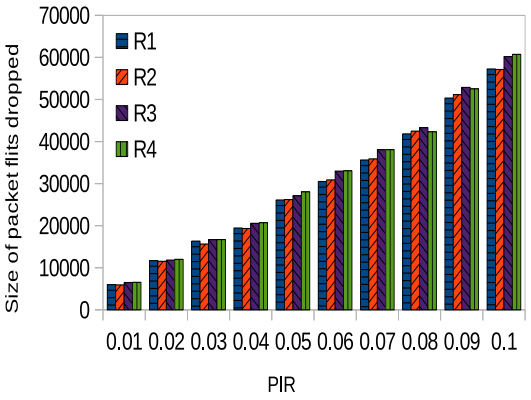
Table 3.18: Size of SA0 or SA1 detected in octagon network of 16-bit channels.

TR	Wire Sets			Faults Detected		
	#DWs	#CWs	#HWs	#SA0	#SA1	#SAF
R1	180	30	30	240	240	480
R2	180	30	30	240	240	480
R3	60	10	10	80	80	160
R4	60	10	10	80	80	160
Total	480	80	80	640	640	1280

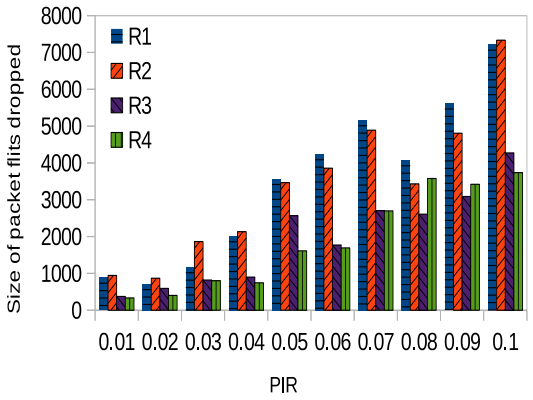
In Figure 3.20, it is seen that the proposed test mechanism is applied four times on the octagon network. Sequencing the test mechanism concludes that the channel-SAFs in the network can be addressed in just four test rounds. Note that, first two rounds are executed separately for two iterations while third and fourth rounds are executed only once. Thus, the network channels for SAFs can be tested in six test iterations and at the cost of $T_{n/w} = 66$ clocks only. On each iteration, a subset of channels (dotted arrows) undergoes the testing yielding the LCM. Figure 3.21 shows the LCM achieved in an iteration of a test round. For example, 5 channels (Figure 3.20a) have been tested in the first round (R1), first iteration (Tit=1). So, 12.50% LCM is achieved. Note that, same LCM is achieved on the first iteration of every round. Again, same LCM (25%) is achieved on second iteration (Tit=2) of both R1 and R2. Thus, cumulative LCM (CLCM) achievement shown in Figure 3.22 ensures that all



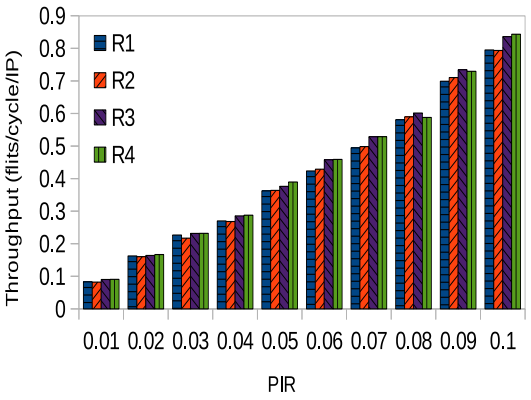
(a) Amount of packet flits injected in octagon NoC.



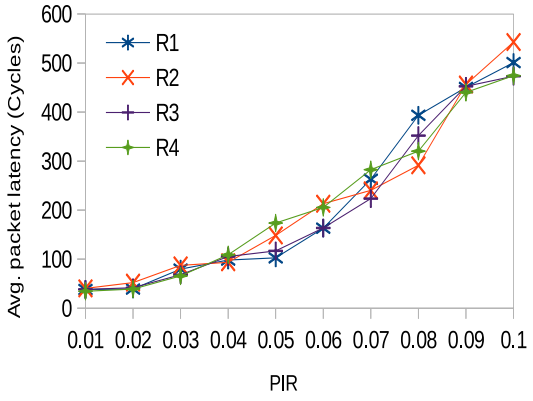
(b) Amount of packet flits received in octagon NoC.



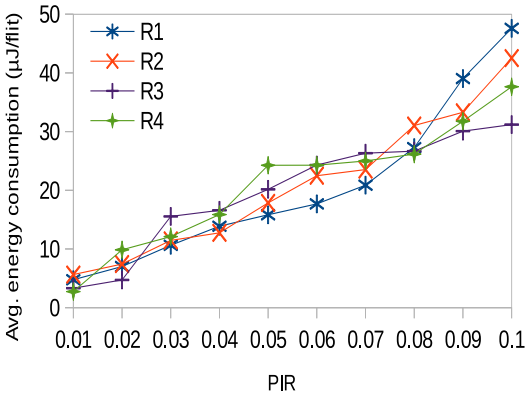
(c) Amount of packet flits lost in octagon NoC.



(d) Throughput behavior in octagon NoC.



(e) Latency behavior in octagon NoC.



(f) Power consumed by a flit in octagon NoC.

Figure 3.23: On-line evaluation of the proposed test solution applied at the test rounds on the octagon NoC with 16-bit channels.

channels have been tested. The bit-width of a channel is 16-bits. Also, the bit-width of the channel-wire sets (DWs, CWs, HWs) is maintained as mentioned in Section 3.6. Figure 3.10 shows the FCM achievement by applying the proposed test mechanism on a 16-bit channel, can also be used for the octagon network. The size of SA0s and SA1s detected by the test mechanism after a test round is provided in Table 3.18. It shows that all faults have been covered. The effect of channel-SAFs at system-level is realized in terms of channel-errors and Figure 3.11 can be referred for the errors caused due to SAFs on DWs, CWs, and HWs.

To properly evaluate the proposed test mechanism along with graph-color based advanced test scheduling approach (variation of the D-Model) on an octagon network of 16-bit channels, it is needed to carry simulation on the network. The simulation setup using the Noxim [22] simulator is extended to the 16-bit octagon network. The performance of the network in on-line mode can be observed in Figure 3.23. Packet flits are injected on the network at the PIR=0.01–0.10 and are provided in Figure 3.23a. The same size of flits is used in each test round. The SAFs whence experienced in a routing path may then infect the injected flits. The infected flits may be corrupted, misrouted, and dropped. The original flits are received with corrupted and misrouted flits in the network. Figure 3.23b shows the size of flits received in the network while the size of dropped flits due to channel-SAFs and timeout error is provided in Figure 3.23c. It is observed that 11–27% and 6–13% of the injected traffic are observed as corrupted and misrouted flits, respectively. Whereas, 7–11% flits are dropped due to SAFs in addition to 8–14% timeout flits. Thus, dropping of flits at last two rounds is comparatively less than first two rounds. It is because of the fact that the size of channels under test in later cases is less (33%) than the former cases. In addition, the later rounds last for the single iteration. The network performance characteristics namely throughput, packet latency, and energy consumption are seen in Figures 3.23d, 3.23e, and 3.23f, consecutively.

3.9 Benefits over Prior Works

The proposed test model scales to all large-scale $M \times N$ networks with size and channel-width. The scalable property of the model is established by its application on small to large size mesh networks. Besides the scalability, a variant of the model is also discussed to widen its scope. The proposed test model thus ensures its application to general NoCs irrespective of network size, channel width, and network type. Additionally, the proposed solution must be well accepted. Therefore, it should have advantages over the prior approaches in terms of various quality metrics. In this section, a comparison is studied among the prior test models including the proposed solution on the set of $2 \times 2 - 15 \times 15$ mesh and octagon networks. The following prior test approaches[1] namely 2×2-Model [38, 39], partial 2×2-model (P-2×2-

[1]Few preliminary works of the author are taken in the comparison study in this chapter as well as rest of the chapters. These works are labeled as [RPC-1], [RPC-2], etc., and can be found in author's publication list.

Table 3.19: Comparison of Tits vs. Test models..

N/w Size	2x2-Model [38, 39]	S-Model [6, 7]	**H-Model**	**OE-Model**	**D-Model**
2×2	1	8	3	2	3
3×3	4	18	5	2	5
4×4	9	32	7	4	7
5×5	16	50	9	6	9
6×6	25	72	11	10	11
7×7	36	98	13	12	13
8×8	49	128	15	16	15
9×9	64	162	17	20	17
10×10	81	200	19	26	19
11×11	100	242	21	30	21
12×12	121	288	23	36	23
13×13	144	338	25	42	25
14×14	169	392	27	50	27
15×15	196	450	29	56	29

Model) [163], sequential model (S-Model) [6, 7], hierarchical approach (H-Model) [RPC-18], and odd-even model (OE-Model) [RPC-12, RPC-14] are selected to illustrate the benefits of the proposed test scheme on these prior models in terms of the following quality metrics-hardware area overhead, test time, channel errors, and performance overhead under the similar test environment.

3.9.1 Benefits in Hardware Area Overhead

The test architecture (here a TM) is a crucial component in implementing a test mechanism. The component should occupy least hardware space and be powerful enough to cover channel-faults. In 2×2-Model, basic test configuration consists of a 2×2 mesh network and is iterated on a larger mesh to cover channel-faults. In the test setup, TM units are placed in core blocks. Test packets are delivered from a core to another core in the XY routing path. The separation between sender and receiver is thus four hops (2 interswitch and 2 local channel length). Therefore, handling the test sets, a TM takes a sufficiently higher area and it is $\approx$ 21–23% of a router. The test model scales with mesh networks only due to 2×2 test configuration. However, the test configuration is partially used by the P-2×2-Model to overcome the issue of the 2×2-Model. The area overhead is seen to be 73.84–78.01% more than the proposed TM. In P-2×2-Model, two neighbors TMs separated by three hops (1 interswitch and 2 local channels) exchange test sets to address SAFs. Consequently, little reduction in the area overhead is observed. In S-Model, the SAFs are addressed in interswitch channels only and

Table 3.20: Comparison of Test time (clocks) vs. Test Models.

N/w Size	2x2-Model [38, 39]	S-Model [6, 7]	**H-Model**	**OE-Model**	**D-Model**
2×2	30	88	120	22	33
3×3	120	198	200	22	55
4×4	270	352	280	44	77
5×5	480	550	360	66	99
6×6	750	792	440	110	121
7×7	1080	1078	520	132	143
8×8	1470	1408	600	176	165
9×9	1920	1782	680	220	187
10×10	2430	2200	760	286	209
11×11	3000	2662	840	330	231
12×12	3630	3168	920	396	253
13×13	4320	3718	1000	462	275
14×14	5070	4312	1080	550	297
15×15	5880	4950	1160	616	319

a TM takes 9–12% area. But, local channels are likely to be affected by the faults. The TM block needs additional logic gates to handle sufficient test sets thereby increasing its size to 15–17%. It is still 24.17–31.58% larger than the proposed TM. In H-Model, one channel-wire is taken at a time. Therefore, a TM takes few additional gates yielding 15.89–23.84% more area overhead. The TM blocks used in OE-Model and the proposed D-Model are alike. No benefit is gained in terms of area overhead metric.

3.9.2 Benefits in Test Time

In the on-line mode, a part of an underlying NoC called subnet is kept busy in testing its channels while rest of the network can be used to transmit packets. The selection of suitable subnets determines the number of test rounds which is completed in few test iterations. Again, these iterations incur the $T_{n/w}$, total test time (cost) in addressing the underlying channel-faults. The number of test iterations and corresponding test time (clocks) needed to address channel-SAFs by the prior models on the $M \times N$ networks are provided in Tables 3.19 and 3.20, respectively. An improvement in test time of the proposed D-Model over the prior models on these networks is provided in Table 3.21. In 2×2-Model, basic test configuration is iterated on larger $M \times N$ mesh NoCs resulting $(M-1) \times (N-1)$ test rounds. Each round is completed with the single iteration where each test packet from test source is analyzed after traversing four hops. All four routing paths in the 2×2 neighborhood are put to test that takes 30 clocks. Apparently, though the model gains by 3 clocks over the D-Model but its

Table 3.21: Improvement (%) on Test time (clocks) by the proposed D-Model over existing test models.

N/w Size	2x2-Model [38, 39]	S-Model [6, 7]	**H-Model**	**OE-Model**
2×2	–	62.5	72.5	–
3×3	54.17	72.22	72.5	–
4×4	71.48	78.13	72.5	–
5×5	79.38	82	72.5	–
6×6	83.87	84.72	72.5	–
7×7	86.76	86.73	72.5	–
8×8	88.78	88.28	72.5	6.25
9×9	90.26	89.51	72.5	15
10×10	91.4	90.5	72.5	26.92
11×11	92.3	91.32	72.5	30
12×12	93.03	92.01	72.5	36.11
13×13	93.63	92.6	72.5	40.48
14×14	94.14	93.11	72.5	46
15×15	94.57	93.56	72.5	48.21

(2×2-Model) application on a 2×2 network becomes equivalent to an off-line test application since all channels are put to test. In the place, OE-Model that selects a neighborhood of 16-nodes in a test round may be preferred. The approach then saves 26.67% test clocks. In S-Model, each test round is equivalent to a subnet that consists of a node and its neighbors. Every round is completed in two iterations. In the first iteration, all outgoing channels while incoming channels of a node in next iteration are tested. Thus, the number of test iterations obtained by this test model on a network of $\aleph$ nodes is $2\aleph$. As seen in Table 3.21, the proposed D-Model improves test clocks by 62.50–93.56%. Although, the H-Model and the D-Model has addressed channel-SAFs on the same number of test iterations, yet the later model saves 72.50% test clocks over the former model because its test mechanism applies every test bit from the test sets serially. Apparently, the OE-Model takes fewer test clocks over the D-Model for the networks up to 7×7 dimensions. On the contrary, application of the OE-Model on smaller networks (of size up to 4×4) has no longer remained in the on-line test mode. The reason is that the basic test subnet (test round) in the approach consists of a neighborhood of 16 nodes. Thus, the number of test rounds on a network of $\aleph$ nodes is $\approx \lceil \aleph/16 \rceil$ resulting $\approx \lceil \aleph/8 \rceil$ test iterations. However, for larger networks, the model becomes time-inefficient. The proposed D-Model, for example, improves test clocks by 6.26–48.21% on the $8 \times 8 - 15 \times 15$ networks. Therefore, the D-Model becomes faster up to $16\times$ in terms of test iterations and save test clocks up to 94.57% over the prior models on a set of networks described in Table 3.12.

The 2 × 2-Model does not fit on a network other than meshes, for example, octagon network. In the place, P-2 × 2-Model can be employed where every interswitch channel and its adjacent local channels are put to test in an iteration. Since test packets are analyzed after traveling three hops, the time per iteration becomes 23 clocks. So, the addressing channel-SAFs on a basic octagon NoC take 552 clocks resulting 88.04% more than that of the variant of D-Model. Other test models can also be applied to the octagon network. Subsequently, 176, 200 clocks are taken in this network by the S-Model and H-Model resulting 62.50%, 67% more test time overhead, respectively than the proposed solution. Note that, the OE-Model though takes only 22 clocks but the test application as mentioned earlier, is no longer in the on-line mode. Thus, the variation of the D-Model becomes 4× faster in terms of test iterations that saves test clocks up to 62.50–88.04% on the octagon network.

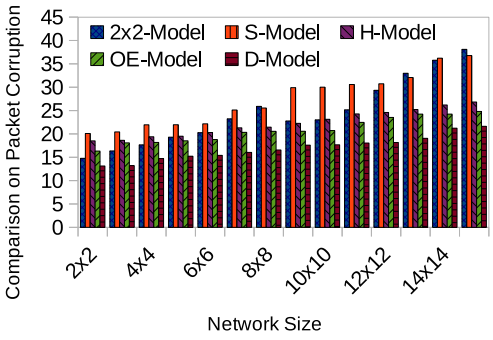


Figure 3.24: Comparison on payload error vs. Test models on networks of 16-bit channels.

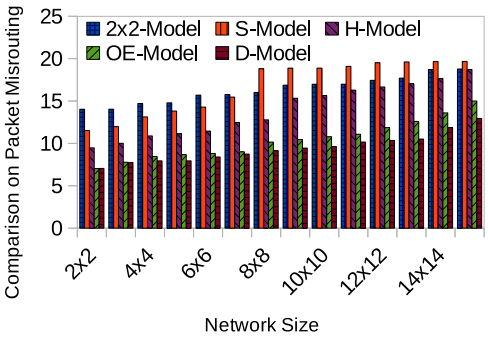


Figure 3.25: Comparison on misrouting error vs. Test models on networks of 16-bit channels.

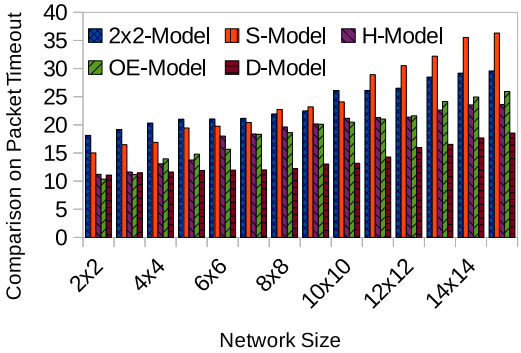


Figure 3.26: Comparison on timeout error vs. Test models on networks of 16-bit channels.

3.9.3 Benefits in Channel Errors

The manufacturing defects in channels put on-chip communication architectures into different system-level failure modes. Depending on the type of faults experienced in the channels, these

failure modes are observed. For example, the failure modes due to channel-SAFs are described with corruption, misrouting, and dropping of packets. Consecutively, these modes are realized in terms of errors in channels that affect the application packets. The substantiality of the errors that produce infected packets depends on the size of SAFs experienced in channels of a routing path. An NoC architecture should embed a post-manufacturing test mechanism to address these faults. It is necessary for a fault-tolerant routing algorithm that uses the diagnosis results to tackle a faulty channel of a routing path. Furthermore, the number of test rounds, as well as test iterations per round, may bring more infected packets. Because application packets get more chances to enter a test region where channel-SAFs are assumed to exist. Figures 3.24, 3.25, and 3.26 manifest the infected packets due to channel-SAFs as the channel errors. The manifestation is observed by applying a set of test schemes including the proposed D-Model on the 2×2–15×15 networks. For example, when the 2×2-Model is applied, 14.78–38.10%, 14.03–18.74%, 18.13–29.57% of the injected packet flits are observed respectively as payload, misrouting, and timeout errors which are noticeably more than as observed by the proposed D-Model.

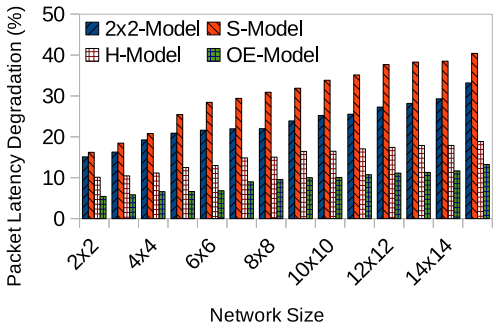


Figure 3.27: Improvement (%) on packet latency by the proposed D-Model over existing test models.

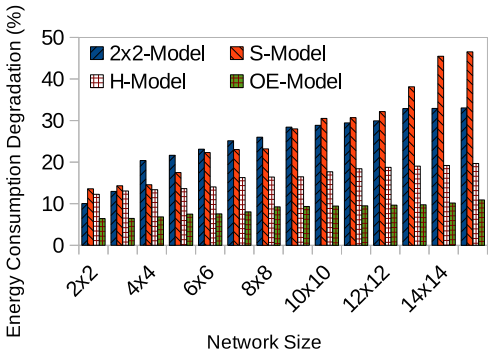


Figure 3.28: Improvement (%) on energy consumption by the proposed D-Model over existing test models.

3.9.4 Benefits in Performance Overhead

The effects of channel-SAFs are realized in the shape of errors which significantly affect the performance characteristics of NoC-based communication systems. Simultaneously, the effects become more evident with the number of test iterations along with other routing constraints. Higher is the number or more is the test time taken in an iteration, worse is the NoC performance degradation/overhead. Here, the performance characteristics in terms of packet latency and energy consumption are observed by conducting system simulations. The simulations are done to evaluate the prior test solutions on the networks characterized in Table 3.12. As a result, one may be acknowledged of the superiority of the proposed solution

over the prior approaches. Figure 3.27 refers the packet latency degradation of the prior models over the D-Model. In the on-line mode, multiple incoming application packets have to wait at a node currently busy in testing its channels and the arbiter unit does not release the packets on the channels until the node has left the ongoing test mode. As a result, the latency of a packet is increased. For example, when the 2×2-Model is applied on a 2×2 network, packet latency on average is 15.11% more and the parameter degrades to 33.19% whence the model is applied to the 15×15 network. Likely, sequencing the simulations by the S-Model, H-Model, and OE-Model on the network sizes ranging from 2×2–15×15, one may expect, respectively 16.22–40.40%, 10.12–18.89%, and 5.43–13.26% more packet latency. The deterioration in energy consumption for the prior methods is shown in Figure 3.28. As mentioned, applications duly wait at a node before forwarding on a channel till its testing is over. In this period, network resources are utilized to hold the packets at the node as well as preceding packets on the routing path. The energy consumed by a packet flit is just like the latency is increased. For example, 10.03% and 33.53% more energy are consumed by an application of the 2×2-Model on 2×2 and 15×15 networks, respectively. Subsequently, applying the S-Model, H-Model, and OE-Model on the set of 2×2–15×15 networks, it can be observed that the proposed D-Model consumes 13.57–46.52%, 12.26–19.68%, and 6.41–10.90% energy less than those prior models.

3.10 Limitations

So far the discussion, the proposed test scheduling has ensured that any $M \times N$ network can be tested for channel faults by the $M + N - 1$ test iterations. Whereas, other networks in one scheduling method are modeled to an $M \times N$ matrix that results in $M + N - 1$ test iterations. Finding the number of test iterations on these networks can also be found by the graph coloring problem treated as another scheduling method. In this case, the result is nearly same as found in the former method. Thus, the parameter $M + N - 1$ can be considered as the number of test iterations on a network in general. In the previous section (Section 3.9), it is noticed that the proposed test solution conveys various benefits in terms of different quality metrics over a set of prior works. One can consider it as a low-cost and fast test scheme for addressing the channel stuck-at faults in on-chip communication systems. At the same time, the proposed scheme scales to large size general NoC systems and the property is ensured by executing the *1-step* test algorithm on activation of diagonal nodes in a test iteration. However, the proposed test scheme induces a few pitfalls with reference to the test iterations and associated overhead.

- **Test Iteration:** With the increase in the network size in terms of nodes, the $M+N-1$ increases.
- **Test Time:** Each test iteration takes few clock cycles before shifting to the next

iteration. Subsequently, the test time is raised with the increase in the test iterations as defined in Equation 3.17.

- **Performance Overhead:** This test time has the significant impact on the network performance while a component of the NoCs is in the on-line test mode and results in the performance overhead which may include latency, energy consumption, and so on.
- **Test Energy:** The approach should also focus on the energy dissipated during testing of the channels for stuck-at faults.

Therefore, the proposed test model, in spite of the fact that it is advantageous to many works, needs to be improved through the reduction in the number of test iterations for the sake of quick detection of channel faults and improving the network performance degradation.

3.11 Conclusion

The proposed work in this chapter has been exclusively dedicated to present a scalable and time efficient on-line test solution to address stuck-at faults in the on-chip communication channels, i.e., the channels of NoCs. Both SA0 and SA1 faults have been considered on the channels shared by a router pair as well as a router and core pair. Further, these faults have been assumed in data, control, and handshake wires of a channel. Through detection, the proposed 1-step test algorithm has ensured the health status of the channels. The diagnosis has identified the faulty wires. This advanced knowledge about the channel wires can be exploited by a fault-tolerant mechanism or any channel repairing mechanism so that the yield and reliability in the networks can be maintained. Therefore, application packets in advance can be prevented to become corrupted or misrouted due to a faulty channel. Multiple instances of the test mechanism are allowed to execute in parallel at the nodes which are located in the diagonal positions. For this reason, the proposed test scheme has been pronounced as the diagonal model. Experimental results have demonstrated the effectiveness of the proposed test solution by its application on various networks. The evaluation has revealed that the implementation of the proposed solution incurs little hardware area overhead and can provide coverage metrics up to 100%. The on-line evaluation has demonstrated the network performance with various system-level failures. A comparison study between a set of prior works and the proposed solution grows interests on the acceptability of the diagonal model. However, this diagonal model shows a basic disadvantage that the number of test iterations depends on the number of odd and even diagonal levels in NoCs. Again, these odd and even diagonal levels are determined with respect to the size of an NoC. Therefore, performance overhead associated with the number of test iterations may be enhanced while the network size increases. In the next chapter, this limitation is targeted at addressing the open faults in channels of the on-chip communication networks, i.e., NoCs.

Chapter 4

Maximal Connectivity Detection in On-Chip Communication Networks

4.1 Introduction

The rapid technological advances over the past decades have transferred the embedded systems to multicore architectures like contemporary CMPs or multiprocessor systems-on-chip (MPSoCs) in order to largely support various intensive-computing complex applications and high-performance communication. Current trends also reveal that multicore architectures are more in demand to support communication in future CMPs as they meet high-performance requirements, such as low energy consumption, high throughput, and low latency [175, 176]. However, with increasing processing elements termed as IP cores, the on-chip communication is considered to be one of the major performance bottlenecks in SoC-based architectures because of their inability to meet the rapidly increasing performance demands. Consequently, NoC-based SoC architectures, such as Intel's 80-Core Teraflops Research Chip [177], iMesh NoC of Tileras TILE64 [90] are considered as the prevalent interconnection networks for future many-core on-chip communication systems [152]. Interconnect and transistor wear out resulting from CMOS scaling leads to a shrinkage in the feature size of the chips. The shrinking nature shortens the lifespans of NoCs in the CMPs and MPSoCs. Because, the permanent faults due to manufacturing defects, process variations in the basic NoC components, e.g., communication channels, unfortunately, increases the probability of channel failures or even the breakdown of the whole NoC-based system. For instance, placements of wires in channels in NoC layouts make the wires susceptible to manufacturing open (also known as cut) faults. Consequently, these channel-open faults (COFs) have great importance on the NoCs from system's reliability and performance point of view. When the communication channels experience open faults (that may disconnect the source and destination), the best effort

delivery, one kind of network service [64], by a routing mechanism cannot persist minimum performance on the NoC system.

The NoCs as the on-chip communication networks consist of the integration of the IP cores, routers and communication channels. Various issues regarding design complexity, synthesis, advantages, challenges of an NoC architecture have been discussed in [36, 51, 152]. The channels are either shared by routers only or routers and their local cores. Accordingly, the channels are categorized into interswitch and local channels. These basic building blocks are interconnected in different ways that outcome in terms of topologies. Frequently, 2D-meshes are the most used NoC topology to academics, researchers, and industrialists (Intel, Tiler, Alteris) due to simple floorplanning, and regularity in designs. In a topology, each router has five I/O ports which are connected to channels in the direction of north (N), east (E), south (S), west (W)) and local (L). The I/O ports in the first four directions are used to connect neighbor routers via interswitch channels while the last I/O or local port is connected to its dedicated core via the local channel. Figure 4.1 represents a general architecture of a 5-port router and its connection with the core.

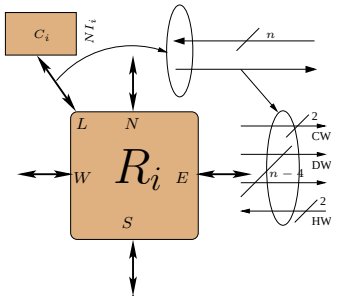


Figure 4.1: Abstract representation of an NoC node and a channel. R_i, C_i are a router and its core, respectively. $N_i \leftarrow \langle R_i, C_i \rangle$ is node. NI_i is a wrapper that wraps the C_i.

A set of n metallic wires is shaped to a channel in NoC architectures or simply termed as an on-chip communication channel. The wires in a channel whether interswitch or local are classified into (i) control wires (CWs) allotted for transporting control information like packet header, packet trailer, (ii) data wires (DWs) allotted for data delivery, and (iii) handshake wires (HWs) responsible for transporting handshake information like "ack, val". Now, the increasing demand of high-performance computation and communication for modern applications is met by increasing the number wires in channels in addition to the ever-growing number of cores and resources per chip. With continued technology scaling, designers have observed a variability in performance and reliability due to the susceptibility of metallic interconnects to many faults, such as open faults. Future prediction also reveals that 10% of the designs (that is increasing in nature) will be found to be defective [152, 176]. Therefore, the system reliability and network performance are supposed to be threatened largely by these interconnect (channel) open faults and are becoming a more critical requirement in

NoC-based designs. This chapter exclusively addresses these issues via testing of the on-chip communication channels for open faults.

Rest of the chapter is organized as follows. Background of the current work is discussed in Section 4.2. The motivation behind and multiple contributions to this work are mentioned in Section 4.3. Proposed test mechanism followed by a convenient test scheduling are discussed in Section 4.4 and Section 4.5, respectively. Section 4.6 evaluates the proposed test model. The model scalability with respect to channel width and network size, and portability with respect to network type are demonstrated in Section 4.7, and 4.8, respectively. Multiple benefits of the proposed model than the existing models in the form of detailed comparative study is analyzed in Section 4.9. Limitations of the proposed approach are discussed in Section 4.10. Section 4.11 concludes the chapter.

4.2 Background

Since last decade, the NoC architectures are witnessed as the better communication infrastructures as compared to SoCs. However, like all SoCs, the components, such as high bandwidth on-chip communication channels must also be tested for faults, for instance, open faults [175]. To cope up with the current technology trends and prevent enormous packet loss with adequately keeping the NoC performance and maintain system reliability, the following three approaches are generally practiced whence NoC channels endure faults. The first approach is to design a *test mechanism* with the capability in detecting faults on channel-wires and diagnosing them to identify faulty wires in the channels under test [28]. The second approach is to use inherent redundancy of the NoCs that keeps spare wires. In order to tackle open faults in channels, the faulty wires from a channel are replaced by spare wires. The approach is termed here as the *replacement methodology* that targets "not to allow" any fault on a channel wire. Though this replacement technique is able to deliver application data reliably on the channels, however, such scheme is not cost-effective because a limited number of spare wires may not be in the position to replace all faulty wires and subsequently reach the expected performance level. Thus, as many spares wires are included in the NoCs, the area overhead is proportionately increased [144, 145]. Third approach says to exercise a *fault-tolerant routing* algorithm [26,46,102,152–154,178]. A fault-tolerant routing that "allows to accommodate" faults on channel wires, directs network traffic (application data packets) over faulty channels, if any on the routing path, by avoiding their faulty wires or diverts the traffic route to a new routing path from a router whose intended outgoing channel in the old routing path seemed to be faulty. However, it might not be guaranteed that there will be no packet loss on exploiting the non-faulty part of a faulty channel. Because traveling traffic on non-faulty part usually utilize comparatively low communication bandwidth. As a result, packet latency on average gets increased. On the other hand, if a faulty channel is

completely avoided by the selected fault-tolerant routing, frequent diversion of traffic may be experienced on every faulty channel found in the new routing path. The literature states that most of the fault-tolerant approaches do not address a test mechanism for channel-faults [6]. Also, the replacement technique can only be executed upon the availability of the set of faulty channels and their faulty channel wires. Therefore, the knowledge of the identified faulty wires in channels is a prerequisite to a replacement as well as fault-tolerant scheme prior to their execution. This chapter focuses on the first approach that includes investigating a cost-effective on-line test mechanism for analyzing open faults on on-chip communication wires, i.e., channel-wires in NoCs so that the diagnosis information may be taken in a fault-tolerant routing or reused later by a replacement technique in order to maintain system reliability, network performance, and improve production yields.

It has been seen that typical in-field test methods work by simply stopping an ongoing application and then allow the testing of channels for possible failures. For the mission-critical systems, the ongoing application, however, cannot be interrupted [44]. The ideal solution is the application of a test mechanism in the on-line mode where a part of a network (NoC) is kept in the test mode for possible channel failure while rest is in the operational mode. However, incoming application packets may have to halt at a router till its outgoing channels (that need to transport the packets) are released from the ongoing testing. Thus, the test mechanism presented in this chapter is not only applicable at the normal mode but also at the test mode of the network. As seen so far in the literature, the prior schemes have several drawbacks that can be classified into following a set of quality characteristics: test area overhead reduction, test size reduction, test time minimization, high fault coverage achievement, and performance overhead reduction. It is, therefore, necessary to consider an improved strategy for the analysis of a channel fault, here open fault, so that these quality parameters can be improved.

4.3 Motivation, Problem Formulation, and Contributions

NoCs need to survive to manufacturing channel faults, such as open fault. An effective testing strategy that implies a fast and scalable built-in self-testing and self-diagnosis procedure for open faults in NoC channels are not only important from the system reliability and yield improvement point of view but are also significant for the network performance [179]. Communication channels are a significant part like cores, router on an NoC and deeply embedded across the chip of a die. Subsequently, poor observability and controllability characterize these channels. In an NoC-based many-core or multi-core communication system, open faults like other manufacturing faults in the channels should be addressed in order to properly analyze their effects with low test cost (in terms of test time, test area overhead, performance overhead, etc.) and prevent the disconnectedness between communicating nodes while running an application. Thus, it is mandatory to protect the system from such faults

(COFs) for the above-mentioned issues. These reasons have motivated to devise here an efficient on-line test mechanism that is dedicated to accounting open faults on control, data, and handshake wires of unidirectional channels in NoCs. The mechanism like an autonomous test system on the basis of a built-in-self-test module is encouraged for the test of manufacturing open faults. The proposed test algorithm detects open faults to ensure the health status of channel-wires and diagnoses them to separate the faulty channel-wires. An open fault in channels of a routing path may influence in disconnecting the connection between sender and receiver nodes, the test algorithm is thus used to find maximal connectivity in the NoCs in terms of the number of *correct* wires in a channel. In addition to testing the channels of a node, nodes in the NoCs are scheduled from their corner positions appearing during concurrent execution of an instance of the test algorithm. Such test execution policy is said to be expedited by the *4-corner principle*. The proposed test scheduling is considered as an optimization problem that should satisfy many constraints, such test time, system performance improvement, etc. Such scheduling, as later will be seen, significantly reduces the test time and consequent performance overhead taken to detect an open fault in NoC channels. Also, it makes the current solution scalable with respect to NoCs in terms of size and channel width, and adaptable with respect to network type. In one statement, the proposed test solution scales with all NoCs in general. It is also seen that the current solution takes same test time at little test area overhead while the channel width on a network increases.

Experimental results report that the implementation of the test mechanism incurs small hardware area as the test area overhead in a node and needs fewer clocks as the test time for addressing the channel open faults. On one hand, it is seen that test area overhead is as much as nearly 7.87% with respect to commonly used routers. On the other, it is seen that the test time per iteration is 8 clocks only. Next, fault simulation has detected all modeled open faults resulting 100% coverage metrics. In fact, even when multiple open faults per channel wire are considered, the capability of fault detection by the proposed scheme is always same. Channel open faults have considerable effects on the performance degradation in the NoC systems due to huge packet loss. For example, general performance metrics like throughput, packet latency, and packet energy consumption are directly disturbed. Simulations using a cycle accurate simulator are done to demonstrate the on-line evaluation of the proposed test solution with respect to these common performance metrics. These NoC performance characteristics are observed at large real like synthetic traffic. Present test solution provides many direct benefits. Simulation results are also compared with a wide range of prior works with respect to the proposed test solution. In comparison to prior works on a selected set of networks, the present scheme reduces hardware area overhead by 35.36–67.73%. At the same time, it reduces the overall test time by 96.43% and becomes 28× faster. Further, the proposed test scheme on performance overhead indicates that packet latency is reduced by about 5.83–42.79% and energy consumption is lowered by about 6.24–46.38%.

Thus, main contributions in this work are summarized as below.

A. Maximal connectivity testing through detecting and locating open faults in channels of NoC architectures. The detection mechanism shows the presence of the fault on a channel wire and thus ensures the health status of the wire in terms of faultiness or non-faultiness. On the other hand, locating an open fault on a channel wire results to a set of all faulty wires in the corresponding channel. Later, this diagnostic information might be exploited by a fault tolerant routing scheme to reconfigure a routing path in a way that it can maximize the utilization of a faulty channel in the routing path.

B. A convenient test scheduling scheme that minimizes the test time to a lowest possible value. It is achieved by executing the test mechanism at the nodes that are identified by the 4-corner principle on an NoC architecture.

C. Evaluation of the proposed test solution with respect to various quality parameters: test area overhead, test time, both test and fault coverage metrics, and network performance metrics on a set of selected NoC architectures.

D. Establishment of the scalable property of the proposed solution on a network independent of its size, and channel width. A larger network and a network with higher channel width are demonstrated independently.

E. Establishment of same test time which is independent of channel width on a network. It illustrates that the proposed test mechanism incurs same time while the channel width varies.

F. Establishment of adaptability property of the proposed solution on a network independent of its type. In this case, the proposed test scheme is illustrated with an octagon network.

G. Analysis of various benefits offered by the proposed solution in reference to many quality characteristics including packet loss, packet latency, and packet energy consumption.

4.4 At-Speed Test Mechanism

Due to the packet loss nature of open faults in on-chip channels, these faults must be tested at the functional speed of chips. Therefore, it is mandatory to provide an on-line test mechanism to detect and diagnose open faults on channel-wires and analyze their impact on the network performance. In the test mode, a subset of channels in an NoC architecture undergoes open fault testing while rest of the network is assumed to be operational. Execution of the test mechanism at the nodes needs testers, each of which has the capability to apply and verify test data for channel's open faults. In addition to exercising the test data, these test infrastructures

(testers) take advantages of the fault model that replicates possible instances of open faults in a channel. Before describing the corresponding test mechanism, it is better to first present a suitable open fault model and packet loss behavior of the faults followed by associated test infrastructures.

Definition 4.1. *A fault is defined as "a physical defect, imperfection, or flaw that occurs within some hardware or software component".*

Definition 4.2. *An open fault also known as a cut fault is a common logic level manufacturing defect that breaks a channel-wire $l_k; 1 \leq k \leq n$ into one or multiple segments. It is denoted by the symbol f_o.*

Definition 4.3. *The channel width n defines a set of n number of single bit metallic wires.*

Definition 4.4. *A channel wire l_k may experience an open fault at single or multiple positions. Each faults is treated as its instance f_q.*

4.4.1 Open Fault Model

Like other manufacturing faults (e.g., stuck-ats), opens are the permanent faults often experienced on channel-wires due to wear-outs in NoC-based communication architectures and have a permanent disrupt. This permanent disrupt is also treated as one of the major defect mechanisms on this component (channel). A portion of a metalization channel layer becomes disconnected due to the unintended breaks. Network traffic in terms of packet flits is transmitted in electrical signals on channels. As a result, these signals are discontinued on the opened channel wires [180]. In this work, open faults in NoC channels are treated as the manifestation of the open faults traditionally followed in VLSI circuits. One can freely reuse an existing open fault model discussed in [113–115, 181] considering a channel configuration.

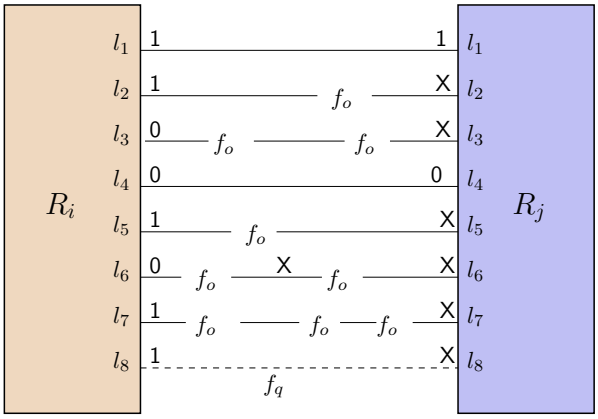


Figure 4.2: Representation of open faults in the channel (R_i, R_j) .

Figure 4.2 demonstrates a high-level representation of open faults on the wires of an interswitch channel shared by the pair of R_i, R_j routers. An open fault is denoted by the symbol f_o on a channel wire $l_k; 1 \leq k \leq n = 8$. As it is well known that an open fault can break a channel wire into multiple segments, the fault successively exists at multiple instances on the wire. The f_q represents q instances of open faults on a wire. For example, the wires l_2, l_3 experience single ($f_q = 1$) and double ($f_q = 2$) instances of open faults, respectively. Equations 4.1, and 4.2, respectively enumerate O_{ch}, O_{NoC}, the size of open faults in a channel of n-bit width and an NoC architecture having #Ch unidirectional channels.

$$O_{ch} = \sum_{k=1}^{n} l_k \times f_q \tag{4.1}$$

$$O_{NoC} = O_{ch} \times \#Ch \tag{4.2}$$

Definition 4.5. *Intra-gate open faults are the manufacturing defects that have self-supported infinite resistance and disconnect a charge/discharge path to the gate output*

Definition 4.6. *Inter-gate open faults are the manufacturing defects that leverage signal propagation on a channel.*

Definition 4.7. *Resistive open faults occur on a channel wire when they partially affect the wire.*

Definition 4.8. *Complete opens are the open defects that are "hard" to predict at the router over a channel even though the signal strength transmitted is high.*

An open fault as a manufacturing defect in the logic level is classified into two types- *intra-gate* and *inter-gate* opens. Inter-gate opens are further sub-categorized into *resistive* and *complete* opens [115]. In the case of resistive opens, packet flit still passes through the wire because of the tunneling effect. However, the signal strength is very weak due to narrow open defect. Sometimes, resistive opens for the reason are termed as weak opens. It is noted that most of the open faults belong to the inter-gate type. It can be noted that a high percentage of these faults furthermore is considered as complete opens [113, 181, 182]. In this work, complete opens as channel-open faults are considered. Application data in terms of packet flits are lost while a channel wire l_k experiences an open fault. It does not matter whether the open appears in single or multiple instances since the effect in both cases is same. Since the effect of an open fault is independent of the f_q, it suffices to assume single instance, i.e., $f_q = 1$ per channel wire. For example, an *n-bit* channel has n opens. An open fault, e.g., f_o disconnects two routers R_i, R_j over a faulty wire l_2. One wire segment connects to the R_i which is called here *stable* router. Another segment disconnects the R_j from the R_i. This R_j is called as *floating* router. Being a flit signal is lost on a faulty wire and hard to

predict at the floating router, one can model an open fault to a stuck-at-0 (SA0) or stuck-at-1 (SA1) fault for the wire. For the reason, an open fault is often regarded as a stuck-open fault. Then every flit in the logic of floating router is received with "0" or "1" while a channel wire becomes faulty with the stuck-open fault. Before forwarding a packet, floating router discards the faulty flit from the packet. Here, this lost packet flit is denoted by the symbol "X".

4.4.2 Packet Dropping: A System Level Failure

NoC-based systems with high-speed communication channels provide many quality-of-services (QoSs). However, faults in channels lower the efficiency of NoCs as well as prevent such architectures from their correct functionalities due to failures resulted by them. For example, when a channel-wire experiences an open fault then any packet flit sent over it is dropped/lost due to non-receiving of the flit at the router/core over the faulty channel. Therefore, the fault puts an NoC into the system level failure mode called packet dropping (loss). An on-chip channel consists of control, data, and handshake wires. Depending on the location of the fault in a channel, such packet dropping mode is divided into two types – partial and full. The *partial packet dropping mode* occurs when a data wire (DW) l_k experiences an open fault. Since packets in NoC system are transmitted in terms of flits then only information on the l_k is lost without affecting rest part of the packet. A handshake wire (HW) transports "val" and "ack" signals of a packet that report receiving of the packet. If l_k (a handshake wire) is affected by an open fault, then no "val" and "ack" flits are received at the source. In spite of the receipt of a packet, the source may initiate retransmission of the packet. Like data wires, the failure mode due to open faults in handshakes wires is realized in the form of partial packet dropping. The *full packet dropping mode* occurs when a control wire (CW) experiences an open fault. If the faulty l_k is a control wire that is dedicated to carrying the packet header flit (beginning of packet (bop) signal), then no routing path is set up between the stable and floating elements (router, core) over the l_k. If any attempt is made to transmit a packet via the channel including this faulty control wire, then the whole packet is lost due to the loss of the packet header on the wire and since rest of the packet flits follow the header flit. If the faulty l_k is a control wire that carries packet trailer flit, then also the end elements over the l_k never receives the flit, i.e., end of packet (eop) signal. The whole packet is naturally lost. As many as faulty control wires appear in a routing path, packet loss is correspondingly increased. In addition to packet loss caused due to faulty control wires, the effect can be enhanced with a packet timeout error. A timeout error occurs when the receiver never has the packet trailer within a predefined time period despite the fact that trailer carrying channel wire is correct. Various reasons, such as channel width, packet size, routing strategy, traffic size in the network result in the timeout error. Collectively, degraded network performance is observed for a regular application at the runtime.

4.4.3 Test Infrastructure

Figure 4.2 demonstrates an abstract representation of open fault model on a communication channel and the set of all opens that lead to the packet loss. The communication channels whether interswitch or local in an NoC architecture are broadly classified as either bidirectional or unidirectional. Test requirements and implementation for open faults in channels accordingly differ. However, it is seen in practice and literature that most of the NoC channels belong to the second class. One reason includes the simple implementation of test modules (TMs) as self-test structures for channel faults, such as opens. Open faults in this chapter are considered in unidirectional channels.

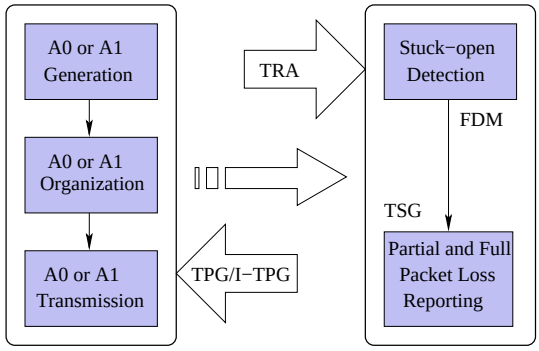


Figure 4.3: A general view of test module for addressing open faults in channels.

A high-level representation of a TM architecture is shown in Figure 4.3. A TM architecture consists of a pair of test pattern generator (TPG) and response analyzer (TRA) which are treated as the channel testers. The pair is granted as the basic building block to run the proposed test mechanism and suffices for addressing channel-opens. In order to lower test cost and performance overhead, the TM is implemented inside both core and router of a node. Noted that the TM complies with router's as well as core's logic. The TPG whether in a core/router generates the test sequence, header, and trailer information followed by their organization into a test packet. Additionally, the TPG launches the test packet on the channels under test from sender side of the channels. The core-TPG as the transmitter side *Tx*, therefore, unicasts the test packet on the local channel from the core to its linked router where the TRA as the receiver side *Rx* receives the packet. At the same time, the router-TPG as the *Tx* in the node multicasts the test packet on the interswitch and local channels to its neighbor TRAs (located in the dedicated core and adjacent routers). Consequently, the TPG of the router is made stronger by integrating additional circuit block. One can treat this integrated TPG (I-TPG) unit like a multicast wrapper unit (MWU) [3]. The TRA at the *Rx* inside a core/router verifies the responses (received test sequence) in addition to receiving the test packets. The verification is done to detect an open fault on a channel-wire as well as to identify a faulty channel-wire.

Table 4.1: A typical packet organization with A1 test pattern for detecting the struck-open fault in a channel.

Header Flit	Payload Flit	Trailer Flit
	$A1$	

Open faults at the run-time are detected and diagnosed by receiving and analysis of test packets. In this work as mentioned, channel-opens are modeled to corresponding stuck-at-0 faults. These faults can be exercised by the single test sequence namely All-One (A1). The test set is sufficient to detect all open faults in a channel. The raw set is generated by the core and router TPG units and is then packeted by adding header and trailer information. The credit-based wormhole switching is used to organize the raw A1 test set into the test packet. Every packet in an NoC has three fields- header, payload, and trailer. The A1 sequence is fragmented and placed in these fields in order to subsequently detect open faults on control, data, and handshake wires of a channel. A typical test packet organization for a channel is shown in Table 4.1. The packet size whether data/test depends on several parameters, such as network type, routing strategy, etc. But each flit in the packet has the same length and equals to the channel width. The test packet determined at the core and the router of a node is same with the basic difference in the header information. Because of unicast and multicast of the test packet from the core and the router of the node, the packet header in the former case contains the address of single destination while the field carries the address of multiple destinations in the latter case.

4.4.4 Testing of Opens in Channels from a Node

This subsection presents a general test algorithm which is capable of detecting and diagnosing open faults in on-chip communication channels. Using the functional mode of an NoC, a channel-subset is considered under test in a test iteration. Note that the test algorithm for a *n-bit* channel in the selected channel-subset is executed in the sequence of DW, CW, and HW wire-sets. The channel-open faults can be exercised by using single *n-bit* A1 test pattern. This test sequence is organized into a test packet and is applied on these wire-sets so that all modeled faults are detected. The test algorithm presented here is dedicated to addressing channel's open faults. The algorithm applies the test bits of A1 sequence in parallel on all wires of the channel. For the reason, this testing algorithm is named as the *1-step* algorithm. During testing of the channels from a node, one instance of the test algorithm is executed at the core while another instance of the algorithm is run at the router of the node. Every instance whether at core/router exercises only one test packet that is supported with the A1 sequence. First, sufficient A1 pattern is placed in the payload flit of the test packet for DWs of the channel. The header and trailer are mandatory to be applied. Payload follows the header and is followed by the trailer. These header and trailer fields are supported with additional

A1 test bits which are exercised for open faults in CWs. In this case, the "bop" and "eop" signal bits are taken care with the A1 test bits. In case of testing the HWs, another additional A1 test bits need to be considered and placed as "val" and "ack" signal bits of the test packet.

The basic principle of the *1-step* test algorithm is worked out by channel testers in two phases. In the first phase, application of the test packet that contains sufficient test information to detect an open fault on all wires of a channel under test. In the next phase, the analysis in terms of verification of the received packet called as test response. The analysis is required to detect an open fault on a wire and identify the faulty wires in the channel. The TPG and TRA architectures at Tx and Rx sides, i.e., terminal stable and floating elements of the channel under test takes the whole responsibility of the testing process using single test packet. The local channel from a core connected to its dedicated router is tested by sending the test packet using the TPG at the core. The test response is analyzed by the TRA in the connected router. On the other hand, the channels (interswitch and local) from this router are tested by transmitting the similar test packet to its dedicated core and neighbor routers. The test response at the receivers is analyzed by the respective TRAs. Note that the test in both cases is carried out at NoC's functional clock-speed, i.e., every flit of the test packet is transmitted through a channel in one clock. Once, the test packet is received at the TRA over a channel, an open fault can be detected by the unit. For instance, if any DW suffers from an open fault, then the TRA's logic receives logic-0 over the DW. Therefore, an open fault is detected. As the flit on the wire is to be dropped during application, the TRA's logic then treats this logic-0 as a "X" in the received payload flit. The TRAs otherwise receive same payload flit as it is sent. Thus, the state of faultiness or non-faultiness of the DW is established. Faults affecting CWs and HWs are tackled in a similar way. The CWs of a channel is responsible for transporting the packet header and trailer flits. If these wires are affected by open faults, the TRA over the channel accepts delivery of modified either header/trailer or both. Therefore, the whole packet during an application must be dropped in the router instead of forwarding it as affected by open faults.

Detecting of the open faults for a channel is not enough. Because the permanent fault like an open fault results in frequent traffic loss in the system leading to significant performance degradation. To address such permanent fault in a channel, the faulty channel-wires must be identified such that a fault-tolerant routing can exercise these wires to aid system performance and in order to improve system reliability. This necessitates the diagnosis of the response flit for the open faults. The identification operation for opened wires is done by the pattern checking called *signature analysis* that involves the verification of the response flit with local test sequence at the TRA. The TRA collects the response flit and performs logical "AND" operation on this flit with the local test sequence. The specific bit position containing "0" or "X" in the result infers that the wire is affected by the open fault and is treated to be faulty channel wire. Let the channel (R_i, R_j) provided in Figure 4.2 is taken for the test of

open faults. The TPG of the R_i (stable element) transmits the test packet containing the A1 sequence "1111 1111". The response is received by the TRA in the R_j (floating element). Suppose, the wires l_2, l_6 are affected by the open fault. Then, TRA over the channel receives the response as "1011 1011". One can treat it as "1X11 1X11" too. This response flit is verified with the local "1111 1111" vector already generated by the TPG in the R_j. The logical "ANDing" of these patterns as the verification operation on the response flit is provided in Equation 4.3 and 4.4, respectively. The 0s and Xs in the diagnosed patterns ensure the faulty wires in the channel (R_i, R_j).

$$\begin{cases} \wedge \begin{cases} 1011\ \ 1011 \longrightarrow \text{Response flit} \\ 1111\ \ 1111 \longrightarrow \text{Local test flit} \end{cases} \\ - - - - - - - - - - - - - - - - \\ \quad 1011\ 1011 \longrightarrow \text{Diagnosed Pattern} \end{cases} \tag{4.3}$$

$$\begin{cases} \wedge \begin{cases} 1X11\ 1X11 \longrightarrow \text{Response flit} \\ 1111\ \ 1111 \longrightarrow \text{Local test flit} \end{cases} \\ - - - - - - - - - - - - - - - - \\ \quad 1X11\ 1X11 \longrightarrow \text{Diagnosed Pattern} \end{cases} \tag{4.4}$$

Faults in channels at runtime of the NoC are realized in terms of errors in the channels. Unlike other channel faults, such as stuck-at faults, the open faults bring a system into two types of channel errors. One is payload error (PE) and another is the packet dropping error. The open faults in DWs and HWs result in partial packet loss. This failure mode is therefore manifested to the PE. On the other hand, the open faults on the CWs whether the wires carry the packet header/trailer, result in full packet loss. Thus, this failure mode is manifested as the packet dropping error. In addition, a network can observe packet timeout error (TE) in spite of the fact that no faulty channels are in the routing path. The timeout error also results in complete loss of a packet and enhances retransmission of the packet. One can estimate the packet dropping error due to open faults in channels with the timeout error in the network. A simple logical operation is used to diagnose the open faults via the analysis of the test response in the receiving side of the channel under test. The diagnosis of channel opens is performed by the fault diagnosis module (FDM) unit in the TRA and the unit forwards the result to the signal bit generator of the TRA called TSG. This unit declares the type of channel errors (PE, TE) on the basis of the faulty wire type found in the diagnosed result. The TSG generates two-bit signals mentioned in Table 4.2 to announce a channel error. For example, "01" is used to indicate that a DW, as well as an HW of a channel that has undergone the test, is infected by the open fault followed by its inclination to the payload error. Therefore, wires l_2, l_6 in

the channel (Figure 4.2) that results in partial packet loss will result in the PE if they belong to DW and HW categories. The "11" can be used as the reserve bits to indicate an error due to any other specific fault, such as a transient fault in the channel.

Table 4.2: The 2-bit TRA signals for the channel errors due to channel's open faults.

Bits	Error Type	Wire Type
00	No Error	All
01	Payload Error	DWs and HWs
10	Timeout Error	CWs
11	Other	All

4.5 Test Scheduling

The proposed test paradigm that detects maximal connectivity in an NoC-based communication architecture on addressing open faults on wires of channels basically acquires the confidence of either faultiness or non-faultiness of the wires and then isolates the faulty wires. A major part of the budget for manufacturing the NoC architecture is generally allocated for the pre-manufacturing as well as post-manufacturing testing of the NoC components for various faults. In this work, the test of on-chip communication channels for open faults is performed that contributes to the overall test cost for the NoC architecture. The test cost here is realized in terms of the test time incurred by the proposed test algorithm initiated at a node and its style of the application on the NoC. As a part of larger computation that is intended to meet high-performance communication in an NoC-based multiprocessor system, the test time is critically important to the on-line system performance, at least in terms of packet latency and energy consumption. Many types of research have devoted to reducing the test time. Here, it is improved by using the 4-corner principle that initiates the test algorithm from the corner nodes in an NoC architecture and proceeds hierarchically till no channel remains untested.

4.5.1 Determination of Corner Nodes

The physical organization that defines the interconnection among the routers and cores is reflected to an NoC topology. Based on the orientation of the interconnection, various topologies, such as mesh, torus, octagon, butterfly, small world, hybrid NoC architectures are developed [18,183]. Showing the regards to other topologies, mesh networks are preferred both in literature and applications because of their several advantages. For example, every channel has equal length and the NoC structure is symmetric that facilitates routing of packets. In addition to a mesh network, nodes in an octagon network communicate by at most two hops.

Table 4.3: Matrix representation of a 4×4 mesh/torus NoC.

$\aleph_1$	$\aleph_2$	$\aleph_3$	$\aleph_4$
$\aleph_5$	$\aleph_6$	$\aleph_7$	$\aleph_8$
$\aleph_9$	$\aleph_{10}$	$\aleph_{11}$	$\aleph_{12}$
$\aleph_{13}$	$\aleph_{14}$	$\aleph_{15}$	$\aleph_{16}$

Table 4.4: Matrix representation of a reduced mesh NoC.

$\aleph_1$	$\aleph_2$	$\aleph_3$
$\aleph_4$	$\aleph_5$	$\aleph_6$
$\aleph_7$	$\aleph_8$	$\aleph_9$
$\aleph_{10}$	$\aleph_{11}$	$\aleph_{12}$

Table 4.5: Different matrix representations of an Octagon NoC.

(a) First matrix representation.

$\aleph_1$	$\aleph_2$	$\aleph_3$	$\aleph_4$
$\aleph_5$	$\aleph_6$	$\aleph_7$	$\aleph_8$

(b) Second matrix representation.

$\aleph_1$	$\aleph_2$	$\aleph_3$
$\aleph_4$	$\aleph_5$	$\aleph_6$
$\aleph_7$	$\aleph_8$	–

(c) Third matrix representation.

$\aleph_1$	$\aleph_2$	$\aleph_3$
$\aleph_8$	–	$\aleph_4$
$\aleph_7$	$\aleph_6$	$\aleph_5$

Consequently, the network provides high performance for many multiprocessor SoCs [86, 87]. In this work, both mesh and octagon networks have been considered for the testing of channel open faults. Many tools are used to demonstrate a system. For instance, the graphical representation of the system as one of the tools helps in the easy demonstration. Equivalently, this tool is used to select the corner nodes that initiate the test mechanism on the above-mentioned networks.

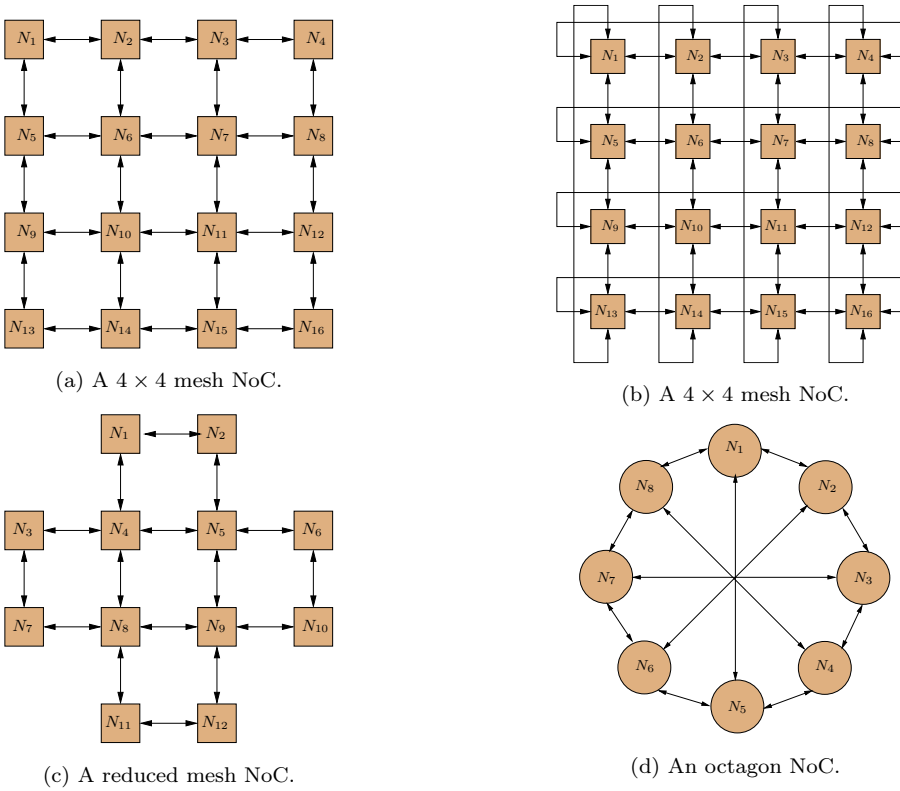
Definition 4.9. *An NoC topology is considered to be a graph* $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ *where* $\mathbf{V}$ *and* $\mathbf{E}$ *are the set of nodes and channels, respectively. Thus,* $\mathbf{V} = \{\aleph_\mathbf{i} : \aleph_\mathbf{i} \longleftarrow < \Re_\mathbf{i}, \complement_\mathbf{i} >\}$*, and* $\mathbf{E} = \{\mathbf{Ch_i} : \mathbf{Ch_i} \longleftarrow (\Re_\mathbf{i}, \Re_\mathbf{j}), \mathbf{Ch_i} \longleftarrow (\Re_\mathbf{i}, \complement_\mathbf{i})\}$*;* $i, j \in \mathbb{N}$.

Definition 4.10. *A node* $\aleph_i$ *in an NoC is a pair of router* $\Re_i$ *and its dedicated core* $\complement_i$*. It is represented as* $\aleph_i \longleftarrow < \Re_i, \complement_i >$.

Definition 4.11. *A channel* $\mathbf{Ch_i} \longleftarrow (\Re_\mathbf{i}, \Re_\mathbf{j})$ *in an NoC is said to be an interswitch communication channel when it is shared by two adjacent routers* $\Re_i, \Re_j$.

Definition 4.12. *A channel* $\mathbf{Ch_i} \longleftarrow (\Re_\mathbf{i}, \complement_\mathbf{i})$ *in an NoC is said to be a local communication channel when it is shared by two adjacent elements where one element is a router* $\Re_i$ *and another element is its connected core* $\complement_i$.

On the basis of the type of communication channels, the $\mathbf{G}$ is either undirected or directed graph. Usually, an NoC system with bidirectional and unidirectional channels are represented with an undirected and a directed $\mathbf{G}$, respectively. A matrix $\mathbf{M}$ is often used to describe the $\mathbf{G}$. The size of this $\mathbf{M}$ in terms of the number of rows and columns here depends on the number of nodes in the $\mathbf{G}$. For example, the $\mathbf{G}$ of a $\mathbf{P} \times \mathbf{Q}$ mesh network can equivalently be treated as the $\mathbf{M}[\mathbf{P} : \mathbf{Q}]$. Each element $\mathbf{M_{pq}}; 1 \leq p \leq P, 1 \leq q \leq Q$ in the $\mathbf{M}$ is a node



(a) A 4×4 mesh NoC.

(b) A 4×4 mesh NoC.

(c) A reduced mesh NoC.

(d) An octagon NoC.

Figure 4.4: Graphical representation of various NoC architectures.

of the network. Table 4.3 represents the $\mathbf{M}$ of a 4×4 mesh network (Figure 4.4a). It is very simple to consider a node as the *corner node* in a mesh network. The node $\aleph_i$ is said to be a corner node if the $\aleph_i$ has two neighbor nodes. Nodes $\aleph_1, \aleph_4, \aleph_{13}, \aleph_{16}$ in Figure 4.4a are the corner nodes. Although corner nodes are defined in a mesh network, however, the definition may not be fitted for other networks, e.g., octagon, butterfly, fat-tree, hybrid topologies, etc. In this case, two simple assumptions: one for $P \times Q$ networks, and another for a network with $\mathbf{N}$ nodes, are made to select corner nodes using matrix representation of the networks. In the former case, a $P \times Q$ network, in general, is similarly treated as a mesh network. For instance, Table 4.3 can be reused for the 4×4 torus and nodes $\aleph_1, \aleph_4, \aleph_{13}, \aleph_{16}$ are the corner nodes (Figure 4.4b). In other cases where a network does not resemble a $P \times Q$ structure, then its nodes are organized into the $\mathbf{M}$ with P rows and Q columns such that $P \times Q \leq \mathbf{N}$ and the P, Q are loosely related as $P = Q, P = Q-1, \text{or} Q = P-1$. This relation is made for reducing the test iterations which can be seen in the later part of this section. In spite of the fact, one can feel free to construct the $\mathbf{M}$. Each node $\aleph_i; 1 \leq i \leq \mathbf{N}$ of the network may be considered in row/column major order in the $\mathbf{M}$. Thus, the corner nodes on the network can consider the elements $\mathbf{M_{11}}, \mathbf{M_{1p}}, \mathbf{M_{q1}}, \mathbf{M_{pq}}$ in the $\mathbf{M}$. For instance, one can represent a reduced mesh (Figure 4.4c) into a 4×3 matrix as provided in Table 4.4. Here, nodes $\{\aleph_1, \aleph_3, \aleph_{10}, \aleph_{12}\}$ can be taken as corner nodes for initiation of the test algorithm. Similarly, matrix representation of

an octagon network (Figure 4.4d) is provided in Table 4.5. As mentioned, one can represent an NoC into different matrices. Here three ways of matrix representation of the octagon network are seen in Tables 4.5a, 4.5b, and 4.5c, respectively. Nodes $\{\aleph_1, \aleph_4, \aleph_5, \aleph_7\}$ in Table 4.5a and $\{\aleph_1, \aleph_3, \aleph_7, -\}$ in Table 4.5b are the possible corner nodes found subsequently on the tabular representation of the octagon network. Note that every node in the octagon network is a border node that has three adjacent neighbor nodes. So, $\mathbf{M}$ can be constructed as shown in Table 4.5c and consider $\{\aleph_1, \aleph_3, \aleph_5, \aleph_7\}$ as the set of corner nodes.

4.5.2 The 4-Corner Principle

The NoC architectures provide the facilities of parallelism, the heterogeneity that allow execution of multiple dissimilar applications simultaneously. Taking these advantages, the test algorithm is considered in the on-line mode where a channel subset undergoes the test of open faults while rest of the network may complete any application. However, an incoming application packet has to wait at a router which is busy in testing its channels and needs to forward the packet on a channel under test. It is the job of the arbiter in the logic block of the router (RLB) to distinguish a test and an incoming application packet followed by halting the application packet at the router as long as the intended forwarding channel is in the test mode. Consequently, this tentative halt may leverage the network performance at least in terms of latency and energy consumption. The performance degradation is additionally enhanced while the application packet reaches the destination though multiple test iterations. It necessitates the reduction of the number of test iterations and is done through the scheduling of nodes driven by the 4-corner principle.

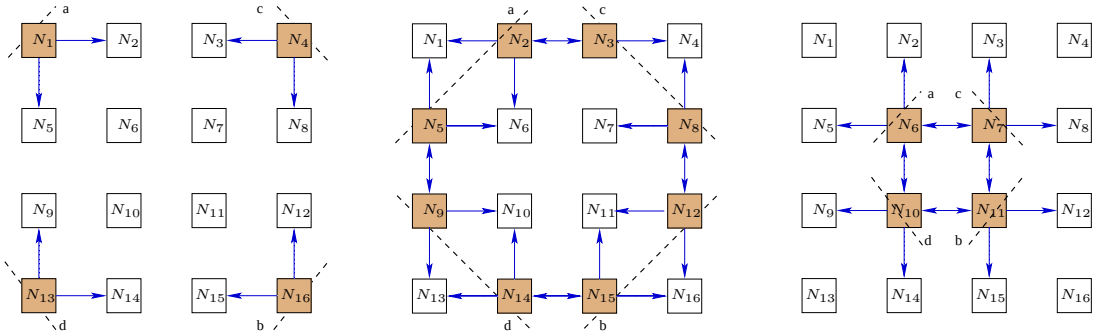
The test algorithm execution is governed by the 4-Corner principle. The principle is stated as a variant of testing the channels for open (as well as other) faults concurrently from four directions in an NoC topology. The objective is to complete testing of NoC channels speedily by initiating the test algorithm with concurrent activation of multiple nodes located at four corners. One can assume the execution of the test strategy based on the 4-corner principle is like a multi-threaded breadth-first search (BFS) traversal [170] that starts from four-cornered nodes of a network. Execution of the test mechanism starts from a corner node, proceeds by exploring the nodes in next level, and finishes at some inner node. In case of a network of higher size, the execution once finished at the corner nodes is shifted to their neighbor nodes where the same job is performed in the next test iteration. The procedure is continued till there is an untested channel. Note that during test execution at a level $\mathbf{L}$, TPG of a node at the level transmits test packet to a TRA of a node at the $\mathbf{L-1}$ in addition to sending the packet to nodes in the next level $\mathbf{L+1}$. However, in order to prevent further execution of the test algorithm by nodes at the $\mathbf{L-1}$ that have tested its channels, a status bit is maintained at the nodes. The corner nodes on the network (either a $\mathbf{P \times Q}$ grid or

the network with **N** nodes) are selected by its matrix representation as shown earlier. One may assume these nodes are in four conventional directions: either at the north (N), east (E), south (S), and west (W) or at north-east (N-E), south-east (S-E), south-west (S-W), and north-west (N-W) in an orthogonal place. For example, nodes $\aleph_1, \aleph_3, \aleph_5, \aleph_7$ in the octagon network (Figure 4.4d) fit the former case, i.e., these nodes act as corner nodes in N, E, S, W directions. The nodes $\aleph_1, \aleph_4, \aleph_{13}, \aleph_{16}$ in the 4×4 mesh network (Figure 4.4a) are treated to be located at the corners in N-W, N-E, S-W, and S-E direction, respectively. Note that corner nodes determined with reference to matrix **M** representation of a network are always assumed in the second directional case due to node locations $\mathbf{M_{11}}, \mathbf{M_{1p}}, \mathbf{M_{q1}}, \mathbf{M_{pq}}$ in the **M**.

Although the test applications from the four directions are executed in parallel, yet a synchronization among these executions must be maintained so that no channel undergoes the test twice. It is possible while the channels are bidirectional and the nodes that execute the test mechanism are nearest. For example, if the channels in the 2×2 network are bi-directional then interswitch channels get tested twice. This can be tackled by initiating the test execution in the order of the nodes $\mathbf{M_{11}}, \mathbf{M_{pq}}, \mathbf{M_{1p}}, \mathbf{M_{q1}}$ in the directions: N-W, S-E, N-E, and S-W, respectively. The execution order is labeled as $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$, respectively. This sequence is called as the *node activation sequence* in a test iteration (Tit). The sequence of test executions from a corner node can be demonstrated with a tree that is referred here as an *execution tree*. The root of the tree represents a corner node for initiating the test algorithm while nodes at other levels correspond to the progress of the channel testing in that direction. Since four instances of the test application are carried out simultaneously, the completion of channel testing on a network using the 4-corner principle may be represented by a forest. In other words, the whole execution is represented by a forest called here *a 4-corner forest*. The forest perpetually consists of four execution trees on every network.

The 4-corner principle depends on four driving factors at a router besides the proposed concurrent activation. The factors are the number of neighbor routers, active I/O ports of a router, pair of test packet generator (TPG) and analyzer (TRA), and mode of communication (simplex/duplex, unicast/multicast). The pair $\langle TPG, TRA \rangle$ is assumed at every router and core for minimum performance overhead. Simplex mode is used in unidirectional channels while the duplex mode is used in bi-directional channels. In addition, both multicast and unicast mode of test packet transmission is accomplished by the TPG of a router and core to their neighbors, respectively.

The proposed test algorithm begins illustration on the 4×4 mesh network shown in Figure 4.4a. The test application on this network is demonstrated in Figure 4.5. As per the 4-corner principle, the test algorithm is initiated from corner nodes $\aleph_1, \aleph_{16}, \aleph_4, \aleph_{13}$ located at directions N-W, S-E, N-E, and S-W, respectively (Figure 4.5a). Consider the test execution at the node $\aleph_1$. The TPG in router R_1 multicasts the test packet to routers R_2, R_5 and its core C_1 for detecting open faults on the channels $(R_1, R_2), (R_1, R_5), (R_1, C_1)$. At the same

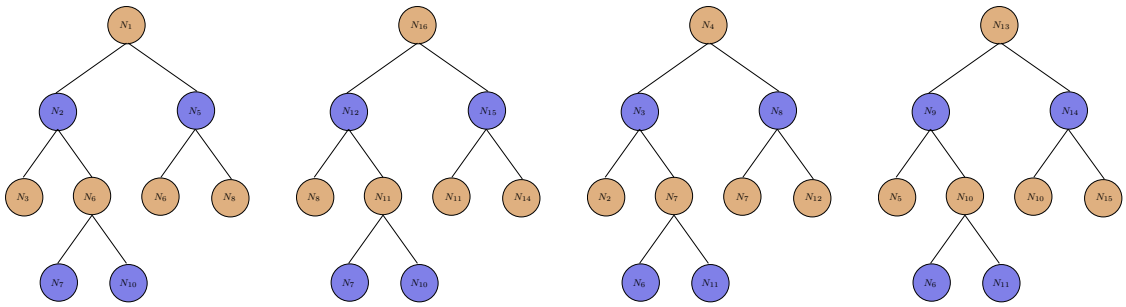


(a) Test execution at the first iteration.

(b) Test execution at the second iteration.

(c) Test execution at the third iteration.

Figure 4.5: Execution of the test mechanism driven by the 4-corner principle in a 4×4 network.



(a) The execution tree rooted at node N_1.

(b) The execution tree rooted node N_{16}.

(c) The execution tree rooted at node N_4.

(d) The execution tree rooted at node N_{13}.

Figure 4.6: The 4-corner forest on execution of the test mechanism initiated from corner nodes of the 4×4 network.

time, TPG in the core C_1 unicasts the similar test packet to the router R_1 for detection of the fault on the channel (C_1, R_1). The received packet at both cases is analyzed at the receiver TRAs which ensure the health status of the channel wires, identify faulty wires, and report channel errors on the basis of faulty wire types in the channels. Similarly, the test execution is accomplished by other corner nodes $\aleph_{16}, \aleph_4, \aleph_{13}$ concurrently with the $\aleph_1$ to detect and diagnose open faults in the channels $(R_{16}, R_{12}), (R_{16}, R_{15}), (R_{16}, C_{16}), (C_{16}, R_{16})$ from the $\aleph_{16}$, $(R_4, R_3), (R_4, R_8), (R_4, C_4), (C_4, R_4)$ from the $\aleph_4$, $(R_{13}, R_9), (R_{13}, R_{14}), (R_{13}, C_{13}), (C_{13}, R_{13})$ from the $\aleph_{13}$. The test execution in this session is termed as test iteration and said to be completed at first level, i.e., $\mathbf{L = 1}$. The next level test executions or simply next test iterations are shown in Figures 4.5b and 4.5c, respectively to check open faults in rest of the channels in the network. Figures 4.6a, 4.6b, 4.6c, and 4.6d are the execution trees rooted at the nodes $\aleph_1, \aleph_{16}, \aleph_4, \aleph_{13}$, respectively in the 4-corner forest (Figure 4.6) of the 4×4 mesh network. One may in another way illustrate the application of the test algorithm using the 4-corner principle using the concept of threading on a network. The 4-corner principle first forks four children threads. These threads are assumed as first level threads. Each thread

executes an instance of the proposed testing technique. Each instance starts its execution at the corner nodes. Once the testing of the channels for open faults from these nodes is over, each child thread forks to second level threads which are executed at the neighbor nodes of the corner nodes. The process is continued till there is an untested channel.

4.5.3 Computation of Test Iterations and Time

The proposed test algorithm handles the testing of communication channels in NoC architectures using four concurrent flows in a single session. Also, it has been noted that a test packet is delivered and analyzed using a predetermined fault-free TM block in order to ensure the health status of channel wires such that these wires do not lead to application packet loss. The test cost in terms of the time needed to test all channels is one of the major challenges for a test algorithm. Additionally, the overall test time in the on-line mode influences the system performance, such as latency degradation. On the contrary, an inefficient test mechanism may deliver higher test cost and many related issues like performance overhead. To reduce these issues, the proposed test solution has attempted (1) defining a test packet flow in the test algorithm and (2) considering an algorithm based test scheduling technique. Reduction of the test time by the defined packet flow is done in two ways on the basis of packet routing by TMs of a node. The first way states the transmission of the test packet through *unicast* routing. The test packet is sent from one source (core-TPG) known as the test source to only one destination (router-TRA) known as the test destination. The core's outgoing channel, in this case, is under test. Most used NoC architectures are regular by nature that conforms each basic component, e.g., channels of the architectures have the identical configuration. In the presence of channels of the same type, the test packet can be routed simultaneously from the single test source (router-TPG) to multiple test destinations (TRAs in neighbor routers and the TRA in own core). The second way thus, states the transmission of the test packet through *multicast* routing. In this case, router's outgoing channels undergo the test. The test execution in this manner ensures all outgoing channels of a node get tested in parallel. Therefore, T_{ch} defined in Table 4.6 is same for these channels and equated in Equation 4.5. The parameter depends on a piece of time $\mathbf{t} \in \mathbf{T}$ where $\mathbf{T} = \{T_{tpg}, T_{tpo}, T_{tpt}, T_{tra}\}$ consists of different time pieces incurred by a TM.

$$T_{ch} = \sum_{t \in T} t \tag{4.5}$$

More efficient testing of outgoing channels of a node in parallel using unicasting and multicasting is one footstep to reducing the overall test time $T_{n/w}$ (defined in Table 4.6). In addition to unicasting and multicasting scenarios, the proposed test scheduling as the next footstep addresses the test time optimization problem in an NoC-based communication system. Here, the nodes selected as per the 4-corner principle execute the test algorithm

Table 4.6: Acronyms used in determining the test cost for open faults.

Acronym	Definition
T_{tpg}	The amount of time taken by a TPG to derive the test data for open faults in channels.
T_{tpo}	The amount of time taken by a TPG to packetize the raw test data for open faults in channels.
T_{tpt}	The amount of time taken by a TPG to deliver the test data for open faults in channels.
T_{tra}	The amount of time taken by a TRA to analyze the test responses for open faults in channels.
T_{ch}	The amount of time taken by the test algorithm to detect an open fault on wires of a channel.
T_{it}	The amount of test time taken by the test algorithm on a test iteration to detect an open fault on wires of a channel.
#Tits	The number of test iterations needed to complete the testing of open faults in channels of an NoC.
$T_{n/w}$	The amount of overall test time incurded due to testing open faults in channels of an NoC.

concurrently. This indicates that T_{it} defined in Table 4.6 is equal to T_{ch} as shown in Equation 4.6.

$$T_{it} = T_{ch} \tag{4.6}$$

The test execution flow from the nodes in a test iteration is shifted to the nodes in the next test iteration. The nodes are periodically selected on the basis of their corner locations. Therefore, the four concurrent test flows are completed by a finite value of #Tits defined in Table 4.6. The #Tits depends on two influential factors- network size and network type. In case of $P \times Q$ grid types networks, such as mesh and torus, #Tits can be determined as equated in Equation 4.7. One additional iteration needs to be executed when $P = Q$ and P, Q are odd numbers. If the networks are not grid-like structures, one can, however, represent the nodes of the networks as a $P \times Q$ matrix as discussed in Section 4.5.1. Thus, it is seen that all channels of a $P \times Q$ network or the network with $\mathbf{N}$ nodes undergo the testing for #Tits iterations. Alternatively, one can consider the maximum distance $\mathbf{D_{max}}$ from root to a leaf node at any of the trees in the 4-corner forest of a network, as the #Tits. Thus, the overall test time $T_{n/w}$ for the network follows the Equation 4.8

$$\#Tits = \begin{cases} \lceil (P+Q-2)/2 \rceil, & \text{an NoC has } P \times Q \text{ nodes.} \\ \mathbf{D_{max}}, & \text{an NoC has } \mathbf{N} \text{ nodes.} \end{cases} \tag{4.7}$$

$$T_{n/w} = T_{it} \times \#Tits \tag{4.8}$$

4.6 Simulation Results

The goal of this section is to demonstrate the effectiveness of the proposed on-line test solution on a large set of NoC architectures. The effectiveness is studied by the application of the proposed solution on a set of both conventional and unconventional networks. Many $P \times Q$ mesh NoCs and an octagon NoC are taken as the first and second category of networks, respectively. In the current as well as in the next section, the proposed test solution is applied to the mesh networks. Application of this test solution on octagon network is studied in Section 4.8. The architectural characteristics of the mesh networks included in this work are described in Table 4.7. The characteristics are described by the parameters- network size, #R (number of routers), #C (number of cores), #Ch (total channels), #W (total channel-wires), #R-R (total interswitch channels), and #R-C (total local channels). Note that the architectural features of the mesh networks are currently considered for *n=16-bit* channels. In a 16-bit channel, 2-bit for control wires, 12 bit for data-wires, and rest 2-bit for handshake wires are assigned.

Table 4.7: Characteristics of 16-bit $M \times N$ NoCs.

Size	#R	#C	#Ch	#W	#R-R	#R-C
2×2	4	4	16	256	8	8
3×2	6	6	26	416	14	12
3×3	9	9	42	672	24	18
4×3	12	12	58	928	34	24
4×4	16	16	80	1280	48	32
5×4	20	20	102	1632	62	40
5×5	25	25	130	2080	80	50
6×5	30	30	158	2528	98	60
6×6	36	36	192	3072	120	72
7×6	42	42	226	3616	142	84
7×7	49	49	266	4256	168	98
8×7	56	56	306	4896	194	112
8×8	64	64	352	5632	224	128

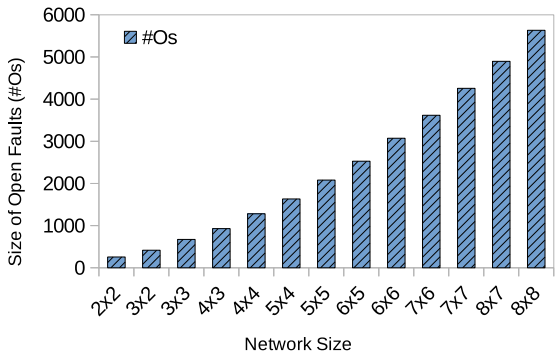


Figure 4.7: The size of open faults injected in the 16-bit networks.

In section 4.4, it is mentioned that the effects of single and multiple open faults on a channel wire are same. Therefore, the single instance of open faults on each wire of a channel is enough for the evaluation of the proposed test solution on a network. Considering the Equation 4.2, the size of the open faults O_{NoC} (or #Os) in an NoC that can undergo the testing is provided in Figure 4.7. The evaluation of the proposed test scheme on the detection and diagnosis of these injected open faults in channels of an NoC is carried out with respect to four commonly used quality characteristics: (a) silicon hardware area incurred on the implementation of the test mechanism, (b) the overall test time incurred by the test scheme, (c) coverage metric that includes both test and fault coverages, and (d) network behavior in presence of faulty of channels which are affected by open faults. These quality parameters now illustrated with respect to a 16-bit 4×4 mesh network. These parameters on a network with higher channel width and size are evaluated in Section 4.7.

Table 4.8: Area overhead incurred by the proposed TM blocks for 16-bit channels.

H/w Blocks	#GCs	RASoC (%)	Hermes (%)	Xpipe (%)	Ætheral (%)	Vicis (%)	Core (%)
C-TPG	45	2.67	2.66	2.95	0.17	0.22	0.04–0.67
C-TRA	32	1.9	1.89	2.10	0.12	0.16	0.03–0.48
C-TM	77	4.56	4.56	5.05	0.29	0.38	0.07–1.14
R-TPG	70	4.15	4.14	4.59	0.26	0.34	–
R-TRA	50	2.96	2.96	3.28	0.19	0.24	–
R-TM	120	7.11	7.10	7.87	0.44	0.59	–

4.6.1 Silicon Area Evaluation

The proposed test algorithm for channel's open faults is designed at a node in order to address them on the node's outgoing channels on a test iteration. The TM blocks placed at the nodes

are implemented to execute the test algorithm that detects and diagnoses the modeled open faults in channels. In the core and the router of a node, the TPG blocks as the test sources apply and deliver the test packet on the outgoing channels of the node. The test packet includes required test data of the A1 type. The TRA blocks as the test destinations over the channels under test analyze the test responses (received test data). One can implement the test algorithm in terms of an experimental setup in the following way. Place the TPGs as the test sources on the left side and the TRAs as the test destinations at the right side of the channels modeled with open faults. This experimental setup is implemented by the Xilinx 10.1 ISE Design Suite. Thus, the test cost from the point of view of silicon area overhead is basically the hardware area taken by these modules in a core and router where the test algorithm needs to be implemented. The Design Suite is used to synthesize the area of these hardware blocks in gate counts (GCs). Now, the test packet contains the A1 sequence when open faults are modeled in the form of stuck-at-0 faults. However, if one is free to consider an open fault in terms of stuck-at-1 fault, then the test packet should contain A0 sequence as the required test data. Thus, TM blocks must be flexible to derive, deliver, and analyze both A0 and A1 test sequences. Table 4.8 provides the synthesis result for the TM blocks designed for 16-bit channels. It is observed that a TM block occupies nearly 7.10–7.87% area in a five port router. The area overhead is observed for different routers. Following general-purpose routers, namely RASoC [65], Hermes [2], and Xpipe [66] are considered here to measure the TM-area on these routers. Further, TM blocks in cores only handle open faults on its outgoing local channel. Therefore, the area of a core-TM is comparatively quite small than a router-TM. Also, the area of a core is nearly 4 to 5 times larger than a router. Therefore, the area occupied by a core-TM is nearly 1.14%. Thus, TM-areas are very small and can be well accepted to address open faults in channels. There are some application-specific routers, such as Ætheral [70] and Vicis [184] whose sizes are significantly high. The area overhead due to the TM blocks in these routers are nearly 0.44–0.59%. Subsequently, these blocks take nearly 0.07% of the cores designed for such large-scale complex applications. Therefore, TM-area is negligible in these cases.

4.6.2 Test Iteration and Time Evaluation

The evaluation of the test time taken by the proposed test algorithm on a test iteration depends on the operations of the TM blocks for open faults. A test application using the TMs starts by the test data generation and finishes by test response analysis. Now the operations: test data generation, organization, and transportation are performed by the TPGs at the test sources. The remaining operation called test response analysis is done in the form of fault detection and diagnosis, and channel error announcement. A TPG generates only one test sequence which is here A1 type and takes only one clock cycle. In the next clock, the

hardware unit derives the header and trailer information for the test packet. Thus, the TPG takes $T_{tpg} = 2$ clocks in deriving the raw test set and required packet header and trailer information. Next, to the generation of these data, they are packeted at the cost of one clock, i.e., $T_{tpo} = 1$ clock. As soon as the test packet is available, the TPG starts shifting the test packet as packet flits. In other words, the packet latency that defines the time needed by the packet to travel a channel is the T_{tpt}. Each flit of the test packet is transmitted per clock. Additionally, a channel under test carries only this packet and any application packet is prevented from routing on the channel. Thus, the test packet with three flits charges $T_{tpt} = 3$ clocks as the packet latency. At the TRA side, on receipt of the test response, the FDM unit detects an open fault if any bit in the response flit contains a "0" or a "X" for a channel wire. To ensure that the wire is actually affected by the open fault, the FDM diagnoses the response flit using a logic operation as provided in Equation 4.3 or 4.4. This operation can be conducted within a single clock. Further, the diagnosis information is fed to the TSG unit that advises the possible channel errors based on the faulty wires. This error declaration takes one clock. Thus, the operations by the TRA are completed by $T_{tra} = 2$ clocks. Considering Equation 4.5, the amount of test time needed by the test algorithm for open faults in a 16-bit channel is $T_{ch} = 8$ clocks. The unicasting and multicasting mechanisms are considered to conduct a concurrent test of all outgoing channels of a node. These channels are then tested by 8 clocks. Now, NoC architectures provide inherent parallelism in transporting the data from multiple nodes. Using this advantage, multiple nodes are allowed to initiate the test algorithm. Assuming Equation 4.6, the algorithm delivers that open faults in the channels under consideration in an iteration can be checked by $T_{it} = T_{ch} = 8$ clocks. The overall test time $T_{n/w}$ (Equation 4.8), however, depends on the number of test iterations #Tits (Equation 4.7) on a network. As seen in Figure 4.8 that the open faults in channels of the 4×4 mesh NoC can be covered by #Tits=3 iterations. Subsequently, the test cost in the form of overall test time for this network is $T_{n/w} = 8 \times 3 = 24$ clocks only (Figure 4.9).

4.6.3 Test and Fault Coverage Evaluation

The effectiveness of the proposed test methodology from the point of view of both test and fault coverage metrics is evaluated by conducting the fault simulations on the selected 16-bit mesh networks. The fault simulations are run for the targeted fault model, namely stuck-open using the selected test pattern set. A subset of channels from the nodes explored as per the 4-corner rule is placed in the test mode. On the testing of these channels, the test coverage metric commonly known as link coverage metric (LCM) is achieved. Continuing the test iteration initiating from the corner nodes one can achieve that all channels have been checked for open faults. Thus, the overall link coverage metric termed as the cumulative LCM (CLCM) is 100%. Figure 4.10 provides a summary of the LCM evaluation on an iteration for a

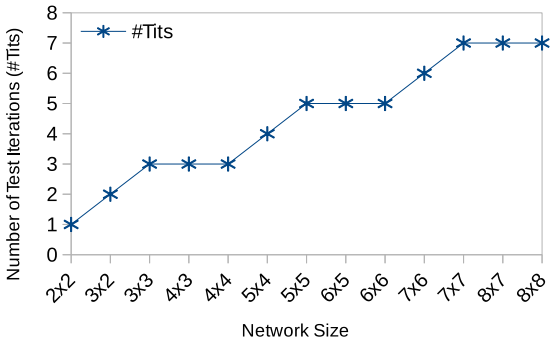


Figure 4.8: The number of test iterations required to detect open faults in the $P \times Q$ networks.

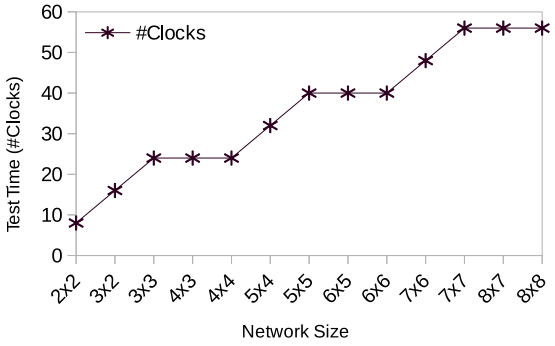


Figure 4.9: The test time incurred by the 4-corner driven test mechanism in the 16-bit $P \times Q$ networks.

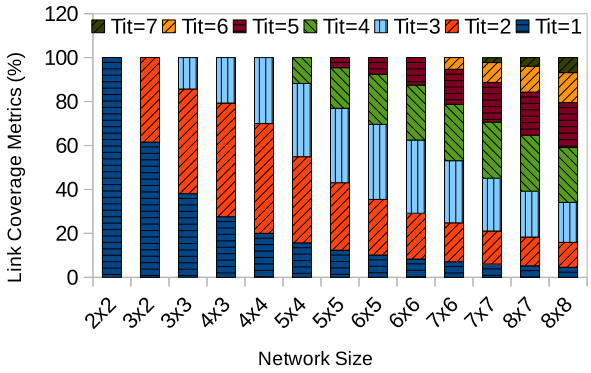


Figure 4.10: The LCM achieved on test executions in the 16-bit $P \times Q$ networks.

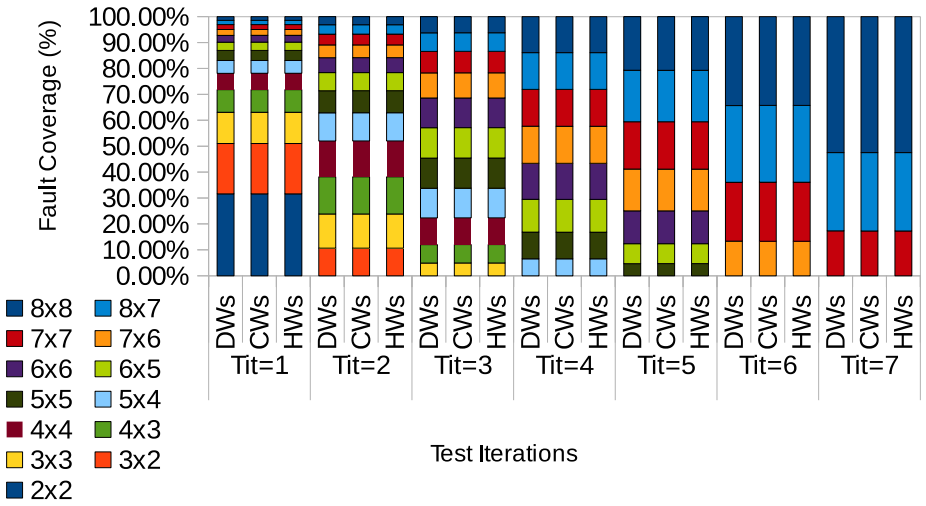


Figure 4.11: Size of open faults detected in a test iteration on a 16-bit $P \times Q$ network.

16-bit 4×4 mesh network. Similar to the LCM, fault simulations determine another coverage metric called the fault coverage metric (FCM) that defines the size of open faults covered on testing. During a fault simulation on a subset of channels for open faults, the simulation is done in three phases depending on the type of channel wires. In the first phase, the faults on the data-wires (DWs) are simulated using the test data placed in the payload field of the test packet. Obviously, the fault coverage on the DWs only is insufficient because the simulation results in only 75% fault coverage. However, the metric can be improved by extending the current fault simulation on the remaining channel wires: control-wires (CWs) and handshake-wires (HWs). It needs to enrich the test set. In order to simulate open faults in the CWs, one must include A1 sequence in the packet header and trailer information. The simulation improves the FCM by 12.5%, i.e., the cumulative FCM (CFCM) on testing the DWs and CWs for open faults is 87.5%. Similarly, handshake information of the test packet should be accommodated with the additional A1 sequence whence one simulates open faults in the HWs. Thus, another 12.5% FCM is achieved. In this sequencing order of fault simulations, the CFCM can be maximized to 100%. The FCM in terms of the number of detected open faults on the DWs, CWs, and HWs on the 16-bit 4×4 mesh network is demonstrated in Figure 4.11. Thus, the fault simulations achieve 100% coverage metrics by finite test iterations in a single test session.

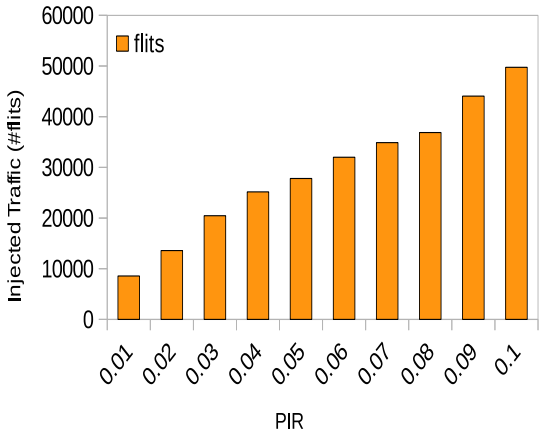
As mentioned earlier, the TRAs on the test response analysis report two types of channel errors: payload, and dropping as the realization of partial and full packet loss, respectively. From the fault simulations, it can be inferred that 87.5% payload error (PE) is possible due to open faults in data and handshake wires. The rest 12.5% is considered as the dropping error due to the faults in control-wires. This error is treated here as the timeout error (TE). Note that a control wire that carries the packet header and trailer equally contributes to the packet timeout. Thus, faulty CWs that carry the packet header information and the packet trailer information result to 6.25% and 6.25% timeout errors, respectively. The CFCM can also be studied from the channel errors thus evaluated.

4.6.4 On-Line Performance Evaluation

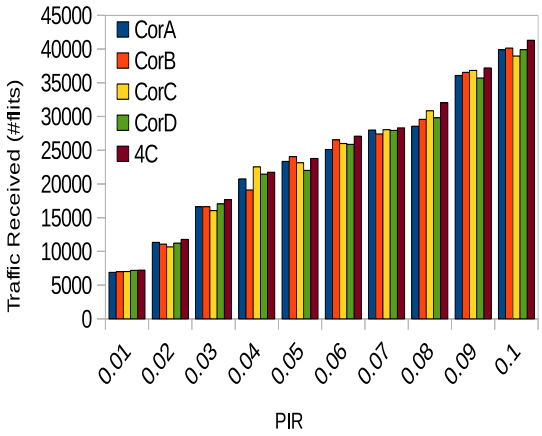
The proposed test mechanism driven by the 4-corner principle rapidly checks the health status of the channels of an NoC that as a result maintains the system reliability. Further, related issues, such as the yield of the system can be improved in the off-line by exploiting the report on channel's health status. Other issues, such as performance degradation due to open faults in channels of a routing path between the source and destination nodes can be prevented by employing a fault-tolerant routing approach that exercises the status report. On the contrary, if a fault-tolerant routing is not employed and the traffic of a regular application is allowed on the faulty channels, the NoCs show abnormal performance due to drastic packet loss.

To evaluate the severe effect of the channel's open faults on the network performance, the proposed test scheme is run on the 16-bit 4×4 mesh as the reference network. The rigorous system simulations on the network are built upon using a cycle accurate NoC simulator popularly known as Noxim [22]. The performance of an NoC-based system has major concerns about the throughput, latency, and power consumption. The simulator has been utilized to compute these mostly used metrics. To assess the system performance in terms of these metrics, the simulations are performed five times. Each of the first four turns is dedicated to an evaluation when the channels under test are selected from a corner node only. The last turn simulation is dedicated to the evaluation when the channels at the four corners are concurrently put in the test mode. In all the cases, a large number of A1 sequences (synthetic traffic) as the regular application data are injected on the network. The size of this traffic is determined by the packet injection rate (PIR) and uniform random traffic distribution. The PIR value is set in the range from 0.01 to 0.1 for the current network. At each PIR value, the simulation is run for 10000 cycles including 10% of the cycles used in simulation startup and statistic collection. The wormhole switching technique segments the large A1 traffic chunk into smaller application packets. Each packet that contains three flits is routed by the XY routing algorithm that first routes the packet in the X direction followed by in the Y direction before reaching the destination node.

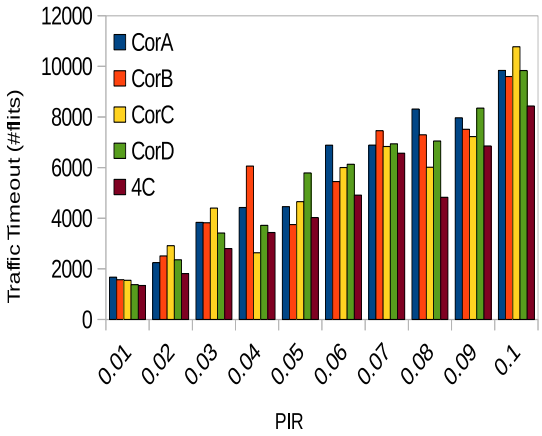
Figure 4.12 illustrates the on-line evaluation of the proposed test solution on the 16-bit 4×4 mesh network. For the evaluation, whether a subset of channels put in the test mode from a corner node or all corner nodes, the same size of traffic is injected in each case. Figure 4.12a provides the size of injected synthetic traffic in terms of packet flits in the network. Consequently, the size of received and dropped traffic are observed in Figures 4.12b, and 4.12c, respectively. The size of lost (dropped) packet flits heavily affect the size of received traffic, even if there is no faulty channel in the network. These lost flits are due to the normal packet timeout error. The error grows for different reasons, e.g., traffic congestion, unavailability of network resources, etc. In the normal mode, it is found that the packet timeout is nearly 4–9%. However, the error is enhanced while one or multiple channels infected by open faults appear in a routing path. It is observed that the size of lost packet flits is nearly 16–22% when the channels at particular corner side are iteratively put in the test mode. This dropping is nearly 13–17% when the channels from all corner sides concurrently undergo the testing which proceeds iteratively from one level to another in the network. Correspondingly, the throughput of the reference network is observed as shown in Figure 4.12d. The packet latency is another important performance metric which is heavily affected by the routing distance between sender and receiver nodes. Since the distance varies with a node pair, normally average distance is considered to measure the (average) latency of the packets. Also, the traffic congestion, faulty channels, and consequently dropped traffic are the common factors that influence the packet latency. Figure 4.12e illustrates the latency behavior in the reference



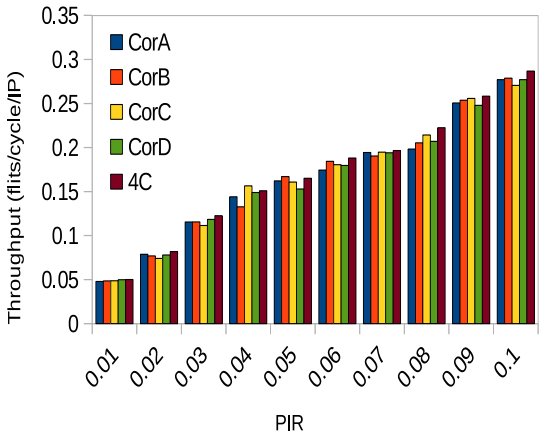
(a) Amount of packet flits injected.



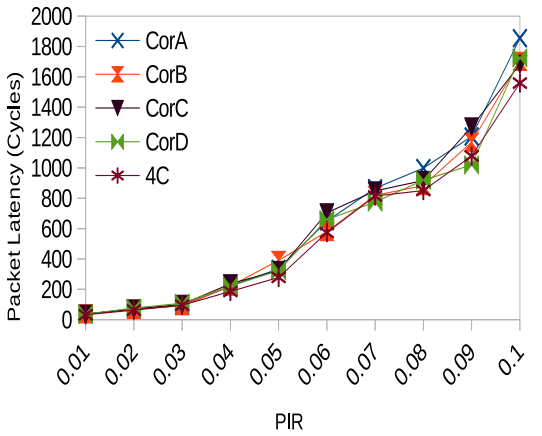
(b) Amount of packet flits received.



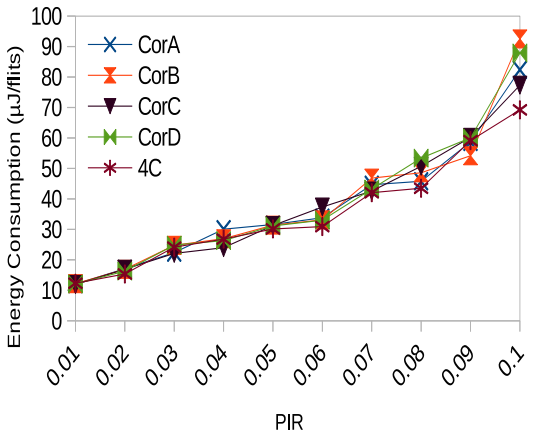
(c) Amount of packet flits lost.



(d) Throughput behavior.



(e) Latency behavior.



(f) Power consumed by a flit.

Figure 4.12: On-line evaluation of the proposed test solution applied at the corner nodes separately and concurrently on the the 16-bit 4×4 mesh NoC.

network. Subsequently, Figure 4.12f gives the amount of energy consumed by a packet flit in order to reach the destination. From the above analysis, a conclusion can be drawn that improved performance (in terms of throughput, latency, and energy consumption) is achieved when the set of channels from all corner nodes are repeatedly placed in the test mode while another part of the network continues an application.

4.7 Solution Scalability

The size, type, and complexity of applications are continuously growing. An NoC-based system should facilitate the transmission of high size packet flits for such a large scale application. Additionally, the NoC in the system needs to be designed with higher channel width in order to enable the high-performance communication and computation. Thus, selection of the channel width and the flit size plays a decisive role in the overall system performance. However, designing an NoC architecture with higher channel width may incur additional cost. In the case, many applications may consider large-scale NoCs with comparative lower channel width. Therefore, it is the designer's choice to build the NoCs that can satisfy the above considerations either explicitly or inclusively. Consequently, the proposed test scheme should scale with higher channel width and network size.

4.7.1 Scaling with Channel Width

The test behavior of the proposed mechanism should be analyzed under different channel configurations, such as higher channel width. The proposed test scheme in the previous section is illustrated on the same 4×4 mesh network. In this subsection, the evaluation of the scheme is extended on the network but its channel width is now 32-bit instead of 16-bit. Increasing channel width is required to accomplish larger scale regular applications that handle voluminous data, such as multimedia information. Simultaneously, increasing manufacturing faults, such as open faults become concerns about system reliability and performance degradation. The system performance is evaluated following other quality parameters: silicon area, test time, and fault coverage, under the proposed test scheme.

4.7.1.1 Silicon Area Evaluation

The size of open faults in channels of the $P \times Q$ (including 4×4) mesh networks having 32-bit channels is provided in Figure 4.13. The architectural properties of these networks as described in Table 4.7 remain same except #W, the size of channel wires of the NoCs. The #W on these 32-bit networks is now doubled with respect to 16-bit networks. The proposed test algorithm detects open faults in a channel by exercising a test packet that contains the A1 sequence. The size of the test set $|A1|$ linearly increases with the channel width n. In order to handle open faults in a 32-bit channel, the TM whether at core/router of a node

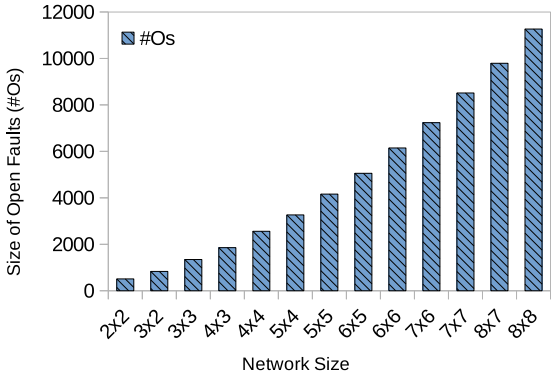


Figure 4.13: The size of open faults injected in the 32-bit networks.

Table 4.9: Area overhead incurred by the proposed TM blocks for 32-bit channels.

H/w Blocks	#GCs	RASoC (%)	Hermes (%)	Xpipe (%)	Ætheral (%)	Vicis (%)	Core (%)
C-TPG	73	4.32	4.32	4.79	0.27	0.36	0.07–1.08
C-TRA	51	3.02	3.02	3.34	0.19	0.25	0.05–0.76
C-TM	124	7.35	7.34	8.13	0.46	0.61	0.12–1.84
R-TPG	105	6.22	6.22	6.89	0.39	0.51	–
R-TRA	69	4.09	4.09	4.52	0.26	0.34	–
R-TM	174	10.31	10.3	11.41	0.64	0.85	–

must derive and analyze a 32-bit A1 sequence. Consequently, the area occupied by the unit is thus increased with additional logic gates and is provided in Table 4.9. The area overhead in terms of gate counts (GCs) is increased by 2–3% as compared to a 16-bit channel resulting in 9.87–10.55% on the RASoC, Hermes, and Xpipe routers. However, the overhead when estimated with respect to the core area is nearly 1.84%.

4.7.1.2 Test Iteration and Time Evaluation

Design of a high bit-width channel on an NoC does not increase the wire length in a channel. This feature does not additionally impose any delay on the T_{tpt} while the test packet flits are shifted from a test source to a test destination. On contrary to little increment on the silicon area overhead incurred by TM blocks, the test time per iteration remains the same as the previous case, i.e., $T_{it} = 8$ clocks are needed to detect open faults in both 16-bit and 32-bit channels. This is due to the fact that additional circuit elements of the TMs designated for 32-bit channels enable these hardware blocks to operate at the same clocks as seen with 16-bit channels. It is noted that the number of test iterations #Tits on an NoC is independent of its channel width and is determined with respect to a scheduling technique. Therefore, the test cost in terms of the overall test time $T_{n/w}$ needed by the proposed scheme remains unchanged

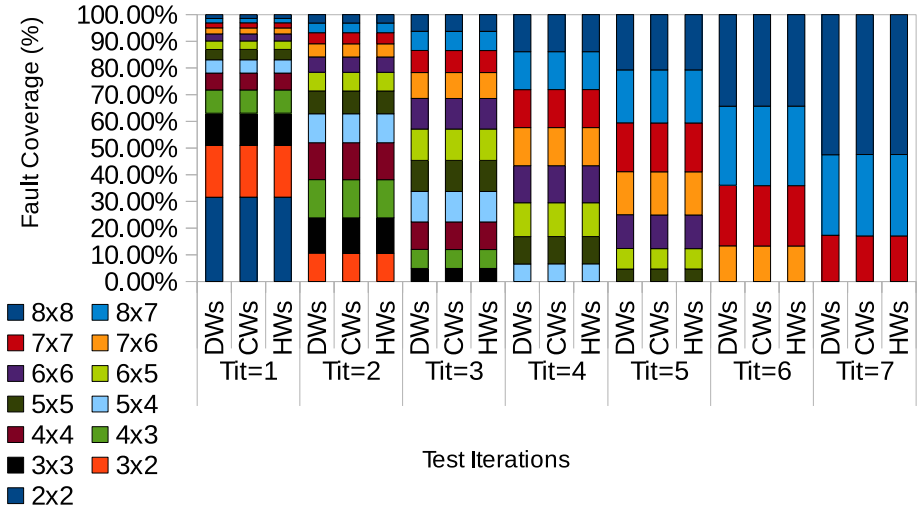
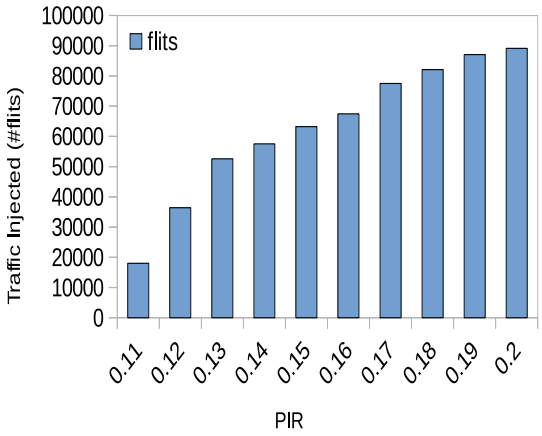


Figure 4.14: Size of open faults detected in a test iteration on the 32-bit NoCs.

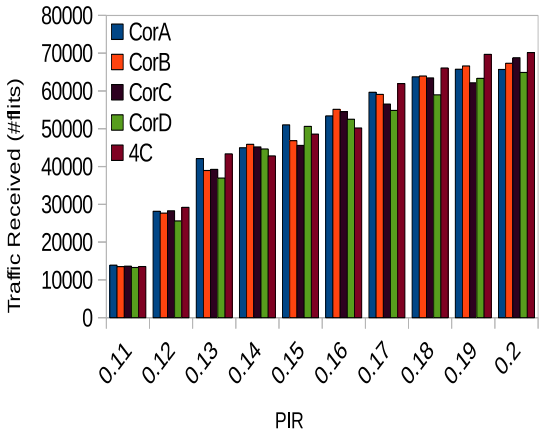
because of the #Tits (Figure 4.8) and the T_{it}. Thus, open faults in the channel of the 32-bit 4×4 mesh network can be detected by 24 clocks (Figure 4.9).

4.7.1.3 Test and Fault Coverage Evaluation

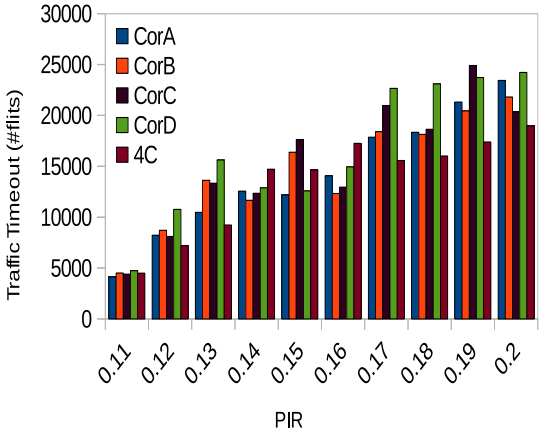
Next, to the evaluation of hardware area and test time, the coverage metric is now investigated on the current reference network, i.e., 32-bit 4×4 mesh NoC. In order to communicate a large chunk of data for an application, each *n-bit* channel must be configured with higher data path, i.e., the size of data wires in the channel must be as many as possible. Like earlier channel configuration, a 32-bit channel is segmented as follows. First 2-bit for control wires and last 2-bit for handshake wires are fixed while rest 28-bits are treated as the data wires. Since the number of channels that undergo the testing in an iteration is equal as seen in the previous section, the LCM of the 32-bit 4×4 and other networks remains same as provided in Figure 4.10. One can draw the conclusion that the LCM is independent of the channel width. Oppositely, the FCM changes with this width because the size of different channel wires is not independent of channel configuration. Figure 4.14 highlights the FCM achieved on the 32-bit 4×4 network via the fault simulation in three phases. In the first phase, the fault simulation on data wires detects 2240 open faults resulting in the FCM of 87.50%. On extending the simulation to control wires in the second phase, another 160 open faults get detected resulting in 6.25% as FCM, i.e., the CFCM is 93.75%. In the final phase, the extended simulation on the handshake wires detects the rest 160 open faults and results in 6.25% FCM that thus makes the CFCM become 100%. Note that the three-phase fault simulation has resulted in 93.75% and 6.25% as payload and timeout errors, respectively.



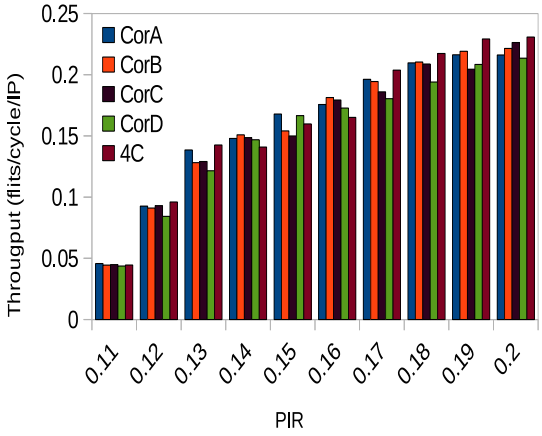
(a) Amount of packet flits injected.



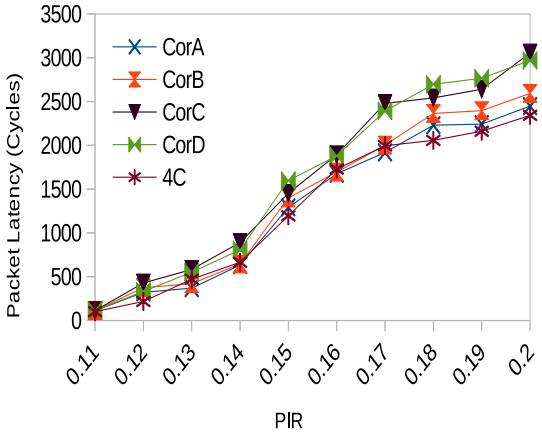
(b) Amount of packet flits received.



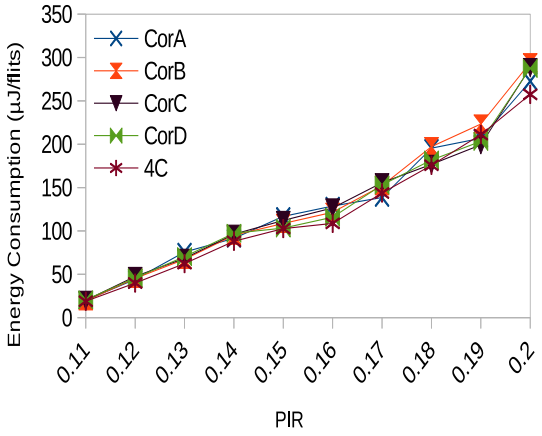
(c) Amount of packet flits lost.



(d) Throughput behavior.



(e) Latency behavior.



(f) Power consumed by a flit.

Figure 4.15: On-line evaluation of the proposed test solution applied at the corner nodes separately and concurrently on the the 32-bit 4×4 mesh NoC.

4.7.1.4 On-Line Performance Evaluation

The performance of the 32-bit 4×4 mesh as an on-chip communication network is characterized by following concerns: throughput, latency, and energy consumption under the proposed scheme. The simulation setup is extended to evaluate the scheme. The PIR value is changed to the range 0.11–0.20 for the consideration of high volume application data. Further, the simulation period is increased from 10000 to 20000 cycles. Figure 4.15 illustrates the behavior of the 32-bit network. The size of application traffic injected on the network is seen in Figure 4.15a. Correspondingly, Figure 4.15b shows the scenario on the size of received traffic in the network. The accepted/received traffic should ideally increase with the injected traffic size, i.e., increasing offered load. However, due to the limitations of faulty channels and/or traffic congestion that realize in terms of the timeout error, many packet flits are dropped in the network. Figure 4.15c depicts this scenario. Subsequently, the concerned performance metrics are seen in Figures 4.15d, 4.15e, and 4.15f for throughput, packet latency, and energy consumption, respectively. The throughput is determined with respect to the received traffic. However, the packet latency and energy consumption increase with the offered load because the contention in the network sufficiently increases.

4.7.2 Scaling with Network Size

So far the discussion, the proposed 4-corner rule-driven test mechanism is illustrated for open faults in channels of the 4×4 mesh network to see the effectiveness of the scheme on this network and exhibit its scalable feature with respect to the channel width. This subsection endeavors to quantify the quality characteristics on the 16-bit 8×8 mesh NoC in order to demonstrate the scheme's scalability with respect to the size of networks. This larger reference network is characterized in Table 4.7. The size of open faults that the proposed test algorithm will take in the testing is provided in Figure 4.7. Iterative execution of the test algorithm from the corner nodes in this network is shown in Figure 4.16. Subsequently, the 4-corner forest is shown in Figure 4.17. In order to properly analyze the quality characteristics on this larger network, the experimental setup is reused. In the setup, a set of channels per iteration on the network is selected followed by putting them in the test mode. The TPGs at one side apply the test packets and TRAs at the other side analyze the corresponding test responses. Since the channel width at present is *n=16-bit*, the hardware area of the TMs is same as provided in Table 4.8. It is found that the test time per iteration incurred by the proposed test mechanism is $T_{it} = 8$ clocks. Considering Figure 4.16 or Figure 4.17, the 8×8 mesh NoC can be tested for channel's open faults in just seven iterations that secures the overall test time $T_{n/w} = 56$ clocks (Figure 4.9). Next, the coverage metrics LCM and FCM are provided in Figure 4.10 and 4.11, respectively. In both cases, the metrics are achieved up to 100%. It is noted that the channel errors in sequencing the fault simulations are similar (i.e., 87.5% as

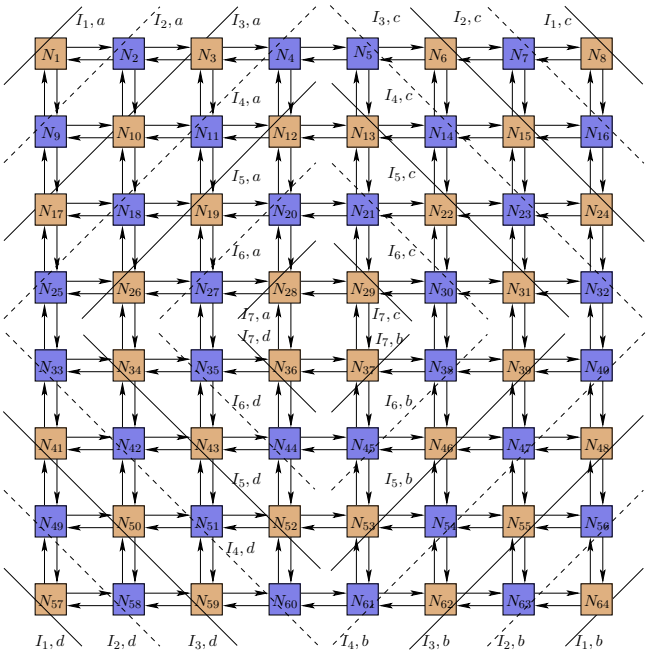
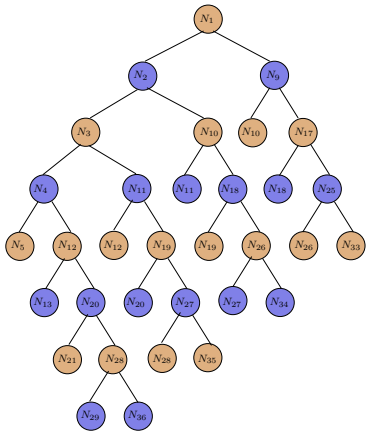
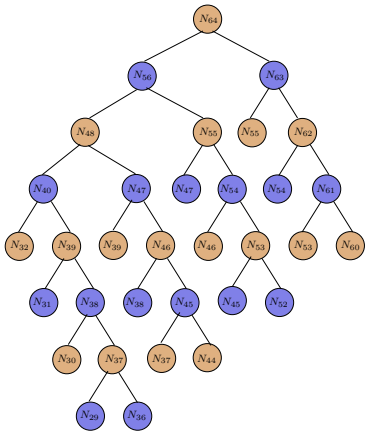


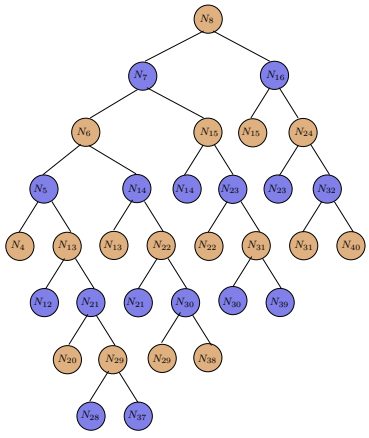
Figure 4.16: Test executions from the corner nodes on the 8×8 network.



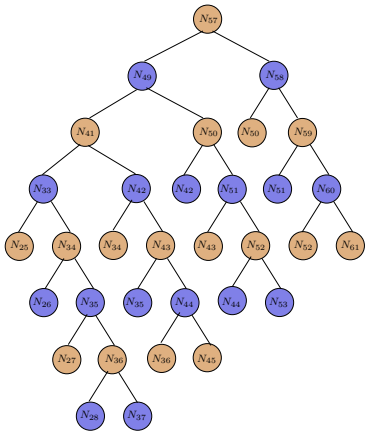
(a) The execution tree rooted at the node N_1.



(b) The execution tree rooted at the node N_{64}.

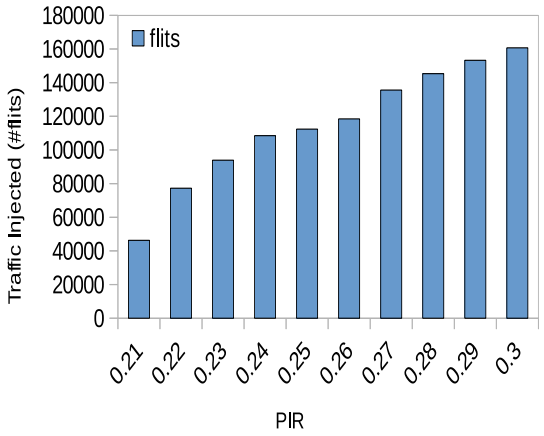


(c) The execution tree rooted at the node N_8.

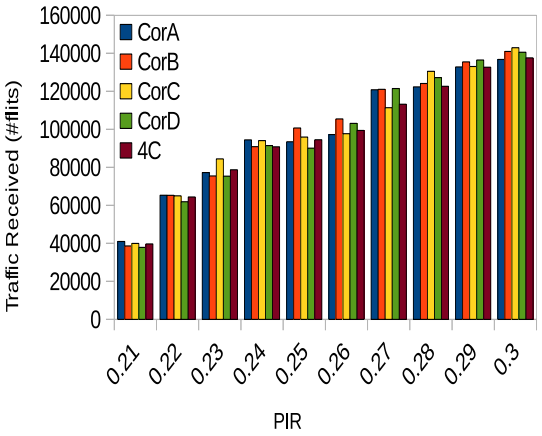


(d) The execution tree rooted at the node N_{57}.

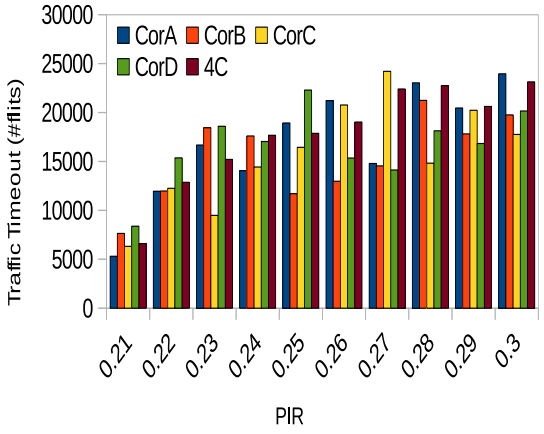
Figure 4.17: The 4-corner forest on the test application initiated from the corner nodes on the 8×8 network.



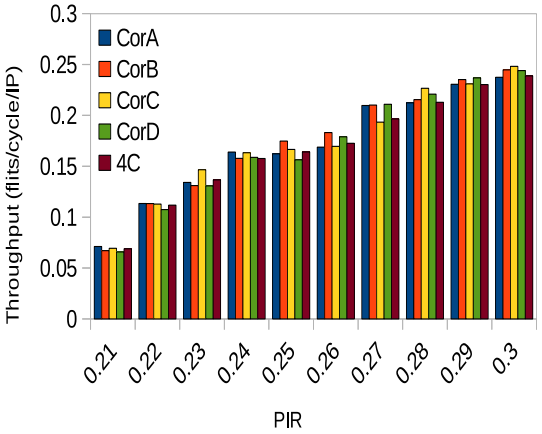
(a) Amount of packet flits injected.



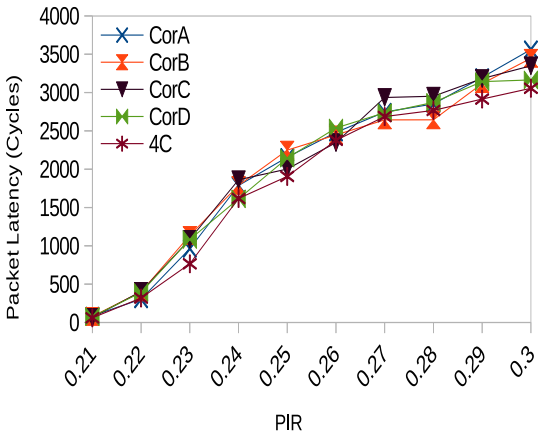
(b) Amount of packet flits received.



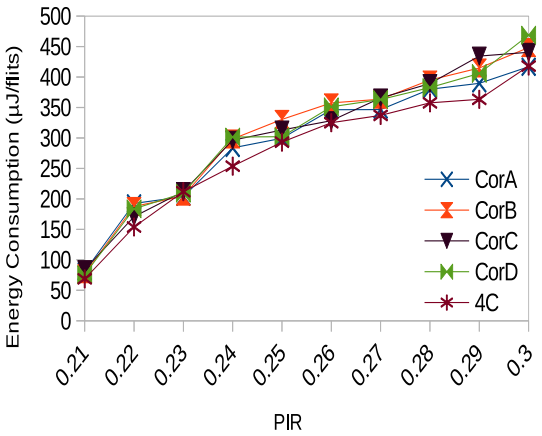
(c) Amount of packet flits lost.



(d) Throughput behavior.



(e) Latency behavior.



(f) Power consumed by a flit.

Figure 4.18: On-line evaluation of the proposed test solution applied at the corner nodes separately and concurrently on the the 16-bit 8×8 mesh NoC.

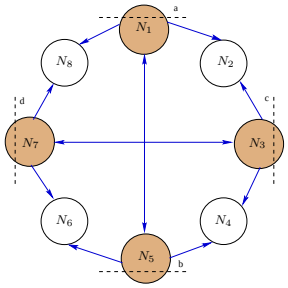
PE and 12.5% as TE) as seen in the 16-bit 4×4 mesh network. One can draw a conclusion that these errors are same with respect to similar channel configurations.

Next, the common performance concerns are considered on the 16-bit 8×8 mesh network under the proposed test methodology. The simulation setup with the Noxim simulator used for the 16-bit 4×4 mesh network is extended. Basic simulation parameters remain same except the PIR value that ranges from 0.21–0.30 in order to observe the network performance at an increased offered load (traffic). Figure 4.18 demonstrates the performance of the network on the application of the proposed scheme. The size of injected traffic in terms of packet flits is shown in Figure 4.18a. Corresponding received and dropped traffics can be seen in Figures 4.18b and 4.18c, respectively. It is seen that the network receives up to 90% of the injected traffic when it is in the normal mode. The rest 10% traffic loss is due to the timeout error which is caused by contention, unavailability of network resources during communication. Note that this packet loss is increased to 17–22% when open faults are experienced in a channel of a routing path. Subsequently, the common performance metrics: throughput, packet latency, and packet energy consumption, are witnessed in Figures 4.18d, 4.18e, and 4.18f, respectively.

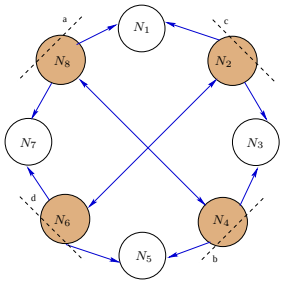
4.8 Solution Portability

Unlike the discussion in previous sections that is pertaining to mostly used conventional mesh-based NoCs, there are many unconventional networks, such as octagon [86, 87] sometimes considered in the literature as well preferred by NoC manufacturers like ST-Microelectronics. Major implementations of these networks are similar. For example, routers, cores used to build a mesh network can construct an octagon network. The only variable is the length of interswitch wires that may incur little additional area overhead. However, one great advantage of these networks is that the communication between nodes can take place with comparatively low hop counts. Another advantage is that these networks enable high performance on the multiprocessor SoCs like as the mesh-based NoCs. To model the efficiency of the proposed test scheme on an unconventional NoC structure, an octagon network (Figure 4.4d), for instance, is employed. Basic architectural design of the octagon consists of 8 routers, 8 connected cores, 24 interswitch channels, and 16 local channels. At the moment, each channel has width *n=16-bit*. Therefore, the test algorithm will test 640 open faults in the channels of the 16-bit octagon network. Now, the common quality parameters: area overhead, test time, fault coverage, and network performance are analyzed on this network.

Every router in the basic octagon NoC has active four I/O ports. However, a router with five I/O ports is used to build a larger octagon network with multiple basic octagon structures. In the larger network, two adjacent octagons share an interswitch channel through the additional I/O port. In this work, the TM units are synthesized with respect to the router

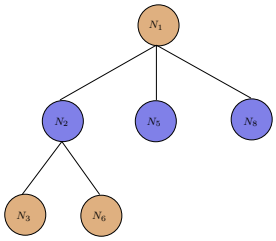


(a) Test execution at the first iteration.

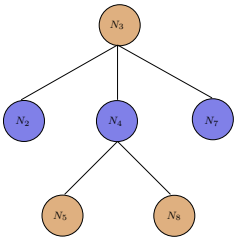


(b) Test execution at the second iteration.

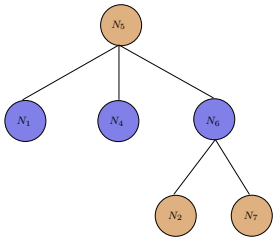
Figure 4.19: The execution of the test mechanism driven by the 4-corner principle on the the octagon network.



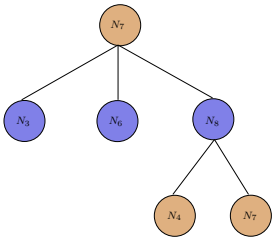
(a) The execution tree rooted at the N_1.



(b) The execution tree rooted at the N_5.



(c) The execution tree rooted at the N_3.



(d) The execution tree rooted at the N_7.

Figure 4.20: The 4-corner forest on execution of the test mechanism initiated from corner nodes of an octagon NoC.

with five I/O ports. The channel width in the referred octagon network is considered 16-bit. Therefore, the hardware area overhead is same as seen on a 16-bit mesh network. One can now draw the conclusion that the TM unit is independent of the network topology and the area overhead per TM in a node remains constant. It is seen that the open faults in channels of the network can be covered by two iterations only. Test executions at these iterations are shown in Figure 4.19. Subsequently, Figure 4.20 illustrates the 4-corner forest on the execution of the proposed test mechanism for the corner nodes in the octagon network. Therefore, considering $T_{it} = 8$ clocks for an iteration, all channels experiencing open faults in the octagon network are tested by $T_{n/w} = 16$ clocks. In the first test iteration, 5 channels from each corner nodes, i.e., 20 channels undergo the testing. Consequently, the LCM is 50%. Rest 20 channels in the network undergo the similar testing in the next iteration. Therefore, CLCM is 100% once this iteration is completed. The channel wires in an iteration undergo the testing in the following sequence: DWs, CWs, and HWs. For instance, the fault simulation on exercising the test packet for DWs detects 240 open faults resulting in the FCM=(240/640)*100=37.5%. Considering the fault simulations on the CWs and HWs in its second and third phases, 40 open faults in each case are detected that results in FCM=6.25%. Thus, the CFCM is 43.75% and 50% at CWs and HWs, respectively. Similar fault coverage is achieved in the next iteration. As each channel is tested by the TMs placed in the nodes, the proposed test mechanism can test multiple channels simultaneously and provide up to 100% coverage metrics.

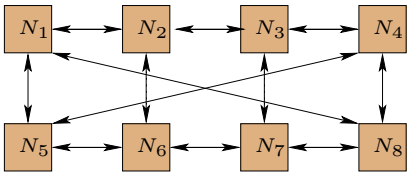
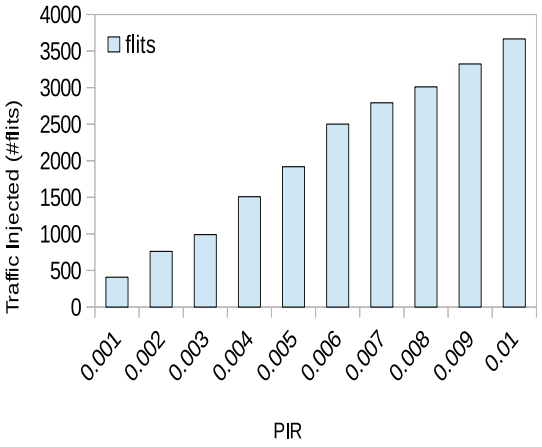


Figure 4.21: Representation of an octagon network into an equivalent 2×4 network.

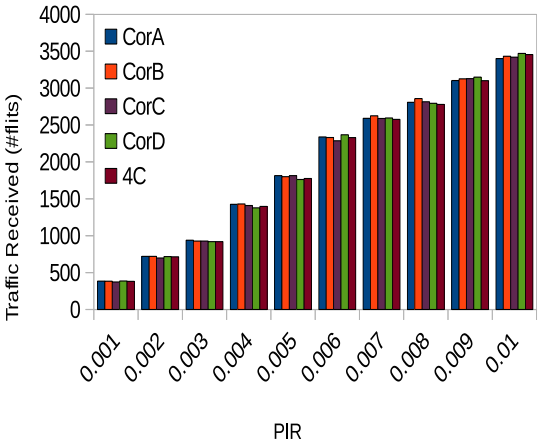
Next, the network performance in terms of the well-defined metrics: throughput, packet latency, and energy consumption are analyzed under the proposed test scheme. The Noxim simulation setup is continued at the PIR=0.001–0.01 to inject the synthetic traffic in the 16-bit octagon network. An equivalent 2×4 network as shown in Figure 4.21 is built in order to evaluate the above-mentioned performance metrics in the octagon network. The network performance under the on-line evaluation of the scheme is demonstrated in Figure 4.22. Size of injected traffic in the network is seen in Figure 4.22a. Correspondingly, the size of received and dropped traffic are seen in Figures 4.22b and 4.22c, respectively. When the channels are not in the test mode and not busy, the regular application can get immediate access to them without any suspension. In this case, it is seen that the size of dropped traffic is nearly 2–3%. However, the dropping rate is increased to 5–8% when there are a contention and a channel experiences open faults. Consequently, the network throughput is observed in Figure 4.22d. Under the assumption that a subset of channels is in the test mode, then many packets of the on-going regular application are halted at the routers of these channels. Subsequently, the packet latency and energy consumption by a packet flit on an average are increased and observed in Figures 4.22e, and 4.22f. Also, it can be noted that such performance degradation depends on the traffic pattern, routing policy, and fault rate in channels.

4.9 Comparison Study

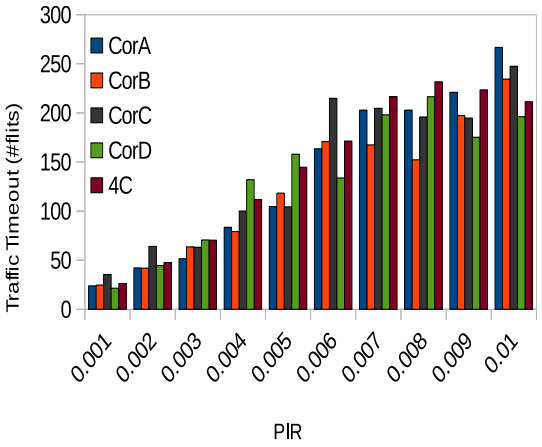
The proposed 4-corner scheduling driven on-line test mechanism named as the 4C-Model for addressing open faults in channels of on-chip interconnection networks is seen to be complimented by many prior test schemes. In this section, the following approaches are selected as the prior test schemes in order to properly compare their efficiency with the proposed 4C-Model. The prior schemes included in this comparison study are the 2×2 subnet selection model (2×2-Model) [37–39], the 2×1 subnet selection model (2×1-Model) [40, 41], the self-diagnosis model (SD-Model) [42, 185], the sequential router selection model (S-Model) [6,7], the hierarchical node selection model (H-Model) [RPC-18], the iterative 4x4 subnet selection model (4x4-Model) [RPC-12, RPC-14], the two hop node selection model (2hop-Model) [RPC-13], and the diagonal node activation model (Diag-Model) [Chapter 3]. Several parameters are evaluated and compared between the prior schemes and the proposed scheme. The comparisons among the schemes with respect to following metrics: silicon



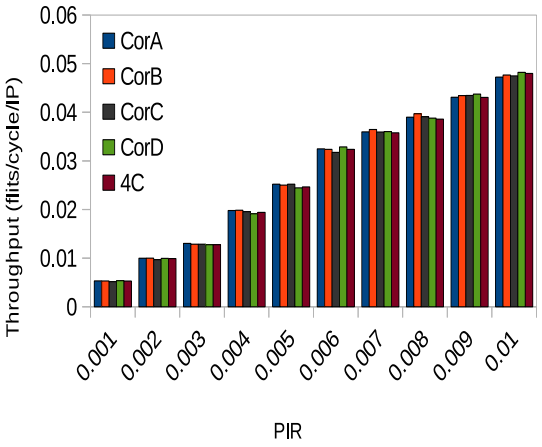
(a) Amount of packet flits injected.



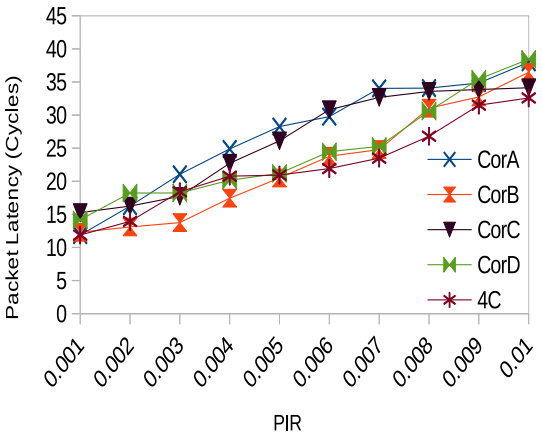
(b) Amount of packet flits received.



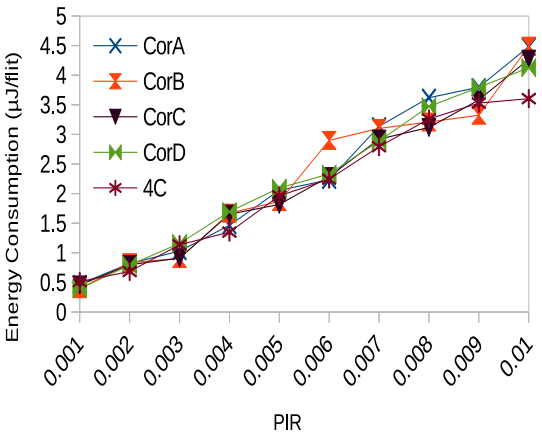
(c) Amount of packet flits lost.



(d) Throughput behavior.



(e) Latency behavior.



(f) Power consumed by a flit.

Figure 4.22: On-line evaluation of the proposed test solution applied at the corner nodes separately and concurrently on the the 16-bit octagon NoC.

hardware area overhead, number of test iterations, amount of test time, achievement on link and fault coverage metrics, size of dropped packets, amount of average packet latency, and amount of average energy consumption by a packet flit are performed on the 16-bit networks characterized in Table 4.7. The demonstration of the comparison study starts with the 4×4 mesh NoC and extended to other networks.

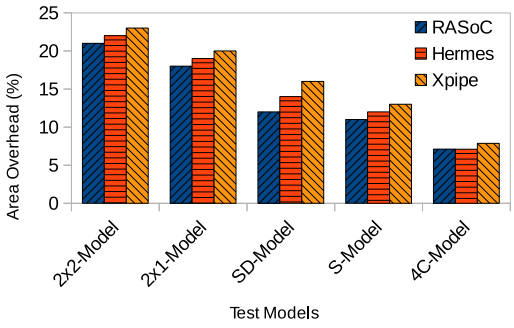


Figure 4.23: Comparisons on hardware area overhead among the test models.

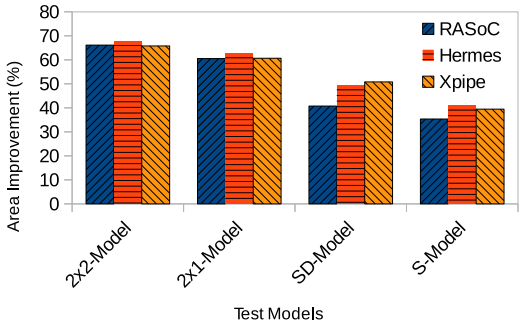


Figure 4.24: Improvement on hardware area overhead by the 4C-Model over prior works.

4.9.1 Comparison on Hardware Area Overhead

The main goal of the proposed 4C-Model is connectivity test via the testing of open faults in communication channels of NoCs. The intention of the test mechanism is to provide a high test resolution in terms of low test time, high fault coverage while keeping the hardware area overhead at a minimum. A BIST-based test mechanism is implemented with a pair of hardware blocks: TPG and TRA generally termed as test module (TM). The area of these blocks, i.e., a TM depends on one or multiple parameters like test size, routing distance between source and destination, and so on. It is noticed that in spite of the use of same test set, the routing distance plays a major role in the additional area overhead. Figure 4.23 describes the comparison of the area taken by a TM as proposed by a test scheme. For instance, in the 2×2-Model, a test packet from a core is routed to another core which is in the XY direction. In other words, the test data travels by four hops before getting analyzed. It shows that the area overhead incurred by a TM is nearly 21–23% on the RASoC, Hermes, and Xpipe routers. With respect to the proposed TM, this area is 65.78–67.73% higher. In other words, the proposed TM saves 65.78–67.73% router area. This improvement is illustrated in Figure 4.24. In the 2×1-Model, the routing distance between TPG and TRA for the analysis of the test data is reduced to three hops as compared to the 2×2-Model. Consequently, little improvement in the area overhead is observed. It is seen that the TM area, in this case, is nearly 18–20% on these routers. This area is 60.51–62.63% more than our TM-area. Further improvement in the area overhead can be observed in the SD-Model and subsequently in the

S-Model. In the SD-Model, test data are analyzed on traversal of two hops. Correspondingly, the TM-area is about 12–16% with respect to the above-mentioned routers. Therefore, the proposed TM takes 40.75–50.81% less area. In the S-Model, although the routing distance between TPG and TRA is the single hop, still the area overhead is shown to be higher than the proposed scheme. This due to the fact that the S-Model is aimed to address faults in interswitch channels only. When this scheme includes faults in the local channels, the TM-area is increased. It is seen that this area is nearly 11–14% and approximately 35.36–40.83% more than that of the proposed TM. The basic working principle of the test algorithms practiced in H-Model, 4x4-Model, 2hop-Model, and Diag-Model is alike. Thus, no benefits on the area overhead are gained. Thus, the proposed 4-Corner driven test scheme improves test area overhead by 35.36–67.73% for 16-bit networks.

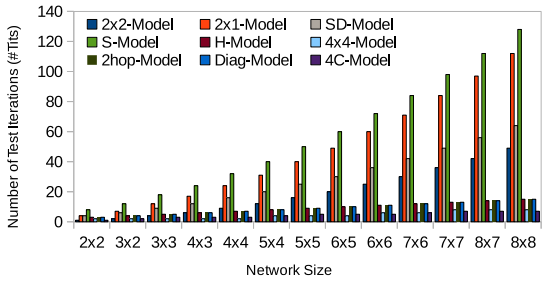


Figure 4.25: Comparison on the test iterations among the test models.

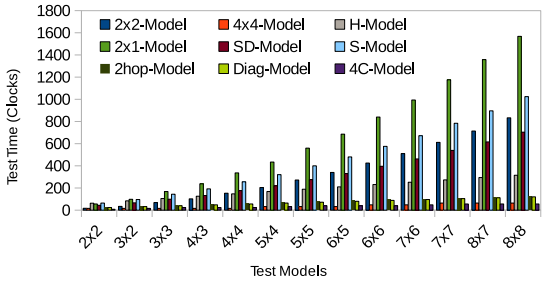


Figure 4.26: Comparison on the test time incurred by the test models.

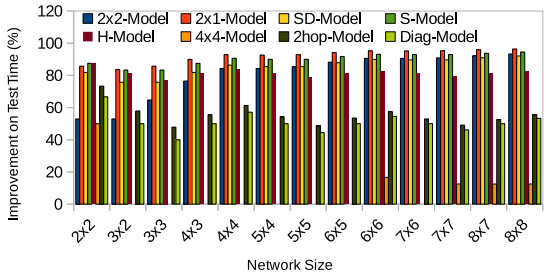


Figure 4.27: Improvement on the test time by the 4C-Model.

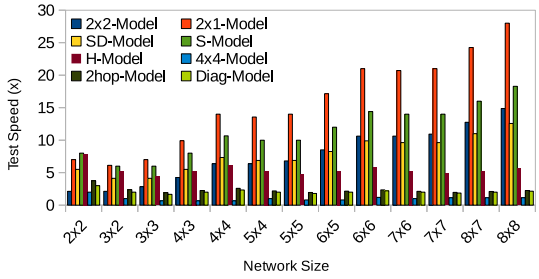


Figure 4.28: The speeding up of the 4C-Model over the prior works with respect to test time.

4.9.2 Comparison on Test Iterations and Time

Next benefit by the proposed 4C-Model over the prior models can be seen with respect to the number of test iterations (#Tits) and the associated amount of test time (#Clocks) incurred by a test method used in these schemes. One target of a test method is to detect a fault in the channels of an NoC as early as possible. However, the overall test time incurred by the test method depends at least on two basic entities. The first entity is the test application mode. The testing can be done either in the off-line mode where the whole network is assumed in

the test mode or in the on-line mode where a part of the network is put in the test mode and rest part is allowed to carry on-going regular applications. In the former case, the testing of a component, for instance, communication channels can be accomplished in just two iterations. However, no application is allowed. There are many applications that are accomplished in the second case. Therefore, selection of a subnet (that is put in the test mode) by a test scheme, determines the number of test iterations on a network (Figure 4.25). For example, the 2×2-Model in the on-line mode selects a 2×2 subnet at a time in the test mode. Therefore, an $M \times N$ network can be tested in (M-1)*(N-1) iterations, e.g., the 4×4 NoC can be tested by #Tits=9 iterations. Similarly, the test iterations can be determined by other prior works as per their subnet selection schemes.

The second entity is the test time incurred by a test algorithm on a test iteration. In each iteration, a test packet travels one or multiple hops before analysis. Therefore, latency of the packet (T_{tpt}) per hop is added to the T_{it} in addition to other factors. For example, in the 2×1-Model, the interconnection of an interswitch and its adjacent local channels can be constituted in the test mode. The test packet in this interconnection travels three hops that equal the $T_{tpt} = 9$ clocks resulting in $T_{it} = 14$ clocks. With this model, it is seen that the 4×4 network requires #Tits=24 iterations that result in the overall test time $T_{n/w} = 14 \times 24 = 336$ clocks (Figure 4.26). The proposed scheme takes only #Tits=3 and $T_{it} = 8$ clocks on this network that have resulted $T_{n/w} = 24$ clocks only. Thus, the proposed scheme saves the test time by 92.86% (Figure 4.27) and becomes $336/24 = 14\times$ faster (Figure 4.28) on the 16-bit 4×4 network. Continuing the test time evaluation by other prior works, the proposed 4C-Model saves the overall test time up to 96.43% and becomes $28\times$ faster on the networks included under comparison.

4.9.3 Comparison on Coverage Metrics

Next, the proposed scheme is compared with other works on the basis of the achievements on the coverage metrics, i.e., LCM and FCM. Considering the open fault model and iterating the test application in the subnets as selected by the schemes, each communication channel of a network is therefore tested for open faults. Naturally, every work whether prior or the proposed scheme provides the LCM up to 100%. The test source (TPG) and test destination (TRA) are independent of the networks and designated for the detection of open faults in channels, i.e., exercises single fault model (SFM). The test mechanism in every scheme should confirm the FCM up to 100%. But, some schemes fail to reach this limit. This is due to the multi-hop routing distance between test source and destination pair. It is observed that more is the routing distance, more is the inability shown by a test mechanism to detect the fault. For instance, every test packet routes four channels in the 2×2-Model. Correspondingly, the fault simulation shows that approximately 77–80% of the modeled faults can be detected, i.e.,

$FCM \leq 80\%$ (Figure 4.29). In the 2×1-Model, received test data undergo the analysis after traveling three hop routing distance. It is observed that the fault simulation achieves the FCM in the range $\approx$ 79–84%, i.e., $FCM \leq 84\%$. Further 2–3% improvement on the FCM is seen when the fault simulation is conducted on the basis of the test mechanism discussed in SD-Model. In this scheme, the routing distance between test source and destination pair is two hop. In rest of the prior works, the test packet follows single hop transmission. Considering the fact, the fault simulation for each of these schemes provides up to 100% fault detection.

4.9.4 Comparison on Performance Overhead

Next to the coverage metrics, the on-line testing of communication channels in NoCs examines the network performance under different channel conditions, such as channels are fault-free, channels are not busy due to traffic congestion, channels experience open faults, and so on. Beside the different channel conditions, the number of test iterations due to the selection of a subnet in an NoC noticeably influences the network performance. The performance of a network is analyzed in terms of various metrics. Three metrics namely lost packets, packet latency, and energy consumption are considered in the comparison study. If the channels are freed from any kind of faults and not busy in any application, then the packets of a regular application can get immediate access of the channels and routed towards the destination at the normal speed of the network. Few packets get dropped in the network due to the delay incurred by the router in processing the packets. Therefore, latency and energy consumption are slightly increased. If the channels are busy involving another application or in the test mode, the transmitted packet forcefully halts at some routers in the routing path. Both packet latency, as well as energy consumption, are affected. For instance, let the packets get halted due to the testing of a subset of channels. As many times the regular application packets enter the test zones (i.e., test iterations), the packets last a long time resulting in the high packet latency and energy consumption. In addition, many packets due to the timeout error are dropped in the network. Further, the size of the dropping of packets is increased while a channel experiences open faults.

Figure 4.30 highlights additional performance overhead in terms of the size of dropped packet flits, average packet latency, and average energy consumption by the flit of a packet. The on-line testing behaviors of the prior schemes are evaluated by extending the simulation setup as followed in 16-bit 4×4 network. It is observed in this network that the 2×2-Model results in 21.75%, 18.25%, 22.49% more in the dropping of packet flits, packet latency, and energy consumption, respectively than the proposed 4C-principle based test mechanism. Thus, the performance overhead incurred by the 2×2-Model than the proposed 4C-Model is in the range 17.86–31%, 15.44–32.97%, and 14.48–34.95% for packet dropping, latency, and energy consumption, respectively. Similarly, the evaluation can be continued for other prior

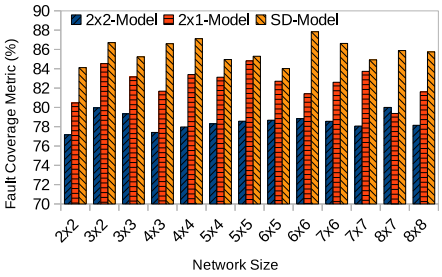
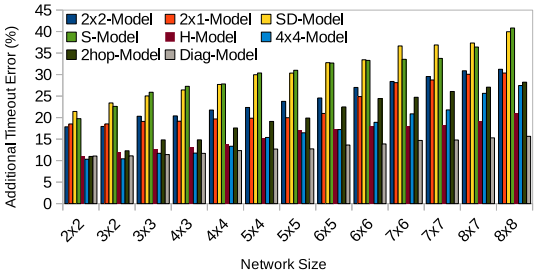
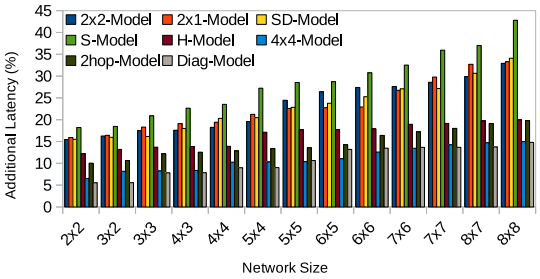


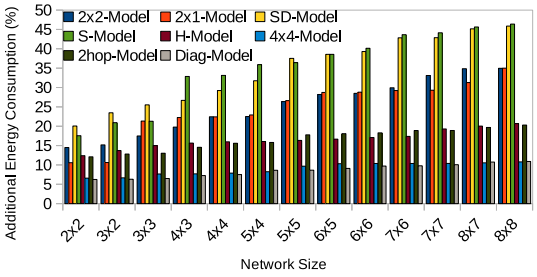
Figure 4.29: Comparisons on the FCM.



(a) Additional amount of packet loss by prior works.



(b) Additional packet latency exhibited by prior works.



(c) Additional energy consumption by prior works.

Figure 4.30: Additional performance overhead incurred by the prior schemes over the proposed 4C-Model.

test schemes. It is seen that the S-Model incurs high-performance degradation with respect to these metrics which are in the range of 19.75–40.83%, 18.21–42.79%, 17.55–46.38%. This is because of the fact that the scheme provides the highest number of test iterations, i.e., highest overall test time. Therefore, there is a possibility that application packets may enter the test regions the maximum number of times. For the similar reasons, the Diag-Model exhibits lowest performance overhead among the prior schemes. However, the approach shows 11.06–15.65% more packet dropping, 5.83–14.81% more packet latency, and 6.24–10.88% more energy consumption than the proposed test solution. Thus, the proposed 4-Corner driven test mechanism prevents the packet loss by 11.06–40.83% (Figure 4.30a), improves the packet latency by 5.83–42.79% (Figure 4.30b) and energy consumption by 6.24–46.38% (Figure 4.30c).

Therefore, fault detection must be conducted at the idle state of the network when no application is running in the network so that no performance overhead is incurred due to testing during the execution of regular applications. On the other hand, the on-line testing must be carried out in such way that the test time per iteration and the number of test iterations are as low as possible so that little performance overhead is incurred during the regular applications. Such performance degradation, however, can be minimized by employing an adaptive routing strategy (fault-tolerant routing algorithm) on the exploitation of the diagnosis results provided by the underlying test mechanism for the channels.

4.10 Limitations

The variation of the 1-step algorithm discussed in Section 4.4 can detect the connectivity between source and destination in the on-chip networks at the expense of few clocks per test iteration. The connectivity detection is carried in the form of testing of open faults in channels of the networks. The overall test time, however, is determined by the test time per iteration multiplied by the number of test iterations required to cover the faults in the channels of an NoC. The test algorithm is iteratively executed on the NoC where each iteration selects multiple nodes. The selection is done with a new scheduling scheme and is driven by the 4-corner principle. With this principle, the nodes are selected from the corner nodes that initiate the test algorithm concurrently and progressively advances the testing of open faults for rest of the channels in the NoC. As a result, this test scheduling method lowers the number of test iterations.

- **Test Iteration:** It is seen that the 4C-principle based test scheduling on a $P \times Q$ network completes the channel testing in just $\lceil (P+Q-2)/2 \rceil$ iterations (Equation 4.7). Though this number is comparatively low with respect to prior works, however, the parameter depends on the values of P, Q that determine the size of the network.
- **Test Time:** With the increasing values of P, Q, i.e., with the increase in the size of networks, the number of test iterations increases in these networks. As a result, the overall test cost in terms of the test time becomes higher.
- **Performance Overhead:** The proposed 4C-Model is applied in the on-line test mode in which an application packet may pass through multiple test regions. Consequently, degraded performance, such as high packet latency, high energy consumption, more packet timeout, etc. may be observed in these networks.
- **Test Energy:** The approach does not focus on the energy dissipated during testing of the channels.

Therefore, the proposed test solution needs to be improved through the reduction in the number of test iterations so that quick test of the channels in NoCs can be accomplished and network performance degradation in the on-line test mode can be improved.

4.11 Conclusion

In this work, a cost-effective on-line test mechanism has been discussed to account open faults in the communication channels of on-chip interconnection networks. Since, the open faults in channels which become a part of a routing path may disconnect the source and destination during communication, these faults must be handled before routing packets on

the path, otherwise, drastic packet loss may be observed. The test mechanism successfully detects all open faults in the local and interswitch channels. Subsequently, the diagnosis of the faults separates the faulty channel-wires from the channels, which can be later reused by a fault tolerant routing scheme in order to meet the reliable communication and prevent the enormous loss of packets in the networks. The test algorithm incurs low test cost in terms of test time. Further, the overall test cost is lowered by governing the test algorithm on the basis of the 4-corner principle. The principle selects multiple nodes that execute an instance of the test algorithm concurrently. For the reason, the proposed test scheme is named as the 4C-Model. Furthermore, this principle makes the proposed test scheme scalable with respect to the general networks. The generality of the proposed 4C-Model is illustrated with three network parameters- network size, channel width, and network type. The comparison study has shown that the proposed test scheme provides the better solution with respect to many evaluation parameters like area overhead, test time, fault coverage, and performance overhead. The proposed test scheme although provides a comparatively lower number of test iterations but like the prior works, this number depends on the size of the on-chip networks. Therefore, associated performance overhead can be observed on a large scale network due to higher test time. The issues related to lowering the number of test iterations and corresponding performance overhead are attempted in the next chapter which is dedicated to addressing the short faults in the channels of on-chip networks.

Chapter 5

Impact of Short-Channel Faults in On-Chip Interconnection Networks

5.1 Introduction

The evolution of integrated circuit technology has enabled system-on-chip (SoC) platforms to encompass a large number of embedded processing cores and a set of heterogeneous or homogeneous components on a single die. Multicore or multiprocessor SoCs (MPSoCs) integrate many IP-blocks seamlessly [2, 3] while maintaining manageable resource utilization and power consumption levels. For example, Tilera [4] launched TILE-Gx36, and TILE-Gx72 processors based various high-end MPSoC products to support large-scale computing applications such as complex and advanced networking, digital multimedia, cloud computing, and wireless infrastructures [6, 7]. The number of processors integrated on a single chip has gradually increased from few cores, e.g., Intel's quad-core processors [49] to thousands of cores, e.g., Adapteva's Epiphany [28, 50]. With the increasing number of such on-chip IP cores, traditional long bus based interconnections used in such designs provide insufficient bandwidth, and therefore, prevent large and complex applications from achieving their expected performance. Thus, these buses cause impediment to performance in terms of latency and power [186]. In order to cope with increasing bandwidth and speed, networks-on-chip (NoCs) have been successfully deployed that provide high-performance and scalable communication mechanism over various interconnection architectures. An NoC as on-chip interconnection network offers a viable, holistic solution to handle complexities such as integration of many IP cores and to alleviate communication bottleneck that arises in MPSoCs [5, 11, 12, 15, 16].

With ever-shrinking die size, an NoC suffers from several manufacturing defects. For example, NoC-channels with increased wire density, are more vulnerable to certain distinct logic-level manufacturing faults such as mutual shorts. The presence of such short-channel

faults badly impacts basic performance metrics, e.g., throughput, packet latency, energy consumption. In order to tackle the resulting loss of yield and reliability, either of the following two methodologies can be practiced. One way is to replace the faulty channel-wires using spare, defect-free wires. However, spare wires duly introduce high silicon area overhead and congestion. Another solution is to deploy some fault-tolerant routing algorithms [26, 44, 46, 102, 151, 187] so that the faulty channel-wires are not used in the communication route. These methods allow an NoC to accommodate short-channel faults and to utilize partially faulty channels in order to achieve reliable data transmission. In both the approaches, one has to identify the faulty channels and this mandates diagnosis of these faults (here, shorts) in the channel. Unfortunately, most of the fault-tolerant approaches do not include a test scheme for detecting channel faults in NoCs. Thus, a mechanism for detecting short-channel faults and identifying their locations in an NoC-based system is highly needed for implementing fault-tolerant routing and to enhance system reliability.

Rest of the chapter is organized as follows. Motivation behind the contributions are discussed in Section 5.2. Section 5.3 outlines how the clusters are formed in a network. Different system-level failure modes caused by the channel-short faults are described in Section 5.4. Section 5.5 discusses the details of short-channel fault model. Section 5.6 presents the test infrastructure. The test methodology and node-scheduling are presented in Section 5.7 and Section 5.8, respectively. Simulation results are reported in Section 5.9. The scalability and adaptability issues are discussed in Section 5.10 and Section 5.11, respectively. Section 5.12 presents comparative study of the proposed model with a set of prior works. Basic limitations of the proposed scheme are discussed in Section 5.13. Section 5.14 concludes the chapter.

5.2 Motivation, Problem Formulation, and Contributions

This chapter presents a distributed, low-cost and fast on-line test solution that addresses short-channel faults in a general NoC communication architecture and analyses their impact on its performance. The proposed test algorithm requires little hardware overhead and detects both intra- and inter-channel short faults in interswitch as well as in local channels followed by identification of faulty wires that are connected to routers and cores. The well known walking-one sequence (W1) [118] is used as test vectors to handle short-channel faults.

The primary goal of the proposed scheme is to design test rounds/iterations that are independent of the size of the network and channel-width. Variable-length test rounds/iterations usually incur high test time (clocks), which in turn, increases the latency of application packets, resulting in degradation of NoC-performance. Another goal is to achieve lower performance overhead and balanced resource utilization for every test round/iteration. This will lead to an effective test scheduling where multiple NoC nodes are allowed to execute

the test algorithm in parallel. An NoC architecture is segmented into two instances on the basis of the out-degree of an NoC node [170]. Each instance is further partitioned into at most four clusters, each with the similar node-degree profile. The clusters govern the test iterations and all the nodes in a cluster in an iteration execute the test algorithm concurrently for detecting the respective short-channel faults. Synthesis results show that the hardware area overhead is nearly 12—14%. Also, for the proposed cluster-based scheme, the same test clock can be used for all NoCs. Experimental results reveal that all short-channel faults can be tested thereby achieving 100% coverage metrics. Note that the number of short-channel faults grows rapidly with channel-width, and their impact on performance degradation may become severe. The impact of these faults is studied on NoC performance in terms of various metrics. Unlike other previous works, the proposed cluster-driven scheme for test scheduling needs a fixed number of test iterations, and thus renders it scalable with NoC size, channel width, and topology. Also, the technique lowers the hardware area overhead up to 27% and runs 21X faster compared to prior work. In addition, packet latency and energy consumption are also reduced significantly. The reduction of these metrics can be observed up to 19.47—-40.16% and 17.57—-34.20%, respectively. Thus, the proposed method improves hardware area overhead, test time, and performance overhead in an NoC.

The proposed model offers several advantages over previous methods. It covers both off-line and on-line test modes, and it completes testing of all channels in an NoC in just two rounds, each consisting of at most four iterations. Since the method can identify faulty channel-wires, it can also be used as a precursor to fault-tolerant routing mechanism where the data-traffic needs to be redirected by avoiding faulty channels.

The main contributions of the chapter are now summarized below:

- Detecting intra- and inter-channel short faults in an NoC. It ensures the state of faultiness/non-faultiness of a channel-wire in the NoC.
- Diagnosing the channel's shorts. It helps to identify faulty channel-wires in a channel. The faulty wire-set may be exercised by a fault-tolerant routing algorithm that can direct the traffic by avoiding this faulty wire-set.
- Efficient test scheduling; it ensures lower test time and offers parallelism. In other words, test time becomes independent of size, and type of the NoC.
- Evaluation of the proposed solution in terms of hardware area overhead, test time, link and fault coverage metrics.
- Studying the effect of the short faults in terms of various well-known performance metrics at sizable traffic.
- Establishing the scalability behavior with a larger network and higher channel width.
- Establishing the solution-adaptability feature with an octagon network.

5.3 Cluster Formation

In this work, a scalable test methodology is proposed that detects short-channel faults and locates faulty channel-wires in an NoC communication architecture. A major challenge is to ensure the same test time irrespective of NoC size and type, and similar performance overhead in a test round. The selection of nodes that are allowed to execute the test algorithm in parallel may reduce the test time and may implement similar test configuration in a test round. Such type of preprocessing task is referred to as test scheduling.

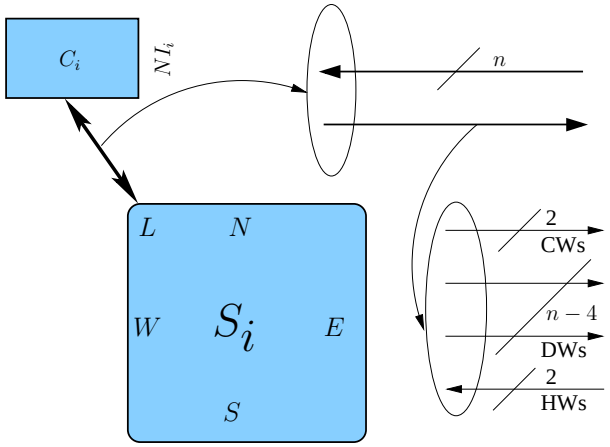


Figure 5.1: Representation of a node.

As stated, an NoC communication architecture consists of processing cores, routers, and channels as transmission medium. The physical organization of these basic blocks determines the topology of the NoC. An NoC topology can be considered as a graph $\mathbb{G} = (\aleph, \complement)$, where $\aleph$ represents the set of nodes $N_i \leftarrow < R_i, C_i >; i \in \mathbb{N}$, and $\complement$ represents the set of channels $Ch_j; j \in \mathbb{N}$. Figure 5.1 represents an N_i. Blocks R_i, C_i denote a router and its dedicated core, respectively. A channel is either interswitch type shared by a pair of routers, i.e., $(R_i, R_j; i \neq j)$ and vice versa or local type shared by a router and its dedicated core, i.e., (R_i, C_i) and vice versa. Each channel consists n wires that are classified into data wires (DWs), control wires (CWs), and handshake wires (HWs). Further, these channels can be classified into bidirectional or unidirectional depending on the type of communication modes-simplex, duplex. Henceforth, the graph $\mathbb{G}$ of an NoC with bi-directional channels is modeled as an undirected graph while the NoC with unidirectional channels (which are assumed in this work) is modeled as a directed graph or digraph [170]. Figure 5.2 shows an abstract view of a 4×4 mesh NoC with unidirectional channels, whereas its graphical representation $\mathbb{G}$ is shown in Figure 5.3.

One dominant factor that drives the test time is the test rounds that determine test iterations. Many researchers have devoted their efforts towards lowering the number of test rounds since the number depends on NoC size and type and affects the NoC performance

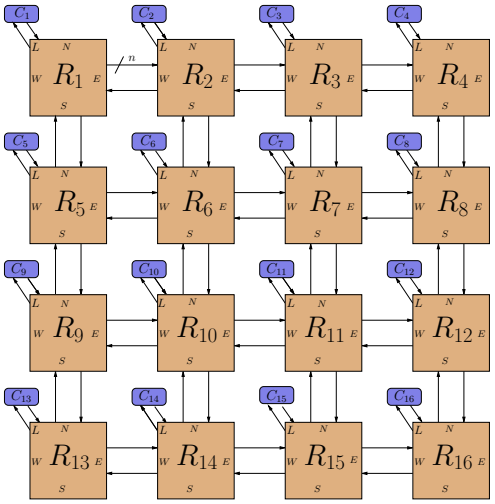


Figure 5.2: Abstract view of a 4×4 mesh.

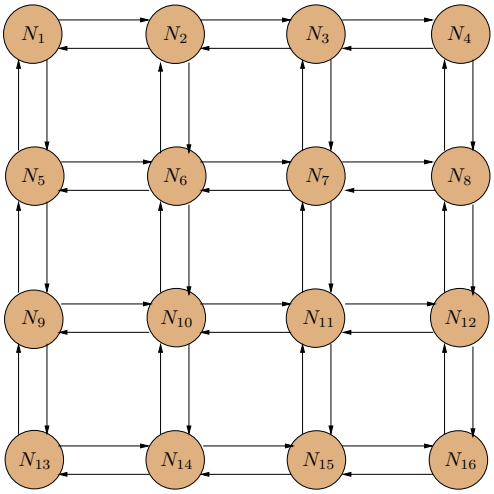


Figure 5.3: Graphical view $\mathbb{G}$ of Figure 5.2.

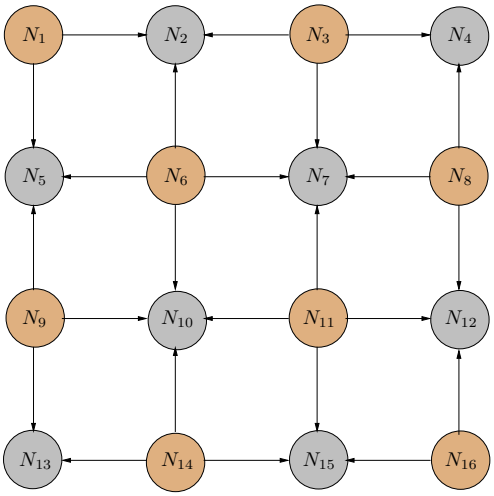


Figure 5.4: The $\mathbb{G}^+$ instance of Figure 5.3.

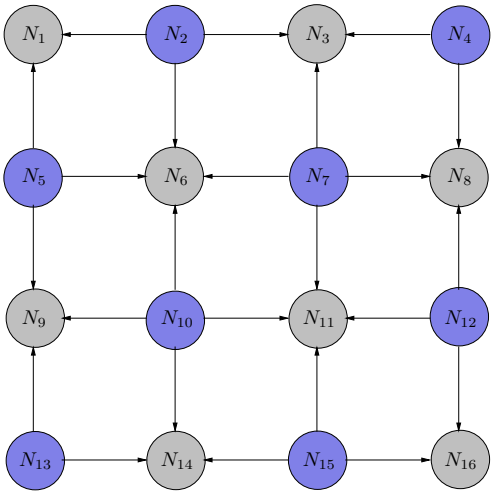


Figure 5.5: The $\mathbb{G}^-$ instance of Figure 5.3.

as a consequence. Clusters are often considered as functional, efficient, and scalable tools in building basic block for modern high-performance computing (HPC) infrastructures, e.g., NoC-based communication architecture. Simultaneously, the tool is often used to solve many complex problems. The advantages of clustering are exploited in the proposed scheme so that the test rounds and/or iterations become invariant with NoC size and type resulting in the need for the same number of test clocks (time) for all NoCs in general.

The $\mathbb{G}$ is split into two instances $\mathbb{G}^+$ and $\mathbb{G}^-$. These instances for Figure 5.3 are shown in Figure 5.4 and Figure 5.5, respectively. The $\mathbb{G}^+$ (or $\mathbb{G}^-$) is constructed with respect to out-degree $\delta^+(N_i)$ of a node N_i [170] at an odd (or even) diagonal level. The diagonal level of the N_i in an $P \times Q$ NoC can be found as discussed in Chapter 3. For example, the $N_1 \leftarrow < R_1, C_1 >$ is at the odd diagonal level while the nodes $\{N_2 \leftarrow < R_2, C_2 >, N_5 \leftarrow < R_5, C_5 >\}$ are at the even diagonal level in Figure 5.3.

Table 5.1: Cluster-set formation for Figure 5.3.

$\mathbb{G}$'s Variant	Cluster-set Type	Cluster-set Elements
$\mathbb{G}^+$	$\mathbb{C}_4^+$	$\{N_6, N_{11}\}$
	$\mathbb{C}_3^+$	$\{N_3, N_8, N_9, N_{14}\}$
	$\mathbb{C}_2^+$	$\{N_1, N_{16}\}$
$\mathbb{G}^-$	$\mathbb{C}_4^-$	$\{N_7, N_{10}\}$
	$\mathbb{C}_3^-$	$\{N_2, N_5, N_{12}, N_{15}\}$
	$\mathbb{C}_2^-$	$\{N_4, N_{13}\}$

The proposed distributed method decomposes an NoC architecture into two instances. Further decomposition of the instances is required into clusters so that the proposed test scheduling becomes effective in terms of the test parallelism, which is one of the most important performance factors for large-scale NoCs. As, the $\mathbb{G}$ is partitioned into $\mathbb{G}^+$ and $\mathbb{G}^-$, a test application on the instances is naturally limited by two test rounds. Each instance is decomposed into clusters. A cluster in $\mathbb{G}^+$ or $\mathbb{G}^-$ consists of set nodes that have the same out-degree δ^+. It is seen that generally, three cluster-sets are possible in $\mathbb{G}^+$ or $\mathbb{G}^-$. Table 5.1 provides the cluster-sets based on $\delta^+(N_i) = 4$, $\delta^+(N_i) = 3$, $\delta^+(N_i) = 2$ for the instances of Figure 5.3 as shown in Figures 5.4, and 5.5. Additional cluster set $\mathbb{C}_1^+$ or $\mathbb{C}_1^-$ for $\delta^+(N_i) = 1$ may be obtained if the NoC is irregular and/or indirect for example, hybrid, reduced mesh NoCs [15]. In the proposed test scheduling as described in Section 5.8 all the nodes in a cluster are allowed to execute the test algorithm concurrently. Each cluster set with a test application then determines a test iteration of a test round. As the number of cluster sets in a $\mathbb{G}^+$ and $\mathbb{G}^-$ are cumulatively few, the overall test cost becomes smaller.

5.4 Network Failures due to Channel-Shorts

In this work, the channel-shorts as the interconnection faults are tackled. The logic level interconnect's short faults have impressive effects on the functional behavior of an NoC system by putting it into several types of failure modes. These modes are categorized here as packet duplication, misrouting, delay, and dropping. These failure modes are realized in terms of channel-errors based on the type of faulty channel-wires. As the current work accounts for interconnect's short faults, it is needed to define these failure modes that may help to better tune the analysis of both coverage and performance metrics.

(1) Packet duplication- An NoC is a packet switched network where each node communicates via packets. Application data are packeted and placed as payload flits. Any payload packet flit in terms of signal if transmitted on the data channel-wire l_a is also available over a channel-wire l_b at the receiver node because of intra-shorts between the channel-

wires. Consequently, receiver node receives multiple copies of the flit over the shorted wires. The network in this case has experienced the packet duplication mode. The size of duplicate packet flits increases while more and more shorts affect the wires in a channel.

(2) Packet misrouting- Every packet is followed by its header flit i.e., beginning of packet (bop) signal. The packet header flit ("bop" signal) is transmitted on a control wire. If the inter-shorts affect a control-wire, the header flit gets modified. The router of the affected control-wire detects the packet misrouting failure and consequently updates its routing table to forward the packet with the modified header to another node instead of an intended node.

(3) Packet delay- Due to intra-shorts, more and more data packet flits are duplicated. As a result, the channels in the network experience heavy traffic that cause congestion. The packets are delayed to reach the destination from their expected time. The network is assumed to undergo the packet delay failure mode. The packet duplication due to intra-shorts is considered to be one responsible factor in this failure mode.

(4) Packet dropping- One reason for this failure mode is the packet delay experienced in the network. Few delayed packets are dropped in the network. The dropped packets enhance the timeout error. A packet timeout is reported in a network when a packet does not receive its trailer flit, end of packet (eop) signal within a predetermined time interval. Further, a control wire carries the trailer flit i.e., "eop" signal which can be modified when the wire experiences an inter-short. Consequently, the router over the affected wire drops the packet carrying the modified trailer. The router never forwards the packet to its destination. In this case, the "eop" signal can never be received by the destination resulting dropping of the packet. The packet dropping caused by the short faults duly enhances the timeout error.

Beside the data-wires and control-wires, the shorts may affect the handshake-wires in channels. The handshake-wires transport information such as "val" and "ack" signals. Whence such wires become faulty, the handshake information is also modified. In presence of modified handshake information, a receiver may have received the correct packet but a sender may resend the packet as it has received a modified "ack" signal. Here, the failure mode as the effect of shorts on handshake-wires is studied with the packet duplication mode. The above-mentioned failure modes are realized in terms of three types of errors at the node over a channel affected by shorts; these are namely, payload (for packet duplication), misrouting (for packet misrouting), and timeout (for packet delay, dropping, timeout) errors.

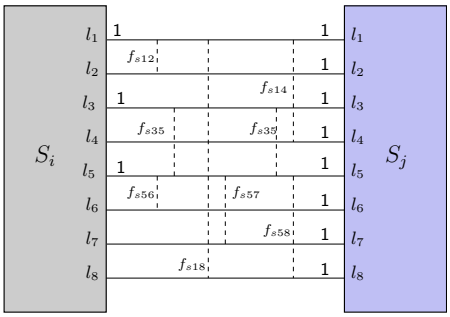


Figure 5.6: Intra-channel shorts in the channel (S_i, S_j).

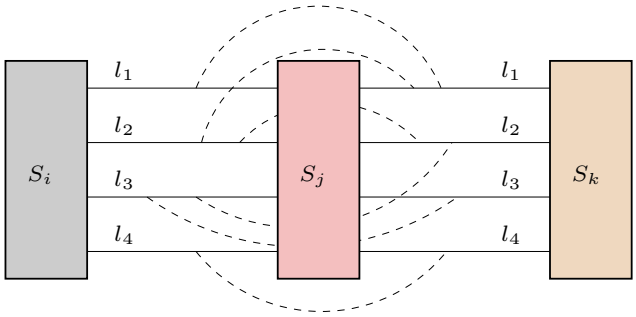


Figure 5.7: Inter-channel shorts between channels (S_i, S_j) and (S_j, S_k).

5.5 Short-Channel Fault Model

Here, a short fault model that has been used in the channels is described. A short-channel fault is a logic level hardware failure that occurs in g, a group of channel-wires. This g is called as fault group. These wires may belong to the same or different channels. Subsequently, shorts are categorized into intra- and inter-channel shorts, respectively. These are also referred to as intra- and inter-shorts, respectively. Figures 5.6 and 5.7 demonstrate intra- and inter-channel shorts, respectively. Each short fault for example is denoted by the f_{sab} between wires l_a, l_b; $a, b = 1, 2, .., 8$ and $a \neq b$ of the channel (S_i, S_j). Here, the S_i is router R_i. These shorts between the wires in the same fault group may occur with single or multiple instances. For example, the $g = \{l_3, l_5\}$ in Figure 5.6 possesses two instances of intra-shorts in the (S_i, S_j). However, single/multiple instances have the same effect with little latency degradation. Since chips are continuously shrinking, adjacent channel wires become closer with increasing channel width and wire density. As a result, the neighboring wires become more susceptible to shorts. However, the occurrence of shorts tends to decrease when g increases. It is seen that the occurrence of shorts beyond $g > 4$ is negligible.

$$\#S1 = \sum_{g=2}^{5} \frac{n!}{g!(n-g)!} \times p \qquad (5.1)$$

$$m = 2(d+1)n \qquad (5.2)$$

$$\#S2 = \frac{m!}{2!(m-2)!} \times q \qquad (5.3)$$

$$\#S = \#S1 \times |\complement| + \sum_{i=1}^{|\aleph|} \#S2 \qquad (5.4)$$

Considering layouts of realistic NoCs that deal with high-density design, it is not reasonable to consider pairwise intra-channel shorts as discussed in [6, 7, 37–39] to analyze their effect on network performances. On the other hand, it is too pessimistic to consider that all possible intra-channel shorts can be experienced in a channel. It is thus sufficient to assume intra-channel shorts for $2 \leq g \leq 5$ to observe network performance more closely. Further, the channels (for example local channels of nodes) that do not share any common node are separated from each other sufficiently. Considerations of pairwise inter-channel shorts between

these channels are acceptable. Equations 5.1, and 5.3 define the $\#S1, \#S2$ as the size of intra- and inter-channel shorts at a node N_i, respectively. The n denotes a channel width. The m defines the number of channel-wires of the N_i with degree d. Equation 5.4 defines the $\#S$, size of shorts in an NoC of $|\complement|$ channels and $|\aleph|$ nodes. The symbols p, q denote instances of intra- and inter-channel shorts in a g. In this work, single instance, i.e., $p = q = 1$ for both type of shorts is considered.

The analysis of the proposed test scheme begins with a 4×4 mesh NoC that consists of 16 nodes and 80 unidirectional channels. The proposed test algorithm has to exercise 549440 intra- and 134144 inter-channel shorts, i.e., 683584 short faults throughout the NoC channels, each is with $n = 16$-bit. Short-channel faults in an NoC may cause various system-level failure modes such as packet duplication, misrouting, and dropping. These failure modes are observed in terms of payload, misrouting, and timeout errors. Since the proposed scheme follows at speed functional (on-line) testing, it needs to consider these failure modes in order to propagate the impact of shorts to a higher level of abstraction and generate suitable test sequences for improving fault-coverage. Note that the quality of the proposed test scheme is measured with low hardware area overhead, low test time, high coverage metrics, and low-performance overhead.

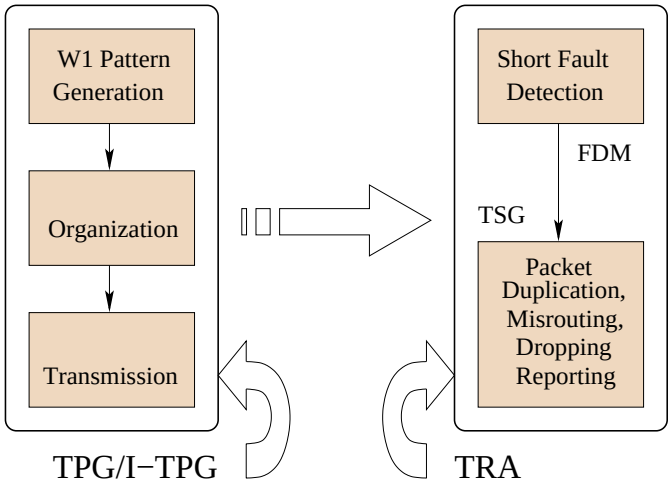


Figure 5.8: Abstract test architecture blocks at an NoC node.

5.6 Test Architecture and Packet Format for Fault Detection

The proposed test mechanism handles intra- and inter-channel shorts in NoC channels by executing a test algorithm at a node. The test architecture for channels consists of a pair of test pattern generator (TPG) and response analyzer (TRA) units. This pair is known as test module (TM). For every node, a TM is placed inside the router and its linked core wrapped up by the network interface (NI) unit, to reduce test time and performance overhead. Figure 5.8

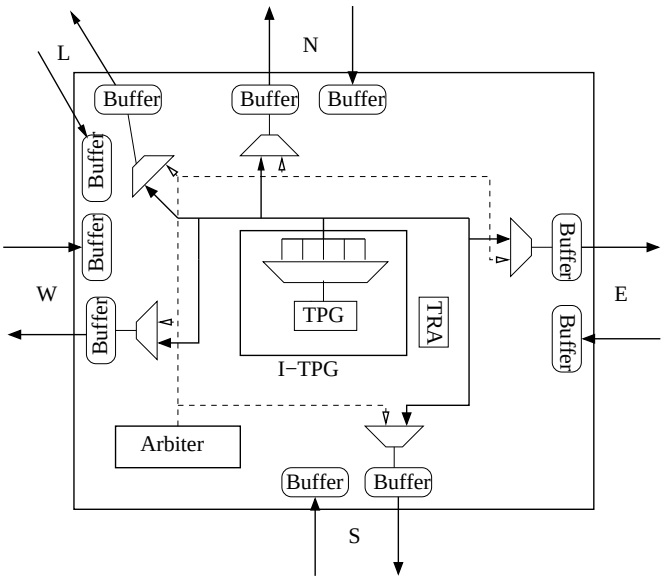


Figure 5.9: Typical interconnections of I-TPG unit with the O/P ports of a router.

shows an abstract representation of a TM. Additionally, each TPG is integrated with another special hardware component that would act like multicast wrapper unit (MWU) [3]. This TPG is now called as integrated TPG (I-TPG). The extra component consists of a demultiplexer and 2×1 multiplexers connected to O/P ports of routers. The I/P line of the demultiplexer is connected to the router TPG and each O/P line is connected to router's O/P port via a multiplexer. Other I/P line of the multiplexer is connected to the crossbar. The basic configuration of an I-TPG is shown in Figure 5.9. Such interconnections would help in monitoring test/application packets arriving at a node. The proposed test solution admits testing in the on-line mode where tests can be exercised to a subset of channels while keeping the rest of the network functional. Therefore, any incoming application packet has to wait at the router till the testing of all channels under the concerned node is completed.

Every TPG unit generates a test sequence in terms of test flits including header and trailer flits. The test sequence is so chosen that all modeled channel-short faults are detected. Without loss of generality, the shorts are covered using the well-known walking-one (W1) sequence [118] as test flits. These flits are packed using the credit-based wormhole switching technique that segments them into several test packets. Note that a packet size may vary with channel width, routing algorithm and other parameters. A typical packet format for a 16-bit channel width is shown in Table 5.2. In addition to assigned functionalities, every TPG has the capability of sending test packets. The TPG at a core unicasts the packets to its linked router while the I-TPG at a router multicasts the packets to the linked core and neighbor routers. Therefore, test packets having identical routing information, are sent from one router to different destination neighbors.

Table 5.2: A test packet organization using W1 sequences for n=8-bit data channel.

	Payload Flits (W1)								
H	0	0	0	0	0	0	0	1	T
e	0	0	0	0	0	0	1	0	r
a	0	0	0	0	0	1	0	0	a
d	0	0	0	0	1	0	0	0	i
e	0	0	0	1	0	0	0	0	l
r	0	0	1	0	0	0	0	0	e
	0	1	0	0	0	0	0	0	r
	1	0	0	0	0	0	0	0	

Table 5.3: The 2-bit TRA signals for the channel errors due to short faults in the channels.

BS	Signal Type	Meaning
00	No Fault	Data received over channel is correct
01	Payload Error (PE)	Data received at multiple ports
10	Misrouted Error (ME)	Data received at unintended node
11	Timeout Error (TE)	Data get dropped

A TRA unit, in addition to its capability of receiving test packets, analyzes received test flits (responses) to detect and identify possible shorts on faulty channel-wires. The TRA unit consists of two components: fault diagnosis module (FDM) and TRA signal generator (TSG). The FDM performs pattern checking using logical operations between the received and local test flits. Shorts can be experienced on control, data, and handshake channel-wires, and each packet consists of the header, payload, and trailer flits. Therefore, shorts on these wires are analyzed in terms of faults (errors) in received packets. Shorts on data and handshake wires result in payload error while the shorts on control wires result in misrouted and timeout errors. The outcome is fed as input to the TSG module that generates a two-bit signal (BS) announcing the type of error detected by the FDM. Table 5.3 describes the map of two-bit TRA signals and possible type of errors.

The work assumes $n = 16$-bit channels (for starting the analysis of the proposed scheme), each of which can be treated as the 16-bit single bus. Therefore, a TPG needs to generate 16 test flits, i.e., $|W1| = 16$ which are packeted by wormhole switching at the routers and cores. Every packet has four flits including header and trailer. Such small test sequences yield reduced area overhead for TMs.

5.7 Testing of Channel-Shorts at a Node

The proposed test scheme targets detection of intra- and inter-shorts and subsequent identification of faulty wires in the channels of a node. The scheme works engaging the functional mode of the NoC leaving aside only a subset of channels, which are under test. The test is performed in three phases. In the first phase, tests are exercised to detect shorts on data wires (DWs). In the second and the third phase, the presence of possible shorts is checked in channels and handshake wires, respectively. While testing channels, test packets are shifted from their parent node to its neighbors, where the responses are observed.

5.7.1 Fault Detection

A subset of the W1 test pattern set is placed into packet's payload by adding header and trailer flits to address shorts on data-wires. The later flits are mandatory to apply as they carry necessary routing information. The payload follows the header and is followed by the trailer. The content of the header flit varies with nodes. During transmission of a test packet, each flit is shifted clockwise. Shifting of a flit is done in such a way that every channel is filled up at the same time. In that case, data-wire under test is set to logic "1" while rest data-wires are set to logic "0". Note that every data-wire in a channel is tested with a test flit in the payload. Shorts occurring on two or more data-wires affect a test flit in payload resulting in error, which can be detected by the TRA of a receiver. A test flit sent on a data-wire is then duplicated and made available on all shorted data-wires. Therefore, the payload error is manifested as packet or data duplication. For example, consider Figure 5.6. When bit "1" is transmitted on l_1 from the S_i, the bit is received on l_2, l_4, l_8 including l_1 on S_j. Thus, l_1 is shorted with l_2, l_4, l_8.

Detecting shorts on control- and handshake-wires are handled subsequently. Control wires carry header and trailer flits. In order to tackle shorts on these wires, additional W1-set is accommodated with control framing bits. Since packets are transmitted with header and trailer flits, these flits are affected due to shorts. When a control wire that carries the header flit, i.e., control-framing bits with the beginning of the packet (bop) signal, the header flit gets modified. In such a case, packet routing is changed at the receiver over the channel. Thus, a packet misrouting is detected by the TRA of the receiver. As a result, the receiver forwards the packet with the modified header, which is available to another receiver. Like the header, every packet is accompanied by a trailer. If a control wire that carries this flit and participates in some short faults, then the end of packet (eop) signal in control-framing bits is also modified. The receiver over the channel rejects the packet thus reporting a timeout error by the TRA of the packet receiver. As a result, the intended receiver may not receive the trailer on time. Like data and control wires of a channel, handshake wires may suffer from short faults. Shorts on handshake wires are detected when handshake protocol bits, i.e., "val"

$$\oplus \begin{cases} 10000000 \\ 01000000 \\ 00010000 \\ 00000001 \\ 10000000 \\ - - - - \\ 01010001 \end{cases} \quad (5.5) \qquad + \begin{cases} 10000000 \\ 01000000 \\ 00010000 \\ 00000001 \\ 10000000 \\ - - - - \\ 11010001 \end{cases} \quad (5.6)$$

and "ack" signals are modified. Therefore, additional W1-set must be maintained in the test packets to address shorts on these wires. In either of the test phases, if a response on a wire is observed to be the same as the test flit transmitted, then the wire is considered to be normal, faulty otherwise. Hence, this procedure establishes the state of faultiness or non-faultiness of the concerned wire.

Since the underlying network is assumed to be in the normal mode of operation except the channels of a node-under-test, there may be a possibility of packet deadlock as a logic fault, in addition to payload, misrouting, and time-out errors. However, the proposed test algorithm ensures that it does not cause any packet deadlock. This happens because the routing path of a test packet in the test configuration is confined to a single hop (i.e., only one channel length), and the test set that is exercised as small packets, is finite.

5.7.2 Fault Diagnosis

Note that the detection of short faults in channels of a node is not enough. These faults must be diagnosed so that the root cause (the set of shorted (faulty) channel-wires) is identified. Once the root cause is known, a fault-tolerant routing algorithm can be invoked to compensate the loss of performance. The diagnosis of the faults is done by the TRA at receiver core and routers. The basic job of each TRA is responsible for verifying the incoming test flits (responses) against the test flits (W1) already generated by the TPG or I-TPG in the receiver. The FDM in a receiver periodically selects the responses and corresponding local test flits to analyze an error in the received packet, i.e., which data, control, or handshake wire is affected by a short fault. Then the component performs the "OR" and "XOR" operations to find the set of shorted wires with and without the wire-under-test.

For example, consider Figure 5.6. Assume that the wire l_1 is under test. Then the I-TPG of S_i sends the bit stream "1000 0000" as a test flit on the wire. Since the wire is shorted with the wires l_2, l_4, l_8 of the channel, the TRA of S_j receives incoming test flit as "1000 0000", "0100 0000", "0001 0000", and "0000 0001". Hence, three additional flits are received. These flits

undergo logic operations with the local "1000 0000", which is generated by the I-TPG at S_j. The logic operations are shown in Equation 5.5 and 5.6. From the operations it is observed that l_2, l_4, l_8 are shorted with the l_1. Thus, the TRA is able to check the payload, misrouting, or timeout error depending on whether l_1 is a data or control wire. The results obtained after analysis are sent to the TSG-unit of TRA, which generates a 2-bit signal in accordance with Table 5.3, followed by announcing a channel error as indicated by the diagnosis procedure. In case of diagnosing packet misrouting, the TRA receives a modified header flit sent by S_i. In this case, S_j forwards the packet to an unintended receiver during data transmission in the network. While diagnosing packet timeout error, the TRA waits for the trailer flit of the packet for a predefined time interval. Otherwise, the packet is assumed to be dropped/lost at S_j and the router does not forward it. This is caused by the modified trailer flits affected by shorts on the control wire carrying the packet trailer. Similarly, the faults on handshake wires can be diagnosed on the basis of incoming test flits over the wires, i.e., from the received "val" and "ack" signals.

5.8 Test Scheduling

The major challenges that need to addressed while designing a test algorithm (Section 5.7) include (1) the selection of the test set needed for detecting shorts on the NoC channels, (2) transmission of test packets, i.e., defining the packet flow so that the goals of the proposed cluster-driven scheme are met. The former issue can be overcome by selecting the W1-set. The raw W1-set is fragmented into smaller packets and the corresponding packets are transmitted during a test phase. It is seen in Section 5.9 that all modeled shorts are detected with such packet format. The latter issue can be handled with simultaneous transmission of test packets from the core and router of a node. Therefore, all outgoing channels of the node get tested in one attempt while the incoming channels get tested in the next attempt. The overall test time can be reduced with test parallelism whence multiple nodes are allowed to execute the test algorithm in parallel. Here a cost-effective scheduling of nodes is provided to improve test-parallelism so that constant-time testing can be achieved for general networks that consist of the same type of cluster-sets.

The proposed scheduling of nodes is based on the clusters as shown in the graphical view of the network (Section 5.3). Note that a network has two set of clusters $\mathbb{C}^+$ and $\mathbb{C}^-$ where test algorithm will be run. Therefore, testing of channels can be completed in just two test rounds R_1 and R_2 in either off- or on-line test mode. The first round R_1 is exercised on the $\mathbb{C}^+$ while the second round R_2 is exercised on the $\mathbb{C}^-$. Each cluster $\mathbb{C}^+$ (or $\mathbb{C}^-$) again consists of three sets $\mathbb{C}_4^+$, $\mathbb{C}_3^+$, and $\mathbb{C}_2^+$ (or $\mathbb{C}_4^-$, $\mathbb{C}_3^-$, and $\mathbb{C}_2^-$) of nodes depending on their outdegree δ^+. All nodes in a cluster-set are selected and allowed to execute the test algorithm concurrently. The proposed test application first selects the cluster-set $\mathbb{C}_4^+$ of nodes, each having $\delta^+(N_i) = 4$.

Table 5.4: Acronyms for test time required for channel shorts on an NoC.

Acronyms	**Definition**
Tits	The number of test iterations.
T_{it}	The amount of test time incurred by the test algorithm at a node.
T_{tpg}	The amount of test time needed by TPG to generate W1-set.
T_{tpo}	The amount of test time needed by TPG to organize W1-set into a packet.
T_{tpt}	The amount of test time needed by TPG to transmit a packet.
T_{tra}	The amount of test time needed by TRA to analyze a test response.
T_{NoC}	The amount of test time needed to test channel short faults on a network.

Once the test of channels from these nodes is over, current test application is shifted to the cluster $\mathbb{C}_3^+$ of nodes, each is with $\delta^+(N_i) = 3$. Subsequently, the test application is repeated in the cluster-set of nodes with $\delta^+(N_i) = 2$. The test application in next round is iterated on the nodes of the cluster-sets $\mathbb{C}_4^-$, $\mathbb{C}_3^-$, and $\mathbb{C}_2^-$. These test applications are called as test iterations ($Tits$). Thus, all channels in an NoC can be tested in just six iterations. One or two additional test iterations may be required when an NoC is an irregular and/or indirect type. Equation 5.7 defines the Tits for a network.

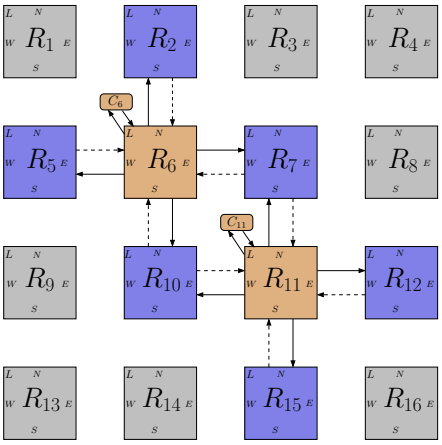
$$Tits = |\mathbb{C}^+| + |\mathbb{C}^-| \tag{5.7}$$

$$T_{it} = T_{tpg} + T_{tpo} + T_{tpt} + T_{tra} \tag{5.8}$$

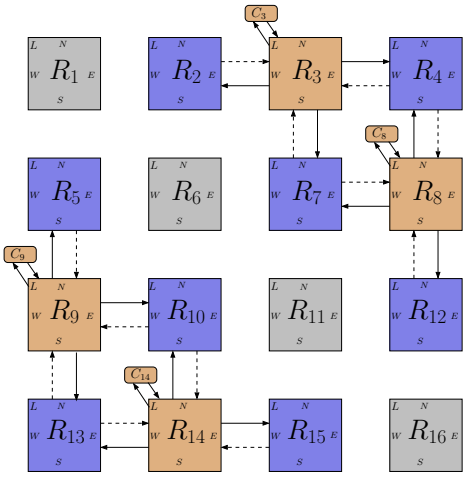
$$T_{NoC} = T_{it} \times Tits \tag{5.9}$$

One common disadvantage of prior methods is the requirement of high test time, which varies with network-size even when the same cluster-sets are obtained. The proposed mechanism is able to test multiple channels of a node in parallel and offers constant-time test completion regardless of the size of the network. The test time T_{it} at a node is defined in Equation 5.8 where T_{tpg}, T_{tpo}, T_{tpt}, and T_{tra} (Table 5.4) define the time needed for deriving the test set including header and trailer, organizing the test set, transmitting the test packets, and analyzing the test responses, respectively. Since all nodes in a cluster-set concurrently execute and apply the test mechanism, the T_{it} is same for a node as well as an iteration. Therefore, one can estimate the total test clocks T_{NoC} (defined in Equation 5.9) needed for testing all channels of an NoC. These parameters are evaluated in Section 5.9.

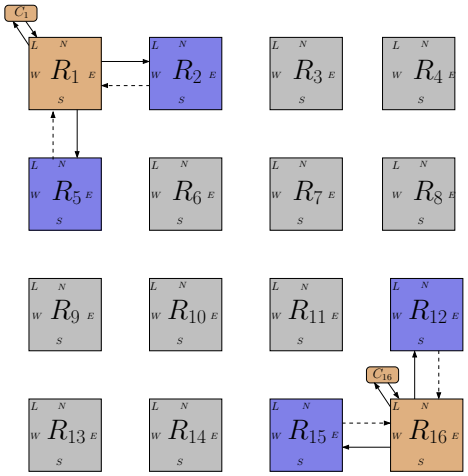
Figure 5.10 illustrates an application of the proposed test solution on a 4×4 mesh network with unidirectional channels. From earlier discussion, two clusters- $\mathbb{C}^+, \mathbb{C}^-$ are found in the network. The corresponding cluster-sets and the nodes in a cluster-set are provided in Table 5.1. In R_1, first test iteration I_1 (shown in Figure 5.10a) is accomplished by the nodes in the $\mathbb{C}_4^+$. The cluster-set consists of two nodes N_6, N_{11} that transmit test packets on their



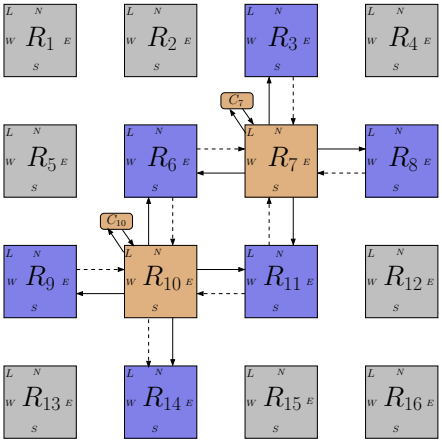
(a) Execution of the test algorithm at nodes $\in \mathbb{C}_4^+$ in the first test iteration I_1.



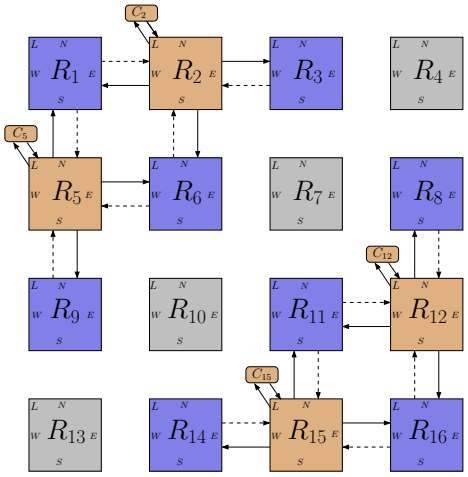
(b) Execution of the test algorithm at nodes $\in \mathbb{C}_3^+$ in the second test iteration I_2.



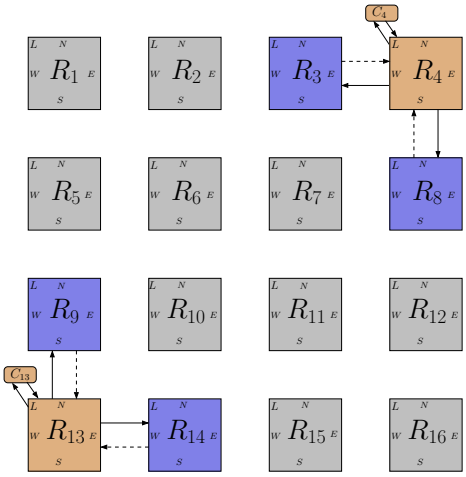
(c) Execution of the test algorithm at nodes $\in \mathbb{C}_2^+$ in the third test iteration I_3.



(d) Execution of the test algorithm at nodes $\in \mathbb{C}_4^-$ in the first test iteration I_1.



(e) Execution of the test algorithm at nodes $in\mathbb{C}_3^-$ in the second test iteration I_2.



(f) Execution of the test algorithm at nodes $\in \mathbb{C}_2^-$ in the third test iteration I_3.

Figure 5.10: Execution of the test algorithm at various cluster nodes in different test iterations (a-c) in the first test round R_1, and (d-f) in the second test round R_2.

outgoing channels (denoted by solid arrows) to neighbors whose TRA analyzes responses to address short faults. The inter-channel shorts on the incoming channels of the nodes denoted by dashed arrows are also detected by these TRAs at the neighbors. Once testing of these channels is over, the next test iteration I_2 in R_1 (shown in Figure 5.10b) is executed at nodes of the cluster-set $\mathbb{C}_3^+$, which consists of four boundary nodes. The last test iteration I_3 in the current test round (shown in Figure 5.10c) is carried out at nodes of the cluster-set $\mathbb{C}_3^+$ that consists of two corner nodes. However, the execution of the test algorithm at cluster nodes in the R_1 does not cover all shorts. Hence, the next test round R_2 is executed in a similar fashion at nodes of the cluster-sets $\mathbb{C}_4^-, \mathbb{C}_3^-, \mathbb{C}_2^-$ as I_1, I_2, I_3 respectively. Figures 5.10d, 5.10e, and 5.10f demonstrate the flow of test round R_2.

5.9 Results

The effectiveness of the proposed test model is demonstrated with its implementation on a set of $P \times P$ mesh NoCs described in Table 5.5. The characteristic parameters #R, #C, $|\mathfrak{C}|$, #W, #R-R, and #R-C represent the number of routers, cores, channels, channel-wires, interswitch channels, and local channels, respectively.

5.9.1 Simulation Setup

In order to evaluate the test clocks (TCs) and coverage metrics for the above networks, a simulation environment is built using the Xilinx 10.1 ISE Design Suite. The setup consists of Spartan3E FPGA family with the XC3S250E device and CP132 library package. The XST [VHDL/Verilog] as synthesis tool is used to analyze test area overhead of the proposed TMs at IP core and its linked router. The channels of a network in an iteration are placed to simulate shorts by transmitting test packets using TPGs and verifying the responses using TRAs. To see the effect of channel-shorts on network performance, the proposed on-line test scheme is implemented using the cycle-accurate simulator Noxim [22]. Table 5.6 provides basic parameters used during simulation. The $n = 16$ in this section and $n = 32$ in Section 5.10 are considered to show method scalability. The W1-sequences are used as application data that is packeted using wormhole switching technique. These packets are routed using the XY-routing algorithm. During simulation, packets passing through the nodes that are currently involved in testing, are held at the routers. The arbiters of the routers do not allow forwarding of these packets till the routers leave the test mode. The packet injection rate (PIR) is set in the range 0.01-0.1 that use random spatial traffic distribution to inject (synthetic) traffic as application data in a network. For each PIR value, simulation is continued for 10000 cycles that include 1000 cycles for warming-up.

Table 5.5: Characteristics of $P \times P$ NoCs.

N/w Size	#R	#C	$\lvert\mathbb{C}\rvert$	#W	#R-R	#R-C
2×2	4	4	16	256	8	8
3×3	9	9	42	672	24	18
4×4	16	16	80	1280	48	32
5×5	25	25	130	2080	80	50
6×6	36	36	192	3072	120	72
7×7	49	49	266	4256	168	98
8×8	64	64	352	5632	224	128

Table 5.6: Simulation Parameters

Parameters	Value
Channel-width	16, 32
Switching	Wormhole
Routing	XY
Traffic	Random
Payload	$W1$
Flit/Clock	1
Cycles Executed	10000

Table 5.7: Synthesis Results for the TMs in 16-bit channels.

n	TM Unit	#LGs	RASoC	Xpipe	Core
16	TPG	118	7%	8%	2.5%
	I-TPG	152	9%	10%	–
	TRA-Core	74	4%	6%	1.5%
	TRA-Router	114	6%	8%	–
	TM_1, TM_2	192, 266	11%, 15%	14%, 18%	4%

Table 5.8: Cluster-set nodes, test iterations and clocks analysis on $P \times P$ NoCs.

N/w	Test application on G^+ at R_1				Test application on G^- at R_2				$Tits$	T_{NoC}
Size	$\lvert\mathbb{C}_4^+\rvert$	$\lvert\mathbb{C}_3^+\rvert$	$\lvert\mathbb{C}_2^+\rvert$	$\lvert\mathbb{C}^+\rvert$	$\lvert\mathbb{C}_4^-\rvert$	$\lvert\mathbb{C}_3^-\rvert$	$\lvert\mathbb{C}_2^-\rvert$	$\lvert\mathbb{C}^-\rvert$		(clocks)
2x2	–	–	2	1	–	–	2	1	2	80
3x3	1	–	4	2	–	4	–	1	3	120
4x4	2	4	2	3	2	4	2	3	6	240
5x5	5	4	4	3	4	8	–	3	6	240
6x6	8	8	2	3	8	8	2	3	6	240
7x7	13	8	4	3	12	12	–	3	6	240
8x8	18	12	2	3	18	12	2	3	6	240

Table 5.9: Fault coverage analysis on $P \times P$ NoCs at $n = 16$.

N/w Size	Payload Error(%)	Timeout Error(%)	Misrouting Error(%)	Total FCM(%)
2x2	73.783	15.648	10.569	100
3x3	70.04	16.134	13.826	100
4x4	74.687	15.284	10.029	100
5x5	70.852	15.735	13.413	100
6x6	75.596	14.923	9.481	100
7x7	67.996	17.895	14.109	100
8x8	71.212	16.563	12.225	100

5.9.2 Area Overhead

A TPG (or an I-TPG) and TRA pair as TM, constitute the main instrument for testing intra- and inter-channel shorts. As mentioned, the pair is considered at both router and IP core of a node so that test clocks and performance overhead are reduced. The pair is implemented using Verilog in Xilinx ISE Design Suite to estimate hardware area overhead in terms of logic gates (LGs). Table 5.7 shows the synthesis results. Two well known NoC routers RASoC [65], and Xpipe [66] are selected to see the space occupied by the TM. An IP core is much larger in terms of area overhead than a router; for example, Plasma 16- (32-) bit RISC microprocessor is almost 4 (5) times larger than an NoC router [188]. So, the proposed TM in a core occupies a very little area. The TPG at a core unicasts test packets, while the element at the router multicasts similar test packets. Therefore, the TPG for a core is simpler than that of a router. The TRA at a core analyzes the responses received from its dedicated router while the unit at a router analyzes the responses received from its neighbor routers and core. Therefore, the later TRA is functionally more powerful than the former TRA. Clearly, size of a TM (say TM_1) at an IP core is smaller than that of say, TM_2 at its router. Further, the size of the TM_2 can be optimized up to 2–5% depending on the active I/O ports (channels) of a router and bit-width of a channel. For example, the corner router (R_1) (Figure 5.2) has three active (two interswitch and one local) channels, the TM_2 in the router is smaller than that in the border router R_2, and inner router R_6.

5.9.3 Test Clocks and Coverage Metrics Evaluation

Our experiments for detecting intra- and inter-shorts in NoC channels are performed using Modelsim simulator where a set of channels in an iteration are placed for testing. The Verilog is used to configure the test environment with fault injection campaign. Figure 5.11 provides the size of intra- and inter-shorts injected in the channels (at $n = 16$) of the network during fault simulation. The corresponding TPGs at one end of the channels are placed to derive test sequence, organize, and transmit the sequence in terms of packets. The TRAs at another end of the channels are placed to analyze the responses. Table 5.8 briefs the number of nodes selected in an iteration that initiate the test algorithm. From the table, one can easily find Tits as well as T_{NoC}. During the experiment, it is observed that a TPG (or I-TPG) takes 5 clocks to generate the W1-set including header and trailer, 1 clock to organize these raw data into 8 packets, and 32 clocks to transmit the test packets. When the first test packet is received, a TRA starts analyzing the packet and continues the operation till the last packet is received. Therefore, two additional clocks are needed to analyze the responses for faults on handshake wires of the channels-under-test. Thus, $T_{it} = 40$ clocks. Putting a channel under test in an iteration improves link coverage metric (LCM). Figure 5.12 and Figure 5.13 show the LCM achieved in a round. The nodes with Deg=4, Deg=3, and Deg=2 map to

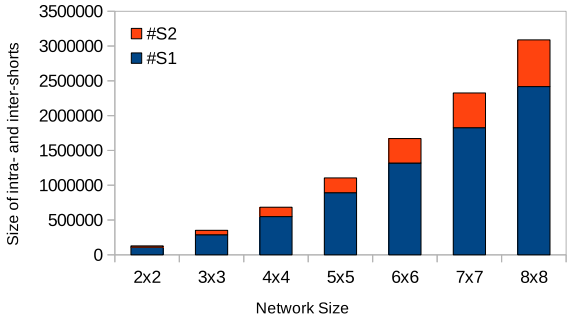


Figure 5.11: Size of intra- and inter-shorts injected in channels of the network for $n = 16$.

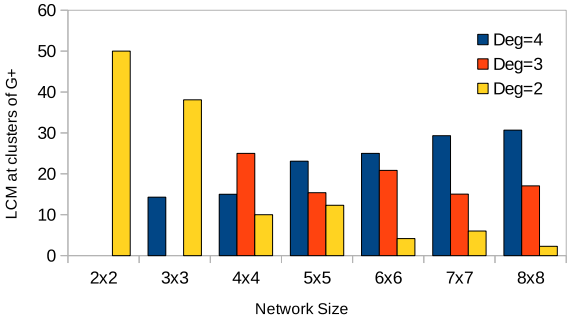


Figure 5.12: Link coverage metric achieved at first round in the network.

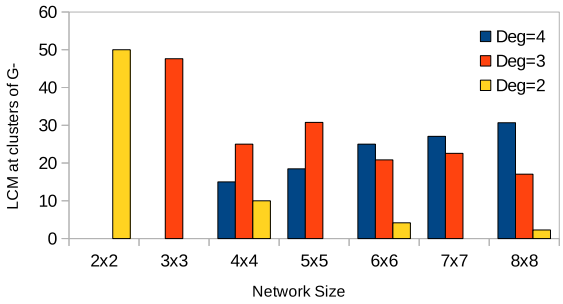


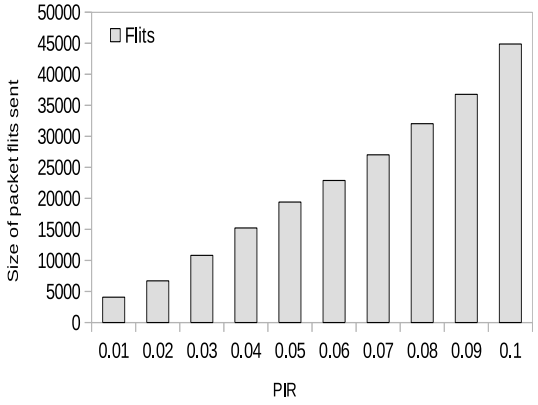
Figure 5.13: Link coverage metric achieved at second round in the network.

a test application by the nodes selected in I_1, I_2, I_3, respectively. As an example, consider Figure 5.10 where it may be observed that 50% of the channels are tested in R_1 while the rest 50% channels are tested in R_2.

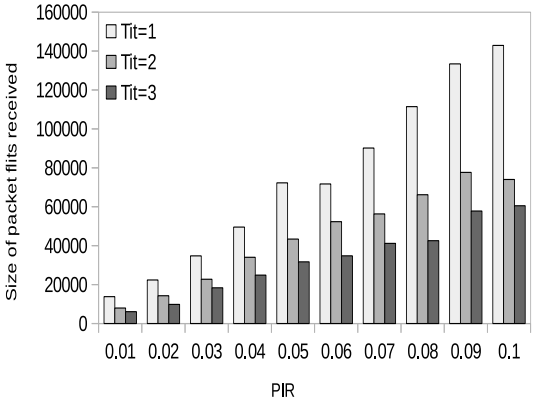
Note that a sufficient number of W1-sequences are produced when the TPGs exercise tests to excite short-faults on data, control, and handshake wires, followed by the analysis of responses made by the TRAs. Thus, decent fault coverage metric (FCM) is achieved. The FCM for the detected faults is analyzed in terms of payload, timeout, and misrouting errors. From Table 5.9, it is observed that the fault simulation achieves 100% FCM like the LCM. Therefore, all modeled short faults are detected by the TRAs successfully.

5.9.4 Performance Evaluation

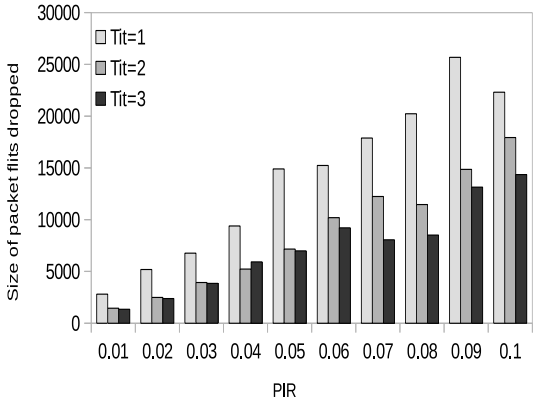
In different test iterations, various performance metrics such as throughput, average packet latency, and power (energy) consumed by a flit are observed. Figure 5.14 shows on-line evaluation of the proposed test solution at different traffic size on the 4×4 network with $n = 16$-bit channel. The traffic profile (packet flits) injected into the network is provided in Figure 5.14a, where the packets are transmitted using the XY-routing algorithm. Figure 5.14b,



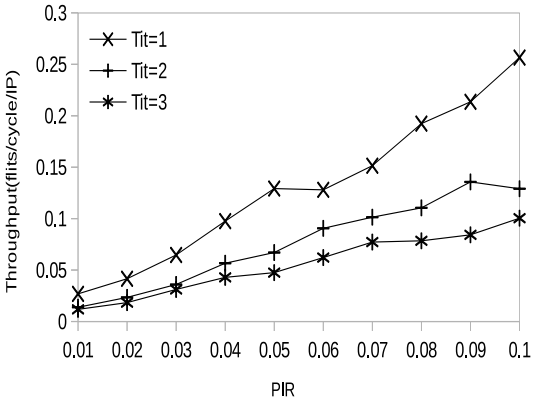
(a) Amount of packet flits injected.



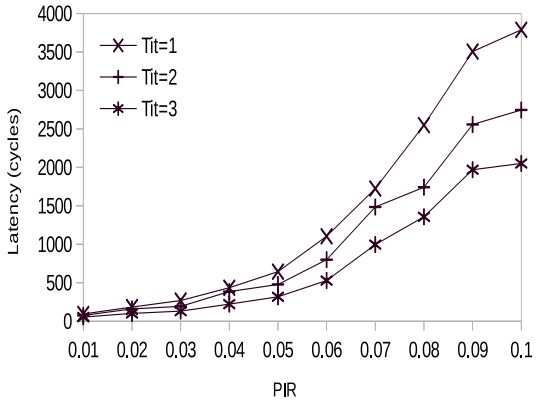
(b) Amount of packet flits received.



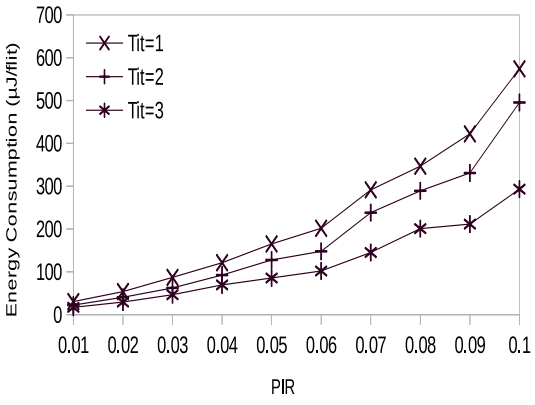
(c) Amount of packet flits dropped.



(d) Throughput behavior.



(e) Latency behavior of a packet.



(f) Energy consumed by a packet flit.

Figure 5.14: On-line evaluation of the proposed test solution at different traffic size on a 16-bit 4×4 network.

and Figure 5.14c provide the size of traffic received and dropped in the network. Note that data traffic traveling through a faulty channel may be duplicated and/or lost in the channel. These changes can be detected in terms of received and dropped traffic. Furthermore, the duplication and dropping of packets may continue when multiple faulty channels appear in the routing path. Figure 5.14d depicts the maximum traffic accepted per IP-block over the simulation period, i.e., the throughput. Besides, the size of received traffic and simulation period, the metric depends on faults on the channels and the pattern of the injected traffic. Figure 5.14e shows the variation of average packet latency. The metric determined by the transmission of the header flit to the receipt of the trailer flit of a packet depends on n and the traffic (load) on a channel along a routing path. In the test mode, the algorithm uses only one test packet and shifts successive packet-flits per clock. Therefore, packet-latency is 4-clock. Since the traffic increases with faulty channels experiencing shorts, packet-latency during simulation is degraded. The latency may also be degraded whence a router is busy with a test application and the arbiter in the router does not grant an incoming application packet to traverse it until the ongoing test application is shifted to the next iteration or completed. When the traffic is duplicated, more power is needed by the communication infrastructure to transmit these traffic. Figure 5.14f correlating to received traffic shows the variation of energy consumed by a packet flit.

5.10 Method Scalability

In the proposed solution, a test application is driven by the nodes in a cluster-set. An NoC can have at most six cluster sets (as stated in Section 5.3 and shown in Table 5.8). However, one or two additional cluster-sets may be obtained when the NoC is of irregular/indirect type,e.g., hybrid, reduced mesh network. Thus, the proposed cluster-set driven test model (C-Model) scales to general NoCs irrespective of size, channel width, and topology.

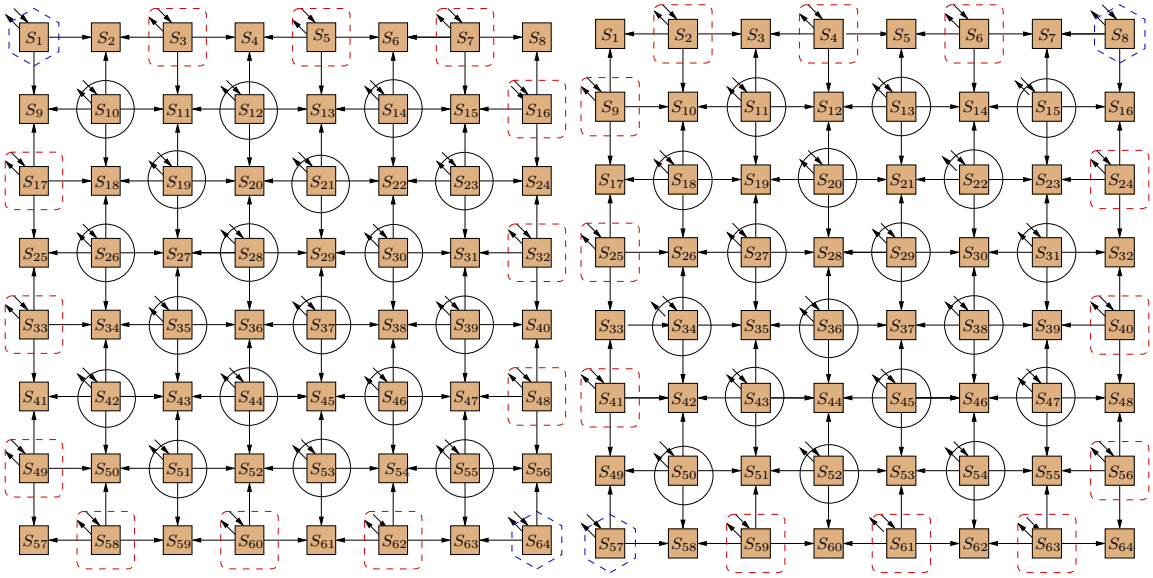
5.10.1 Scalability with NoC Size

In order to account for channel shorts in larger NoCs, the nodes are selected from a cluster-set that concurrently execute the test algorithm and lower the value of T_{NoC}. Now, 3×3–8×8 are included to illustrate the scalable behavior of the proposed solution on larger networks. The characteristics of these networks are shown in Table 5.5. For instance, the proposed solution is evaluated on a 8×8 network that consists of 352 channels. Such a network comprises six cluster-sets. Node-labeling is made in row-major order as shown in Figure 5.3. The nodes in a cluster-set that are in parallel execute the test algorithm in an iteration (Table 5.10). Correspondingly, R_1, R_2 test applications at different I_1, I_2, I_3 are shown in Figures 5.15a, and 5.15b, respectively. Test application R_1, I_1 at the nodes in a cluster-set $\mathbb{C}_4^+$ is demonstrated by the circled nodes as shown in Figure 5.15a. In the figure, the

Table 5.10: Cluster-set formation on a 8×8 network.

$\mathbb{G}$'s Variant	Cluster-set Type	Cluster-set Elements
$\mathbb{G}^+$	$\mathbb{C}_4^+$	$N_{10}, N_{12}, N_{14}, N_{19}, N_{21}, N_{23}, N_{26}, N_{28}, N_{30}, N_{35}, N_{37}, N_{39}, N_{42}, N_{44}, N_{46}, N_{51}, N_{53}, N_{55}$
	$\mathbb{C}_3^+$	$N_3, N_5, N_7, N_{16}, N_{17}, N_{32}, N_{33}, N_{48}, N_{49}, N_{58}, N_{60}, N_{62}$
	$\mathbb{C}_2^+$	N_1, N_{64}
$\mathbb{G}^-$	$\mathbb{C}_4^-$	$N_{11}, N_{13}, N_{15}, N_{18}, N_{20}, N_{22}, N_{27}, N_{29}, N_{31}, N_{34}, N_{36}, N_{38}, N_{43}, N_{45}, N_{47}, N_{50}, N_{52}, N_{54}$
	$\mathbb{C}_3^-$	$N_2, N_4, N_6, N_9, N_{24}, N_{25}, N_{40}, N_{41}, N_{56}, N_{59}, N_{61}, N_{63}$
	$\mathbb{C}_2^-$	N_8, N_{57}

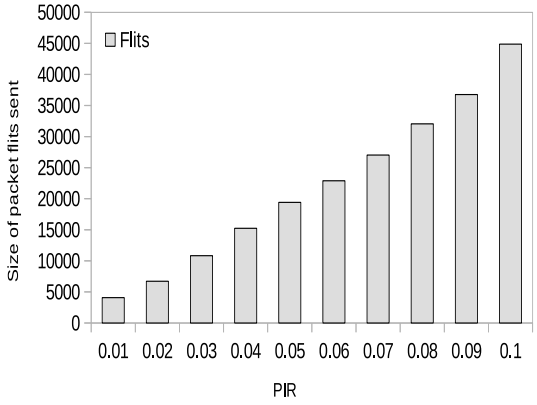
subsequent test applications in I_2, I_3 in R_1 are appropriately described at dashed square and hexagon nodes. Note that the descriptions of the nodes belonging to the cluster-sets $\mathbb{C}_3^+$, $\mathbb{C}_2^+$ are provided in Table 5.10. These three iterations I_1, I_2, I_3 altogether in R_1 cannot cover all the modeled short faults in the channels of 8×8 network. The test applications in I_1, I_2, I_3 is repeated in next round R_2 as described by the circled, dashed square, and dashed hexagon nodes (see Figure 5.15b). These nodes belong to the cluster-sets $\mathbb{C}_4^+, \mathbb{C}_3^+, \mathbb{C}_2^+$ respectively and are described in Table 5.10. Thus, six iterations are sufficient to test all channels at the cost of only 240 clock-cycles. Notice that the proposed solution always leads to six iterations for any $P \times Q; P, Q > 3$ network.



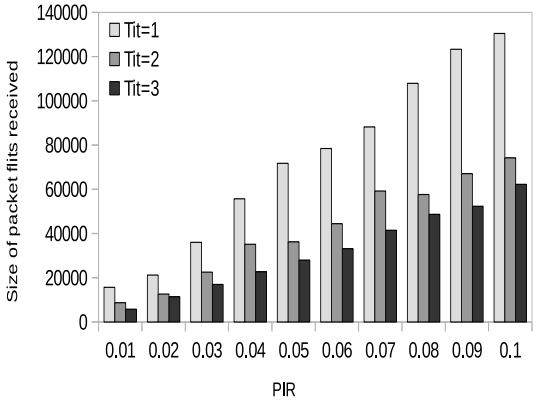
(a) Execution of the test algorithm by nodes in R_1.

(b) Execution of the test algorithm by nodes in R_2.

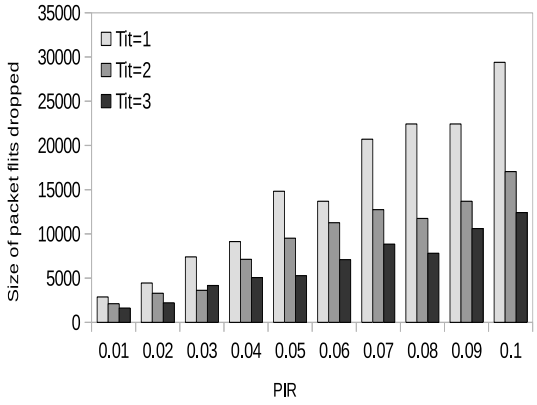
Figure 5.15: Application of the cluster-set driven test solution to illustrate its scalable behavior on an 8×8 network.



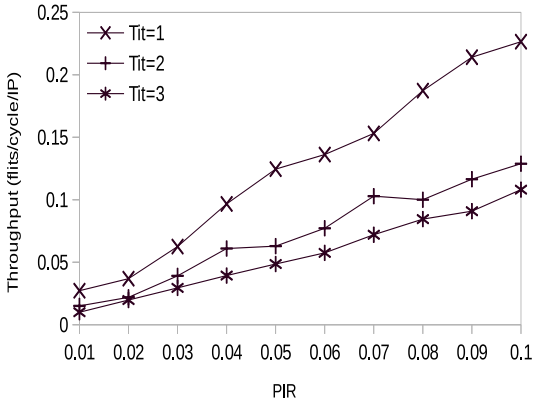
(a) Amount of packet flits injected.



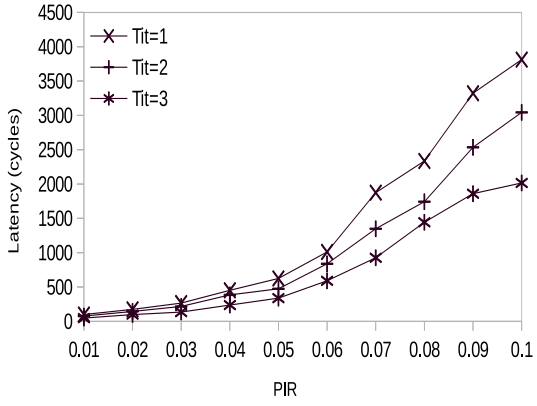
(b) Amount of packet flits received.



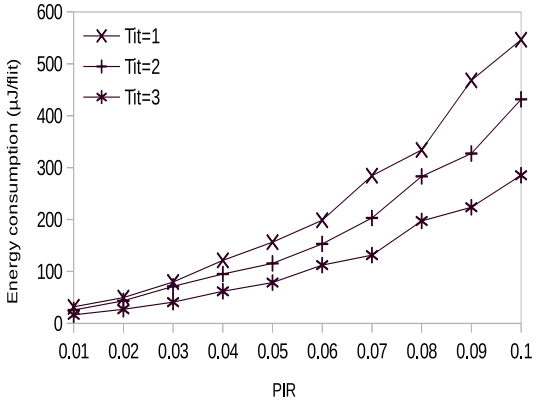
(c) Amount of packet flits dropped.



(d) Throughput behavior.



(e) Latency behavior of a packet.



(f) Energy consumed by a packet flit.

Figure 5.16: On-line evaluation of the proposed test solution at different traffic size on a 16-bit 8×8 network.

LCM-values achieved in a test iteration in round R_1 and R_2 on an 8×8 network and on other networks are depicted in Figure 5.12 and Figure 5.13, respectively. It is observed that 50% (30.68% in I_1, 17.05% in I_2, 2.27% in I_3) LCM is achieved in a round R_1/R_2. Thus, CLCM is achieved to 100%. Consequently, the FCM in terms of channel errors as found after carrying a fault simulation on the 8×8 network is provided in Table 5.9. The effect of channel-shorts is observed with the proposed test solution on performance metrics. Figure 5.16 shows the simulation results of the solution on the 8×8 network for $n = 16$. The simulations are performed at the same time at the range of PIR but injecting heavier traffic in the network. Other simulation parameters are chosen similarly.

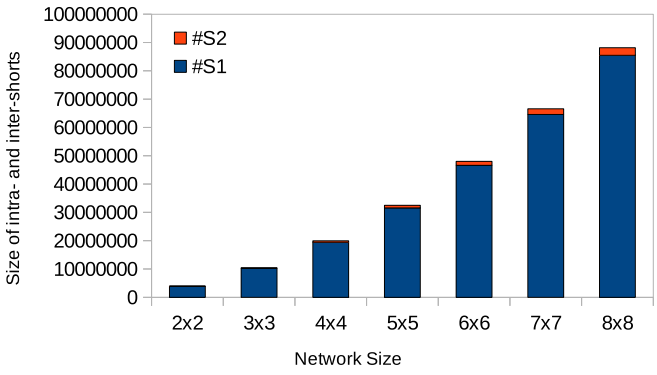


Figure 5.17: Size of intra- and inter-shorts injected in 32-bit channels in NoCs.

5.10.2 Scalability with Channel Width

Many high-performance computing applications, e.g., MPEG-2 video, as well as other networking applications [189], have high bandwidth requirements. The channel width of an NoC-based communication architecture varies with topology. The probability of short-channel faults increases in a wider channel. Therefore, a testing mechanism must scale with such channel configuration. The proposed C-Model also scales to all NoCs with channel width.

Let us consider $n = 32$ for the NoCs characterized in Table 5.5. The characteristics of these networks remain the same except the size of channel-wires #W. The values of #W of the networks are supplied in Column II of Table 5.12 whereas the hardware area overhead of TM_1, TM_2 in an IP core and a router is provided in Table 5.11 for a 32-bit channel. Detecting all shorts in a channel with $n = 32$, a TPG should produce 32 test flits (W1 sequences). The amount of intra- and inter-shorts, i.e., #S1, and #S2 injected on these channel-wires to be exercised during fault simulation are shown in Figure 5.17. As expected, #S (defined in Equation 5.4) increases exponentially with n; the T_{it} (also T_{NoC}) grows almost linearly. During the experiment, it is noted that a TPG takes 10 clocks to derive the W1-set

Table 5.11: Synthesis Results for the TMs in 32-bit channels.

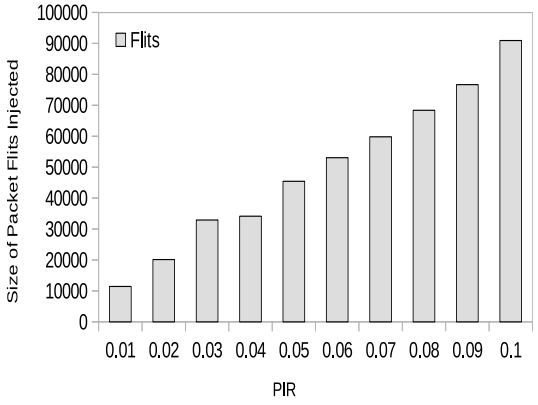
n	TM Unit	#LGs	RASoC	Xpipe	Core
	TPG	186	10%	12%	3%
	I-TPG	219	12%	14%	–
32	TRA-Core	125	5%	7%	2%
	TRA-Router	164	8%	9%	–
	TM_1, TM_2	311, 383	15%, 20%	19%, 23%	5%

Table 5.12: Fault coverage analysis on $P \times P$ NoCs at $n = 32$.

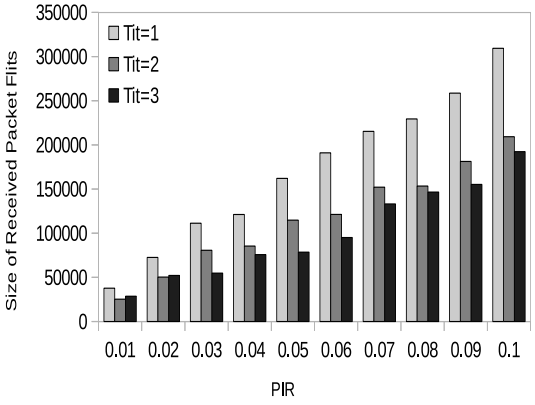
N/w Size	#W	T_{NoC} (clocks)	Payload Error(%)	Timeout Error(%)	Misrouting Error(%)	Total FCM(%)
2x2	512	156	62.673	18.236	19.091	100
3x3	1344	234	73.524	14.397	12.079	100
4x4	2560	468	68.57	15.757	15.673	100
5x5	4160	468	63.893	17.563	18.544	100
6x6	6144	468	61.726	21.465	16.809	100
7x7	8512	468	65.042	17.159	17.799	100
8x8	11264	468	62.284	16.682	21.034	100

including the associated routing information. Another two clocks are needed to organize this test set into 16 test packets. These packets are further transmitted over 64 clocks. TRA at a receiver takes two additional clock cycles to analyze the responses. Thus, $T_{it} = 78$ clocks. As shown in Table 5.8 that Tits are fixed in networks, T_{NoC} in these networks can be taken as shown in Column III of Table 5.12. Note that the LCM-value remains the same as shown in Figure 5.12 and Figure 5.13. The FCM-values achieved during fault simulation vary as shown in Table 5.12.

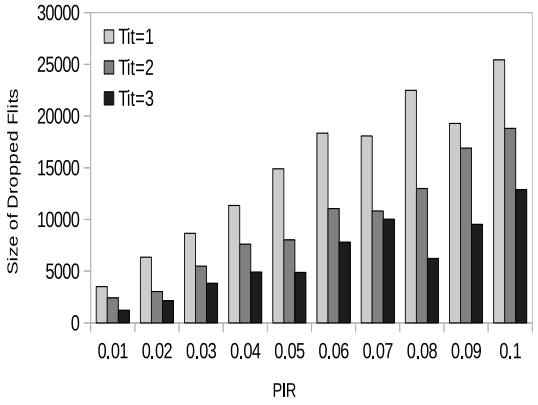
The proposed scheme is evaluated on a 4×4 network with 32-bit channels in order to observe the effect of short faults on performance metrics. The simulation is performed on the network with basic parameters provided in Table 5.6. However, the simulation period is doubled from 10000 cycles to 20000 cycles. Figure 5.18 demonstrates the on-line evaluation of the proposed solution on the network with 32-bit channels. When the channel-width increases, large data-volume can be transmitted (Figure 5.18a) in the network with less performance overhead. However, the number of short-channel faults increases and significantly affects network performance. A large amount of duplicated and misrouted packet flits (Figure 5.18b) is received in the network due to payload and misrouting error. In the sequence, many packet flits are lost (Figure 5.18c) due to timeout error which is further enhanced by the packet dropping failure mode. Figure 5.18e and Figure 5.18f show the behavior of average packet latency and energy consumed by a flit, respectively. As compared to a network with 16-bit



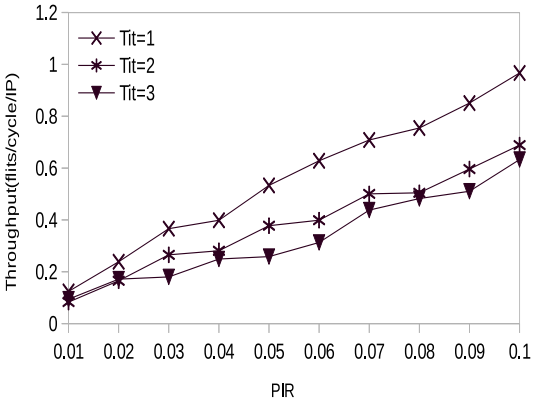
(a) Amount of packet flits injected.



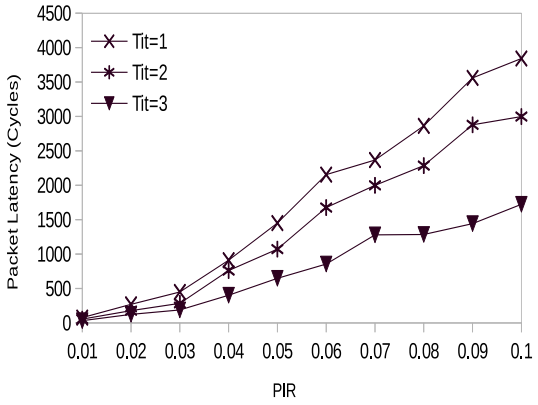
(b) Amount of packet flits received.



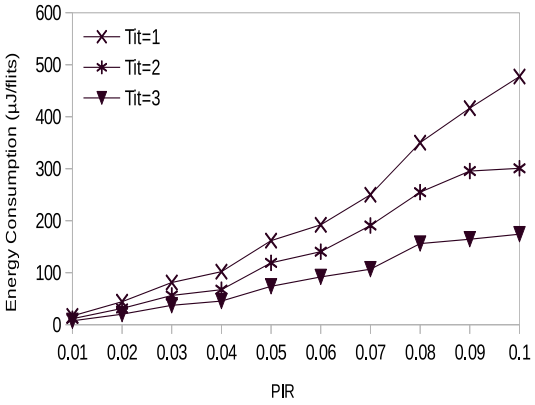
(c) Amount of packet flits dropped.



(d) Throughput behavior.



(e) Latency behavior of a packet.



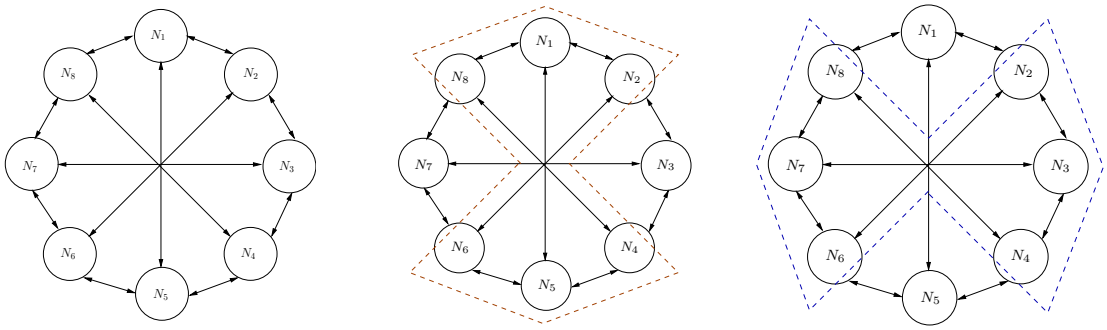
(f) Energy consumed by a packet flit.

Figure 5.18: On-line evaluation of the proposed test solution at different traffic size on a 32-bit 4×4 network.

wide channel, 3-4× higher traffic is injected into the network. However, the packet latency and energy consumption are increased to 20—26% and 11—17%, respectively.

5.11 Method Adaptability

The portability of the proposed C-Model is assumed beyond the limit of network size and channel width, as established in former two subsections. Now, the scalable behavior of the C-Model is seen on a topology other than meshes but involving same or fewer test iterations. It is clear that the *Tits* equals to $|\mathbb{C}^+| + |\mathbb{C}^-|$ found based outdegree of a node in a network. For example, a mesh network has six cluster-sets (*Tits*=6) as each node has a degree equal to 4, 3, or 2. But, there are few networks where each node has same degree. For example, each node in an octagon network (shown in Figure 5.19a) has degree 3. A designer can think of the test applications in the network as shown in Figure 5.19b and 5.19c in order to trade-off between test clocks and performance overhead. Figures 5.20a, and 5.20b show the test applications in I_1, I_2 in the R_1 and are done with the cluster-sets $\{N_1, N_4, N_6\}$, and $\{N_2, N_5, N_8\}$ respectively. Whereas, Figures 5.21a, and 5.21b show the test applications in I_1, I_2 in the R_2 and are done with the cluster-sets $\{N_2, N_4, N_7\}$, and $\{N_3, N_6, N_8\}$ respectively.

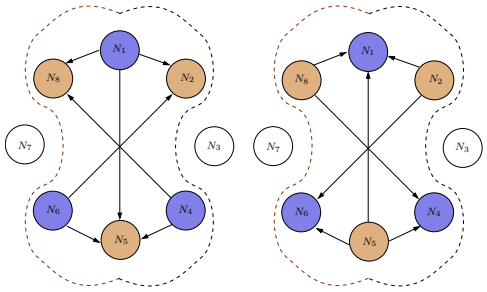


(a) General representation of an octagon network.

(b) Selection of first test region, i.e., Subnet-1.

(c) Selection of second test region, i.e., Subnet-2.

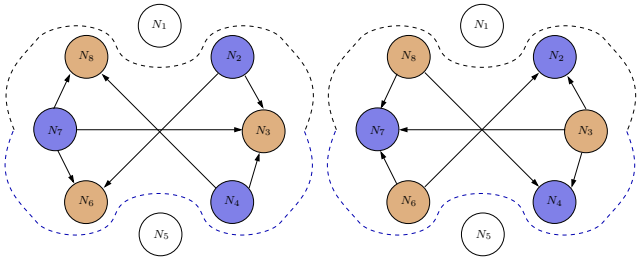
Figure 5.19: Abstract representation of a basic octagon network in addition to a test region selection at different test rounds on the network.



(a) $TR = 1, Tit = 1$.

(b) $TR = 1, Tit = 2$.

Figure 5.20: Application of the test algorithm in first test round on the octagon network.



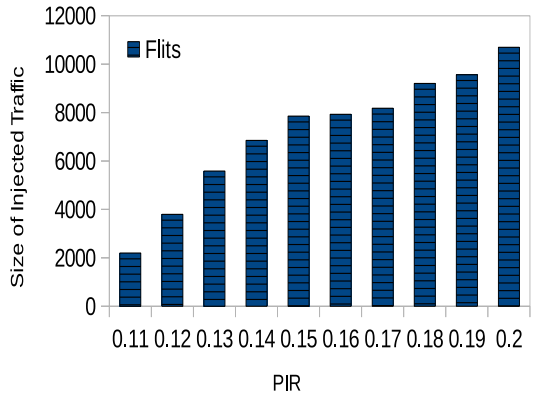
(a) $TR = 2, Tit = 1$.

(b) $TR = 2, Tit = 2$.

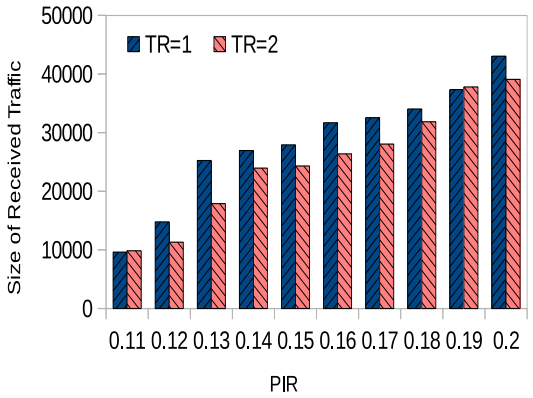
Figure 5.21: Application of the test algorithm in second test round on the octagon network.

The octagon network consists of 8 nodes and 40 channels, each of 16-bit. Therefore, the fault injection campaign needs to inject #S1=274720, #S2=65024, i.e., #S=339744 short faults for testing. As each node in the octagon network has degree 3, the area overhead of the TM_2 can be reconfigured and reduced up to 2–3% for a 16-bit and 32-bit channels. Considering the test time evaluation method discussed earlier, each iteration needs 40 clocks to detect shorts in 16-bit channels. Therefore, the T_{NoC} equals to 160 clocks since the testing of all channels can be completed in just four iterations. The fault simulation campaign is conducted in the sequence of testing DWs, CWs, and HWs in a test round. Let us consider the channels in Subnet-1 i.e., TR=1 shown in Figures 5.19b. In first fault simulation, the shorts exist in DWs. Then, #S1=40898, #S2=19344,i.e., #S=60242 shorts are detected by transmitting sufficient W1 test sequences and analyzing responses. In second fault simulation, shorts are assumed to be on CWs that extend the fault region to DWs. Additional W1 sequences are accommodated in control information ("bop", "eop" signals). The fault simulation campaign detects #S1=89908, #S2=26376 i.e., #S=116284 shorts. Finally, shorts are considered on HWs that extend fault region to both DWs and CWs. Therefore, extra W1 set is placed in the handshake communication information ("val", "ack" signals). The fault simulation detects #S1=178568, #S2=34496 i.e., #S=213064 shorts. All modeled short faults in channels in Subnet-1 in the round are detected resulting 100% fault coverage metric (FCM). However, this FCM is 62.71% at the network level i.e., in reference to total modeled short faults in the octagon network. These detected short faults are realized in the form of three types of channel errors. It can be observed that the fault simulation reports 68.21%, 15.23%, and 16.56% as payload error (PE), misrouting error (ME), and dropping/timeout error (TE), respectively which again cumulatively sums to 100% FCM. In the current test round, every channel wire has undergone the testing which results in 100% link coverage metric (LCM). However, at the network level, this LCM is 61.90% because 26 channels have been put in the test mode. The fault simulation is repeated similarly on the Subnet-2 i.e., TR=2 shown in Figure 5.19c to detect shorts in rest of the channels in the octagon network. Thus, FCM, as well as LCM achieved individually in the network, equals to 100%. A clear advantage of conducting the fault simulation in this manner is that channel short faults are diagnosed optimally resulting 100% coverage metrics at lowest test cost.

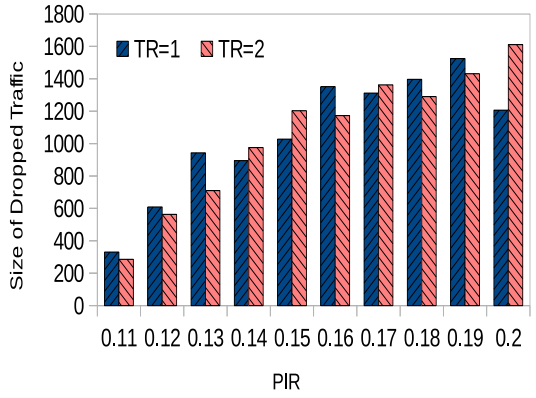
It is evident that channel short faults put an on-chip network into many system-level failure modes and consequently act as an agent to direct performance degradation in the network. The proposed scheme is now evaluated on the 16-bit octagon network and observe severe effects of these faults on standard network performance metrics: throughput, latency, and power consumption. To observe these metrics, the system simulation is performed using Noxim [22], a SystemC-based cycle accurate NoC simulator. Every time the simulation is continued for 10000 cycles including 10% of the cycles for collecting simulation statistics. It can be noted that the octagon network is configured into an equivalent 2×4 mesh network



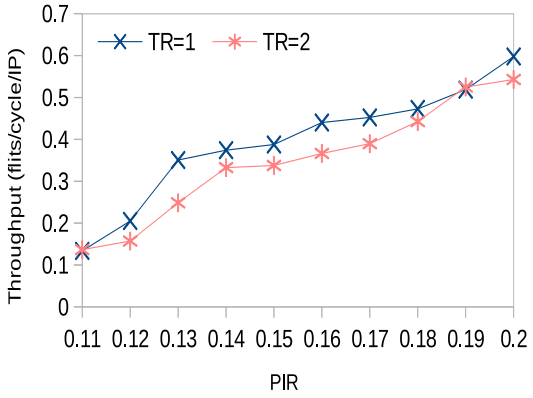
(a) Size of packet flits sent in octagon.



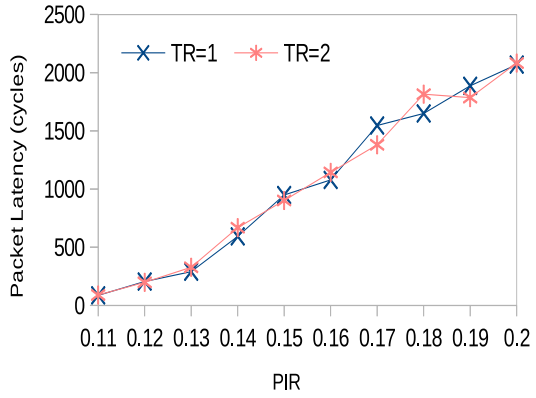
(b) Size of packet flits received in octagon.



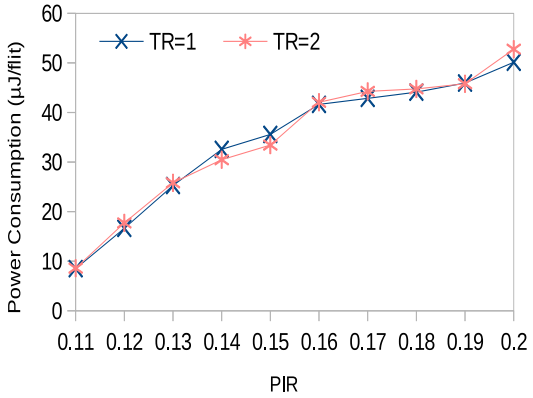
(c) Size of packet flits dropped in octagon.



(d) Behavior of throughput in octagon.



(e) Behavior of packet latency in octagon.



(f) Energy consumed by a packet flit.

Figure 5.22: On-line performance evaluation by the application of the proposed cluster-based test model at both test rounds (TR=1 and TR=2) on 16-bit octagon network.

with the shared channels between diagonally opposite routers. Figure 5.22 demonstrates the on-line evaluation of the proposed cluster-based model on the 16-bit octagon network. The traffic in terms of packet flits is injected at the packet injection rate (PIR) 0.11–0.20 into the network. The W1 sequences as application data (traffic) are packeted using the wormhole switching technique and each application packet is transmitted using the XY routing protocol. Size of application data injected is shown in Figure 5.22a. This traffic while traveling on the routing paths, may be affected by one or multiple faulty channels. Corresponding received and dropping flits observed in the network are provided in Figures 5.22b and 5.22c, respectively. Note that the received flits sum the original, duplicated, and misrouted flits. The size of received flits is seen nearly 3–4.5× of the injected traffic size. A fraction of injected traffic is lost due to packet timeout fault. However, this fault is enhanced when CWs that carry packet trailers experienced shorts. Generally, 5–8% of injected traffic is lost due to timeout. But, with faulty channels, this dropping is enhanced to 12–17%. Consequently, the behavior of performance metrics is changed and is provided in Figures 5.22d, 5.22e, and 5.22f. The throughput in NoC architectures generally depends on the traffic pattern. Here the uniform random traffic is preferred, which is often used in such systems. Accepted traffic as the throughput in the network further depends on the rate of traffic injection and the faulty channels over which the traffic is transported before reaching the destination nodes. These factors directly increase the packet latency. For instance, if multiple channels in a routing path are affected by short faults, more packet flits are duplicated resulting message contention which further increases packet delay and latency. Note that when the injected traffic approaches throughput limit, packet latency seems very high and even increases exponentially. Consequently, energy consumed by a packet flit is observed.

5.12 Comparative Study

The proposed solution presents an efficient mechanism for at-speed testing of short-channel faults in an NoC-based system. The solution is capable of handling various constraints such as hardware-area overhead, test clocks, and performance overhead. The cluster-driven model (C-Model) is compared here with existing test approaches- 2×2 neighborhood selection model (2×2-Model) [37–39], sequential router selection model (S-Model) [6,7], odd-even node selection model (OE-Model) [RPC-14], token-based two-hop model (2hop-Model) [RPC-13], and diagonal node selection model (D-Model) [RPC-6].

5.12.1 Benefits in Area Overhead

In 2×2-Model [37–39], the TMs in IP cores transmit test packets and analyze received packets. The TMs that transmit and analyze the packets are on the XY-paths. Each path has four hops. It is given that each TM takes around 40–50% of a router area depending on the 16-bit

and 32-bit channels. It is undesirable. In S-Model [6, 7], the shorts are considered in 16-bit interswitch channels only, for which a TM takes 16% of a router area. The area overhead is increased to 19% when shorts are extended to local channels. For 32-bit channels, the area overhead becomes 23%, and 28% when shorts are assumed only in interswitch channels, and both in interswitch and local channels, respectively. Therefore, the S-Model saves 17–22% as test area overhead over the 2×2-Model. From Table 5.7, it is seen see that a TM on average takes 15–18% and 20–23% area on a router for 16-bit and 32-bit channels. Thus, the proposed approach saves router area by 4–5% over the S-Model, and 25–27% over the 2×2-Model.

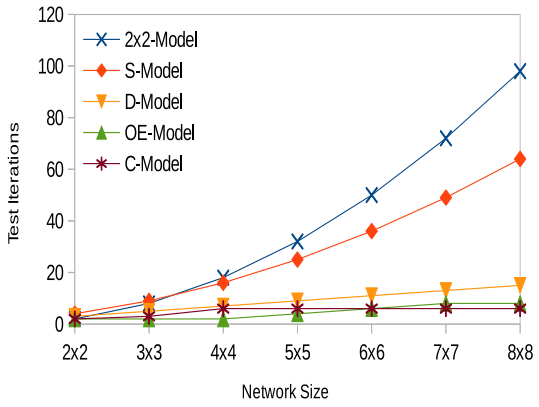


Figure 5.23: Comparison on test iterations needed by a set of test models..

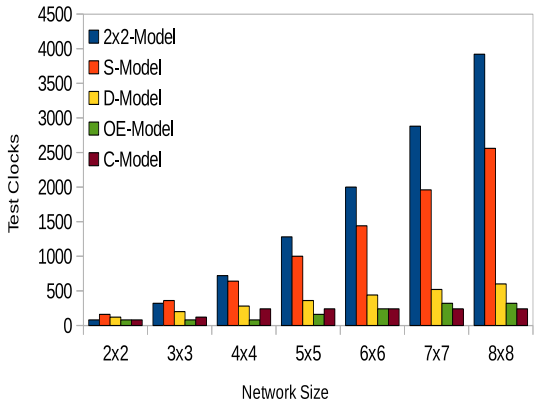


Figure 5.24: Comparison on test clocks needed by a set of test models.

5.12.2 Benefits in Test Clocks

The proposed C-Model is analyzed to study the benefits of test clocks for a 16-bit channel for $P \times Q$ networks. In order to detect the shorts in channels of a larger $P \times Q; P, Q > 3$ network by the 2×2-Model, multiple 2×2 test configurations are applied in a round. Each round takes 250 clocks. It is seen that all channels can be tested in just four rounds and costs 1000 clocks irrespective of the size of the network. However, testing of the channels in the on-line mode and putting a 2×2 subnetwork in the test mode in a $P \times Q$ networks leads to $Tits$=$(P-1) \times (Q-1)$. Correspondingly, the test time will be $250 \times (P-1) \times (Q-1)$ clocks. On the other hand, the test mechanism needs $80 \times (P-1) \times (Q-1)$ clocks. Therefore, 250-80=170, i.e., 68% clocks per iteration are saved. In the S-Model, only one router is selected at a time in order to test its I/O channels for shorts. The test mechanism requires two phases: one for input channels and another for output channels, that costs a total of 80 clocks. Therefore, overall test cost will be $80 \times P \times Q$ which is comparatively very high. Hence, the cost associated with the S-model does not scale well with network-size. In the OE-Model, a subnet of 16 nodes at a time is selected in test mode for a $P \times Q$ network. The model works

well when the number of nodes $|P \times Q| > 16$. Otherwise, test mechanism becomes similar to that for off-line mode. Hence, $Tits = \lceil |P \times Q|/16 \rceil$ for a $P \times Q$ network. Now, in the subnet of 16 nodes of a $P \times Q$ network, the test mechanism is executed twice to cover channel-shorts and it costs 80 clocks. Therefore, the OE-Model takes at least $80 \times \lceil |P \times Q|/16 \rceil$ clocks to detect short faults in all interswitch and local channels. Although the OE-model appears to be time-efficient, it takes longer test time when $P, Q \geq 13$ in reference to the D-model. This is because $Tits$ in the D-Model vary with the number $P + Q - 1$ which is very less than $\lceil |P \times Q|/16 \rceil$ for large $P \times Q; P, Q \geq 13$ networks. Thus, the D-Model takes $40 \times (P + Q - 1)$ clocks on a $P \times Q$ network. Now, the C-Model takes at most $Tits = 6$. Consequently, the test cost is 240 clocks. It shows that C-Model reduces the test time significantly. Thus, the C-Model is much faster (over 21×) and subsequently, it saves test clocks up to 12.50–95.32% over the existing models for a set of $P \times Q$ networks listed in Table 5.5. Figure 5.23 shows a comparison of $Tits$ among the models. One can easily estimate the value of T_{NoC} when the test algorithm is applied to the models (see Figure 5.24).

5.12.3 Benefits in Performance Overhead

The fault simulation is carried out for the C-Model and prior test models on a set of 16-bit $P \times Q$ networks described in Table 5.5. Figures 5.25, 5.26, and 5.27 depict comparative results on the FCM-values achieved by the test models in terms of the channel errors when the proposed test mechanism is adopted. For example, the 2×2-Model provides 73.05%, 10.26%, 16.69% as payload, misrouting, and timeout errors, respectively on a 16-bit 4×4 mesh network.

As the number of test iterations in prior models are higher, it is more likely that many application packets pass through multiple test iterations. Since channel shorts cause packet duplication and other failure modes, the duplicate packets need to use network resources before reaching the destination. Consequently, the packet latency and energy consumption increase. On the other hand, the C-Model requires a fixed number of iterations, which is very less compared to those of other models (Figure 5.23). An improvement on packet latency as well as energy consumption by a flit can be observed as shown in Figures 5.28, and 5.29, respectively. For example, the 2x2-Model, S-Model, OE-Model, 2hop-Model, and D-Model involve 39.63% 40.16%, 30.74%, 29.67%, 26.69% more latency than the proposed C-Model to transport application packets on a 16-bit 8×8 network. On the other hand, these models consume 30.31%, 34.20%, 23.61%, 26.17%, 21.68% extra energy, respectively than the C-Model for transporting a packet flit. Thus, the proposed C-Model reduces packet latency up to 19.47–40.16% and energy consumption by 17.57–34.20% for the NoCs studied in this chapter. Note that these improvements will also increase with the NoC-size.

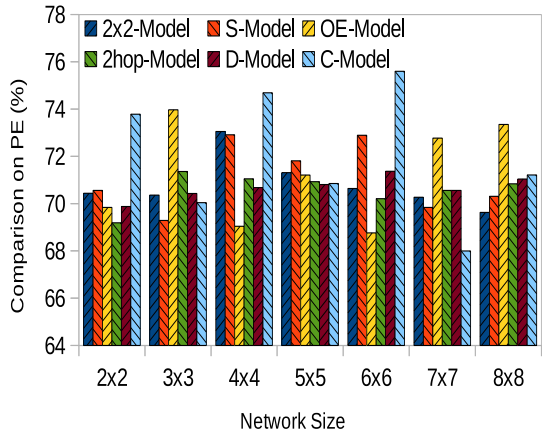


Figure 5.25: Comparison of payload error by a set of test models.

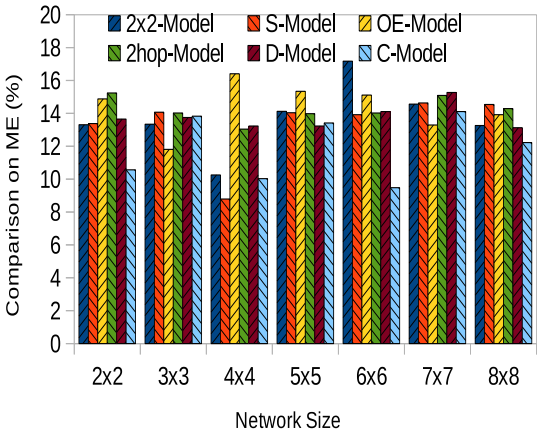


Figure 5.26: Comparison of misrouting error by a set of test models.

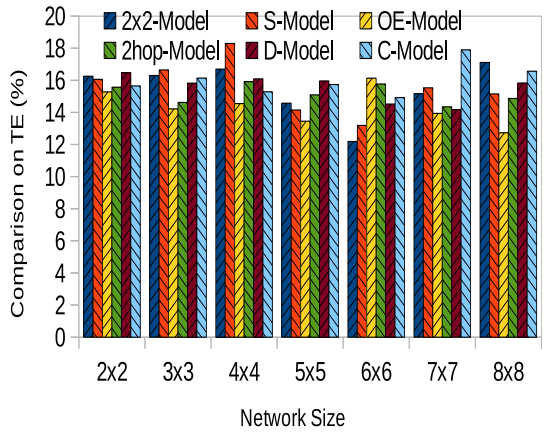


Figure 5.27: Comparison of timeout error by a set of test models.

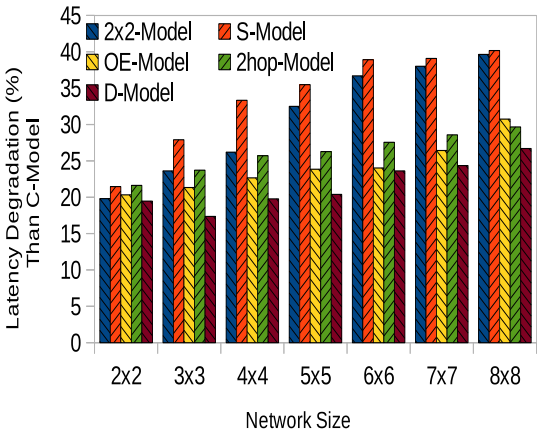


Figure 5.28: Comparison of latency degradation by prior test models.

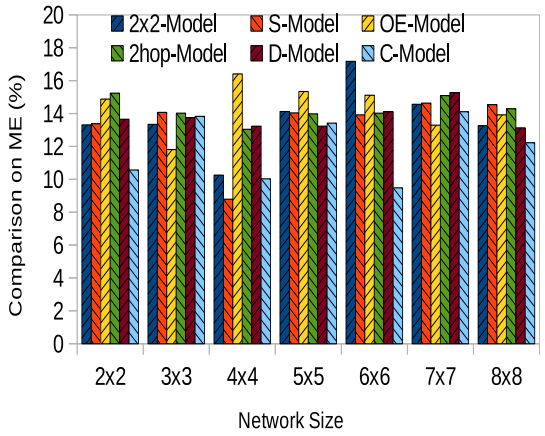


Figure 5.29: Comparative extra energy consumption by by prior test models.

5.13 Limitations

The compact implementation of versatile and dense on-chip interconnection networks, i.e., NoCs on a die has emerged as the technology of choice for multi-core computing. However, because of the increased density, the communication channels in the NoCs often suffer from intra-channel and inter-channel short faults that belong to the set of manufacturing faults. One of the major bottlenecks for NoC-based communication systems is post-manufacturing testing of different components, such the channels. While the cores perform computation, defect-free channels are needed for reliable data exchange. Short-channel faults in an NoC often cause system-level failures that may have the significant impact on its performance. Consequently, these channel-shorts degrade the yield and jeopardize the reliability of the overall system. This work proposes a cluster-based distributed test scheme for on-line testing of short faults in NoC channels. The nodes in a cluster set are appropriately scheduled to reduce test time. The approach scales to larger NoCs irrespective of the size of the network and channel width. Fault simulation shows that the proposed cluster-driven scheme is capable of detecting all modeled channel-shorts. On-line evaluation of the scheme also reveals the extent of the impact that the faults will have on various performance metrics for large traffic. Despite the several advantages, the proposed cluster-set driven test application offers following disadvantages.

- **Subnet Selection:** The decomposition of an NoC into cluster sets suggests that six sets are needed to test the channels of direct networks. However, there are many indirect networks which are custom-specific. In that case, one or two additional cluster sets are needed to cover the faults in all channels of the NoC.

- **Test Iteration:** With the increase of the cluster sets, the number of test iterations is linearly increases.

- **Test Time:** The overall test time incurred by the test algorithm in a test iteration primarily depends on the number of test iterations and the time taken in an iteration. Therefore, the overall test time grows with the increase of the test iterations.

- **Test Energy:** In the first iteration of both test rounds, most of the channels undergo the testing, i.e., the test application, in this case, puts the maximum part of the NoC busy in the testing. Therefore, network resource utilization (e.g., dissipated packet energy), in this case, is much higher than the other test iterations in a round. It is expected that resource utilization should be similar to a test iteration or round. Like the previous works, the test scheme does not provide any knowledge about the network resource utilization during testing a component (here channels) in the NoCs.

- **Temporary Faults:** Till now the proposed test mechanism does not include the test of a temporary fault, e.g., transient faults in the channels.

These issues are targeted in the next chapter providing a cost-effective test solution that addresses both manufacturing and transient faults in NoC channels and an estimation scheme for network resource utilization for testing a channel.

5.14 Conclusion

In tandem with the technology advancement and subsequent evolution of NoC layouts, a cost-effective test scheme has been proposed to meet the requirement of high-speed testing of such networks. A new test algorithm has been developed to address both intra- and inter-shorts in data, control, and handshake wires of interswitch and local channels in an NoC. The proposed method is based on a decomposition technique that splits an NoC architecture into two instances, which are further partitioned into various cluster-sets. This scheme enables all nodes of a cluster-set to execute the test algorithm simultaneously, in one iteration. The proposed algorithm has been implemented and evaluated on several 2D-mesh NoCs by injecting short-channel faults. High-speed testing can be achieved by transmitting the test packets from a node and analyzing the output responses at the neighboring nodes. Experimental results establish the effectiveness of the proposed scheme in terms of area overhead, test clocks, coverage metrics, and performance metrics. Additionally, system simulations have been performed to observe the effect of shorts on network performance. The scalable behavior of the scheme is also demonstrated for NoCs that have different sizes, channel widths, and topologies. The proposed cluster-driven testing significantly improves hardware overhead, test time, packet latency and energy consumption, particularly for large-size NoCs.

Chapter 6

A Test Time and Energy Optimized Scheme for On-Chip Channel-Faults

6.1 Introduction

With the continuous advancements in nanometer technology, last decade has witnessed a compulsion for the demand of high-performance computation and communication by many applications. Consequently, the practice of cost-effective design of modern, complex electronic systems, e.g., SoCs have dramatically been strengthened. In the nanoscale era, a global bus-based SoC system does not enable high-performance communication if the IP cores are increased. For systems in many communication and computing application domains like pervasive and parallel communication, high-end multimedia, these buses do not delightedly reach the expected performance due to communication bottleneck. The network-on-chip (NoC) has emerged as a holistic solution in the place that has brought the concept of computer networking mechanism in the basic operation. The high bandwidth requirements simultaneously are visibly moved with allying huge metallic wires in channels of NoC architecture [2, 3, 38, 52] resulting high wire density.

The channels (both interswitch and local) are the only transmission medium between communication parties in an NoC system. As the technology scales down, NoC channels become more sensitive to growing number of logic level manufacturing (permanent) faults such as shorts. So, fabricating such NoCs without any defect in channels is a major challenge. On the other hand, short faults in channels bring the network into various system level failure modes, such as packet duplication (or overloading), misrouting, delaying, and dropping (loss). The NoCs fabricated using nano-scale technology may show transient behavior that sometimes manifests in permanent faults resulting advancements in the failure modes [149]. Consequently, the performance of the network is significantly influenced by permanent and transient faults or even the chip becomes useless. In particular, the growing size of short

faults including contemporary transient faults in NoC channels has led to growth in reliability needs as well as pressure for yield improvement.

Faults in the channels of an on-chip interconnection network, i.e., NoC are generally referred as the on-chip channel faults. Handling a channel in the network where channel-wires experience faults, either of the following practices may be encouraged. By the first approach, the faulty channel-wires are replaced by spare non-faulty wires. This replacement mechanism becomes sometimes costly as well as adds area overhead. The approach as anticipated cannot be a choice for a circuit designer. By the second approach, faults in channels must be tolerated, i.e., the network is allowed to accommodate faults in channels. The scheme certainly employs a fault-tolerant routing scheme [26, 46, 48, 102] that deals with a faulty channel so that the network may still be operational. With a fault-tolerant routing scheme, the wires affected by a fault in a channel are avoided and alternative fault-free wires are chosen to transmit application data packets. One basic prerequisite of the aforesaid approaches is the identification of faulty wires in a channel prior to their implementation for taking counter measurements against the channel's faults. It is seen from the literature that most of the existing fault-tolerant schemes in NoCs do not come up with a testing mechanism [6, 7].

In order to guarantee outgoing quality while not sacrificing yield and reliability, the functional units of NoC-based systems must undergo a test at the pre- and post-manufacturing stages. Testing such units account for a large portion of the overall manufacturing cost. The main reasons include the pressing demand for zero defective parts-per-million, increasing frequency of operation, the high levels of integration, limited controllability and observability of embedded blocks, integration of heterogeneous devices onto the same substrate, requirement for specialized, high-cost equipment, new defect mechanisms, excessive process variations occurring in advance technology nodes, and long test times, etc. In this chapter, the proposed solution addresses short faults in channels that grow exponentially with the channel-width. So, the fault is more dangerous and responsible for performance degradation and significant resource consumption compared to other faults like stuck-at, cut, etc. Therefore, devising a cost-effective test solution with the capability of detecting channel faults (e.g., shorts) and identifying faulty channel-wires in NoC-based systems is mandatory.

Rest of the chapter is organized as follows. Motivation and contributions to the current work are presented in Section 6.2. Section 6.3 describes the proposed test mechanism followed by the fixed test round oriented test-scheduling in Section 6.4. The latter part of Section 6.4 additionally presents the energy model that may be used to account the energy needed in testing the channel-shorts. Section 6.5 provides simulation results. Section 6.6 establishes scalable behavior of the solution whereas the solution-adaptability is demonstrated in Section 6.7. Section 6.8 gives detailed comparative study to refer the benefits gained by the proposed solution. Basic limitations of the present solution are indicated in Section 6.9. Section 6.10 concludes the chapter.

6.2 Motivation, Problem Formulation, and Contributions

The motivation of the current work is having an on-line distributed time optimized test solution to address permanent faults on data, control, and handshake wires in channels, and analyze their impact on network performance based on XY routing. Here, the channel shorts employing the test algorithm proposed in Chapter 5 are taken in the analysis. Simultaneously, the ability of the test method is extended for detecting other channel-faults, such as packet deadlock, transient faults, and so on. It is seen by the prior schemes that unequal network resources, such as energy dissipation are utilized during testing of the channels in a test round (or iteration). Therefore, a trade-off between the test rounds (or iterations) and resource utilization must be accompanied in the current test solution. It can be achieved through a convenient test scheduling that keeps a part of an NoC in test mode. In the proposed scheduling scheme, a network is partitioned into four equal subnetworks (subnets). Application of the test algorithm on a subnet is treated as the test round. Each test round goes through two test iterations. At the first iteration, channels from all odd nodes while the channels from all even nodes in the second iteration undergo the testing. A test energy model is proposed and can be applied to a subnet to account the amount of energy dissipated by the test packets. Decomposing an NoC architecture into four subnets, and further executing the test algorithm at odd/even nodes in a subnet improves the test parallelism, which is one of the most important performance factors for large-scale networks and scales the solution with network size, channel width, and topology.

The proposed test model is evaluated for various quality metrics like hardware area overhead, test time (clocks), test and fault coverage, and performance in terms of throughput, packet latency, and energy consumption by a packet flit on a set of networks. Synthesis results show that low silicon-area overhead is required to execute the test mechanism. It is interestingly seen that the proposed solution consumes equal test clocks for all NoCs in the on-line mode till the channel-width remains unchanged. Simulation results show that the proposed test model achieves 100% test as well as fault coverage metrics. Simultaneously, the on-line evaluation of the model in terms of various network performance metrics at voluminous traffic size reveals the significant effect of channel-shorts. The proposed solution improves the aforementioned quality metrics than the existing works on the set of networks. It is seen that area overhead is reduced in the range from 4–28%. The test time evaluation on the networks shows that the proposed test solution is up to 16× faster. Further, the simulation results show a significant improvement in the performance overhead, such as packet latency and energy consumption. For instance, the proposed solution needs 14.98–50.67% lesser packet latency and 6.83–43.89% lesser energy consumption of a packet flit as compared to prior schemes. Note that, these improvements grow with network size.

Thus, the multi-fold contributions of this work are listed as follows.

- Detection of intra- and inter-channel shorts on NoC channel-wires. It ensures the health status of the channel-wires. Further, it identifies channel-wires as being either fault-free or faulty, i.e., performs diagnosis.
- Extending the proposed test mechanism for detecting other channel faults like transient faults.
- Test scheduling to lower the test time and ensure the proposed solution to be time-independent of the network size and topology.
- Test energy model for the packets used in testing the on-chip channel-faults.
- Evaluation of the proposed solution in terms of hardware area overhead, test time, and different coverage metrics.
- Studying the effect of the short faults in terms of various well-known performance metrics at sizable traffic.
- Establishing scalable behavior of the proposed solution with respect to network size and channel width.
- Evaluation of the proposed solution on an octagon network to establish the solution-adaptability feature.

Table 6.1: Necessary symbols used in this chapter.

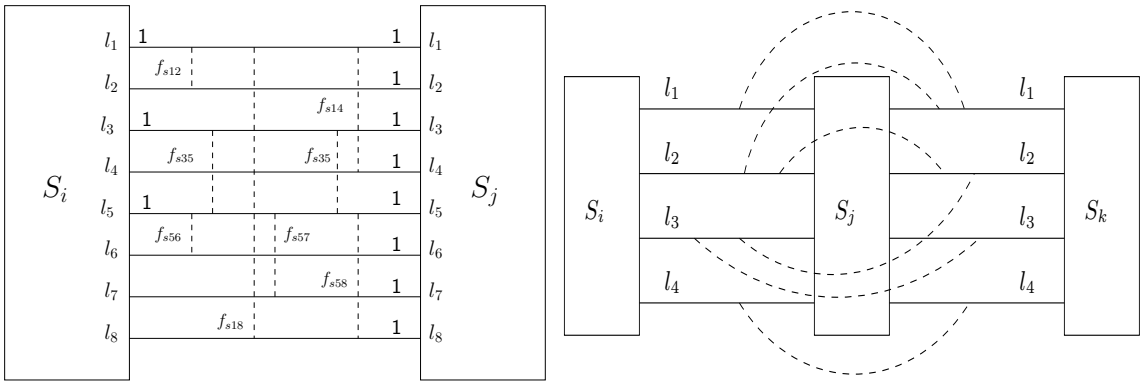
Symbol	**Meaning**
S_i	An *i-th* router; $i \in \mathbb{N}$.
C_i	The dedicated core of the S_i.
N_i	An NoC node that comprises of the pair $< S_i, C_i >$.
n	The bit-width of a channel and commonly represents the number of single bit wires in the channel.
l_a	A channel-wire; $1 \leq a \leq n$.
f_{sab}	A short fault occurred between the channel-wires l_a and l_b.
g	A fault group that occurs among a set of channel-wires.
x_g	The number of occurrences of the g for intra-channel shorts.
y_g	The number of occurrences of the g for inter-channel shorts.
m	The number of channels shared by a node.
$\#S1$	The size of short faults in a channel.
$\#S2$	The size of short faults between among the channels of a node.
$\|R\|$	The number of routers in a network.
$\|\mathsf{C}\|$	The number of channels in a network.
$\#S$	The size of channel-short faults that may be encountered in a network.

6.3 Proposed Test Mechanism

In this section, a distributed, time-optimized test mechanism is presented for detecting permanent faults like intra- and inter-shorts in general NoC channels. Also, the proposed test mechanism is extended to detect other channel faults in the NoC like packet deadlock, transient faults, etc.

6.3.1 Testing of Channel-Shorts From a Node

The basic test algorithm in detecting intra- and inter-shorts in channels of a node in an NoC-based communication system and diagnosing the faulty channel-wires which can later be exercised by a fault-tolerant routing scheme, is inherited from Chapter 5. The channel-shorts are shown in Figure 6.1. Figure 6.1a demonstrates the existence of intra-channel shorts in the channel (S_i, S_j). On the other hand, the inter-channel shorts between the channels (S_i, S_j) and (S_j, S_k) are shown in Figure 6.1b. The symbol f_{sab} represents a short between two wires l_a, l_b, e.g., f_{s12} shown in Figure 6.1a sights an intra-short between the channel-wires l_1, l_2 in (S_i, S_j). Considering the channel-short fault model in Chapter 5, the size of intra-shorts #S1 in a channel with width n, and inter-shorts $\#S2$ at a node N_i of m channels are defined in Equations 6.1, and 6.2, respectively. It can be seen that the size of shorts grows exponentially with the channel-width. Equation 6.3 defines #S, i.e., total size of the channel-shorts under the proposed fault model in an NoC of $|\mathbb{C}|$ channels and $|R|$ routers.



(a) Intra-shorts in the channel (S_i, S_j).

(b) Inter-shorts between channels (S_i, S_j) and (S_j, S_k).

Figure 6.1: Single and multiple manifestation of intra- and inter-shorts in NoC channels.

The illustration of the proposed test solution begins with a 3×3 NoC that has 24 interswitch (router-router) and 18 local (router-core), i.e., 42 channels. Each channel for instance has *n=16-bit* width that can be assumed as a *16-bit* data bus. Assuming the $x_g = 1, y_g = 1$, the proposed test algorithm should encounter #S1=288456 as intra-shorts, #S2=63472 as inter-shorts, i.e., #S=351928 as total shorts. These faults are injected into the channels of the 3×3 NoC prior to execution of the algorithm.

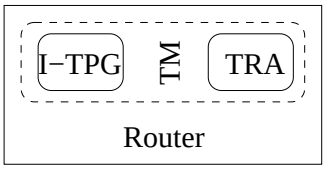
$$\#S1 = \sum_{2 \leq g \leq 5} \frac{n!}{g!(n-g)!} \times x_g \qquad (6.1) \qquad\qquad \#S2 = \frac{(mn)!}{2!(mn-2)!} \times y_g \qquad (6.2)$$

$$\#S = \#S1 \times |\complement| + \sum_{i=1}^{|R|} \#S2 \qquad (6.3)$$

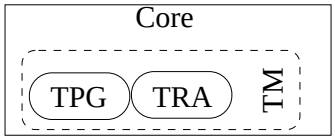
Table 6.2: A test packet organization with W1 sequences for detection of short faults in n-bit channels.

Header	Payload Flits	Trailer
Flit	$W1$	Flit

The basic idea behind the test algorithm works by sending test packets from a sender and analyzing the received packets at a receiver in a network. It is noted that during testing, the functional mode of the underlying network is used except a subset of channels are put busy in testing. In the present solution, the channel-shorts are exemplified with a finite test set that contains walking one (W1) patterns. The test pattern generator (TPG) block in a core or integrated TPG (I-TPG) block in a router generates the test set including header and trailer information in terms of flits. Another hardware block known as test response analyzer (TRA) is designed to analyze the test responses. This block has two units, one is fault diagnosis module (FDM) and another is the signal generator (TSG). These block pair, i.e, $< TPG, TRA >$ at a node is called a test module (TM) and take a small part as the chip area in the node. Figure 6.2 and 6.3 represents a basic test TM unit and TRA unit at a node, respectively. The wormhole switching is used to organize raw test set into test packets by embedding header and tail flits. A typical packet format is shown in Table 6.2. The header and trailer flit generated at routers carry necessary routing information with the address of multiple destinations (neighbors). The routing information of a test packet at a core contains the address of its neighbor router only as the destination. Therefore, packet headers at core and router in a node are different. However, the packet trailer may be the same. The test packets at a node N_i are unicasted on the local channel from the core to the linked router. At the same time, similar test packets are multicasted on the local and interswitch channels from the router of N_i to its linked core and routers of neighbor nodes. Figure 6.4 demonstrates a schematic view of unicast/multicast based test packet transmission from a node. The solid arrows represent the multicasting of the test packets on the channels (local and interswitch) from a router and the dashed arrow represents the unicasting of the test packets on a local channel from a core. Received packets as responses at receivers are analyzed to detect faults on channel-wires and also the receivers report a channel-error. The shorts are targeted on data, control, handshake wires in channels of a node. Therefore, the test algorithm at a node executes in the sequence of testing the data wires (DWs), control wires (CWs), and handshake wires (HWs) of channels from the node.



(a) Test block at a router.



(b) Test block at a core.

Figure 6.2: Representation of test module blocks embedded in router and core of node for detecting short faults in NoC channels.

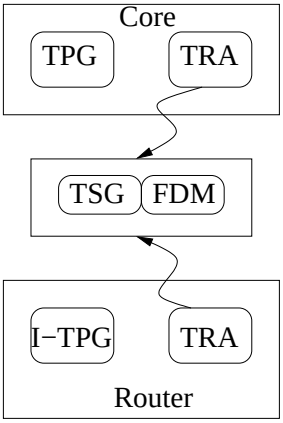


Figure 6.3: Fault analysis components of a TRA unit placed in nodes.

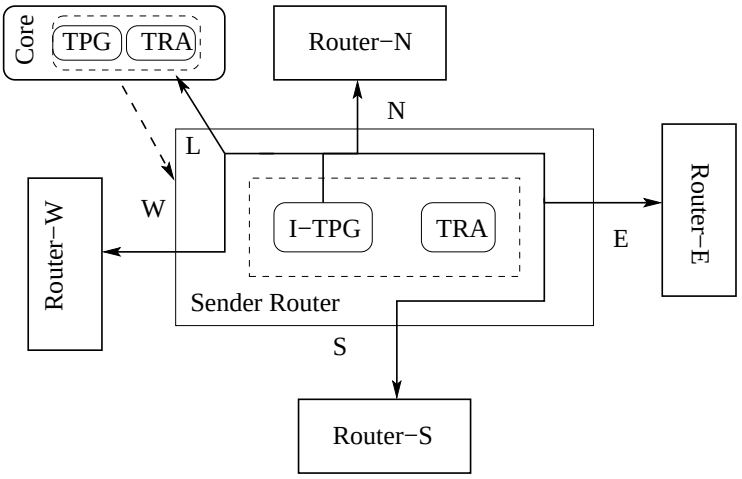


Figure 6.4: Application of test packets at an NoC node.

6.3.1.1 Fault Detection

Let us consider the test of DW set of the channels from a node. Sufficient W1 test set is placed into the packet payload field. The header and trailer flits are embedded to the test set. Header flits carry beginning of packet (bop) and associated routing information for reservation of channels. Trailer flits carry the end of packet (eop) and related routing information to release reserved channels. The payload flits on data-wires of the reserved channels follow the headers and are followed by the trailers. The I-TPG multicasts test packets to its neighbor routers and linked core. Simultaneously, the TPG of the core unicasts test packets to the router. Shifting of test packet flits is done clockwise. Each flit during transmission keeps a channel filled. This is done by setting a data-wire of the channel to logic "1" (which ensures the wire to be under test) and other data-wires in the channel to logic "0". If the data-wire which transmits "1" is shorted with another wire, a receiver over the channel receives the bit

on both wires. It shows that the "1" is duplicated. Thus, packet duplication or overloading is detected. Since packet payload is affected, the TRA results in payload error. For example, consider Figure 6.1a. When the I-TPG of S_i transmits "1" on the wire l_5, the bit is received by the TRA of S_j over l_6, l_7, l_8 including l_5. The TRA thus detects shorts on these wires resulting payload error.

Accomplishing testing of shorts on data-wires does not detect all shorts on the channel. The shorts on control- and handshake-wires must be handled. Tackling shorts on control-wires, the additional W1 set must be accommodated in header and trailer flits i.e., control framing bits. All packets are transmitted with header and trailer flits. Thus, shorts on control-wires may affect these flits. When receiver's TRA over a channel receives a modified header, it detects packet misrouting error. It results turning of packets to a different direction. Subsequently, packets may be routed to unintended nodes. If the TRA receives modified trailer, it detects the packet dropping error. It enhances the timeout faults occurring due to non-receipt of the end of packet (eop) signal. Similar to the data- and control-wires, the shorts on handshake-wires in channels are handled by adding extra W1 set on "val" and "ack" bits and shifting corresponding flits on these wires. Furthermore, when traffic size becomes large then transmitted packets have to wait in channels for a long time that delays a packet. During testing of data-, control-, or handshake-wires, if receiver's TRA receives same test flit as sent over a wire, then the wire is non-faulty. Thus, the state of faultiness or non-faultiness of the wire is determined.

$$\oplus \begin{cases} 00001000 \\ 00000100 \\ 00000010 \\ 00000001 \\ 00001000 \\ --\,-- \\ 00000111 \end{cases} \qquad (6.4) \qquad + \begin{cases} 00001000 \\ 00000100 \\ 00000010 \\ 00000001 \\ 00001000 \\ --\,-- \\ 00001111 \end{cases} \qquad (6.5)$$

6.3.1.2 Fault Diagnosis

Detecting short faults in channels ensures the state of wires in the channels. These faults must be diagnosed to identify the faulty wires which later are exercised by a fault-tolerant routing protocol. The FDM in TRA modules at a node is implemented to diagnose faults by analyzing incoming test flits (responses) with local test flits. The analysis carried out by an FDM of receiver core and router is the verification of a response with its corresponding W1

Table 6.3: The 2-bit TRA signals used to identify faulty channel wires due to short faults.

Bits	Signal Type	Signal Meaning	Affected Wires
00	No Fault	Data received over a channel are correct	–
01	PE	Data received at multiple wires in a channel	Data and handshake wires
10	ME	Data received at unintended node	Control wire that carries packet header
11	TE	Data get dropped	Control wire that carries packet trailer

sequence. Thus, an FDM does the pattern checking. This verification can be performed by logical "XOR" and logical "OR" operations. Former logical operation results in shorted wires excluding the wire sought under test. Later logical operation results in a set of shorted wires including the wire sought under test. On the basis of the diagnosis results, i.e., the type of faulty channel-wires that bring the underlying network into system level failures, the TSG unit of the TRA generates the 2-bit signals that correspond to a channel error. Meaning of 2-bit TRA signals is defined in Table 6.3. Generally, three types of channel errors which are considered to realize the failure modes are payload error (PE), misrouting error (ME), and dropping error (TE).

For instance, consider Figure 6.1a. The S_i transmits "1" i.e., "0000 1000" as test flit on the l_5. The bit-stream is received at S_j as "0000 1000", "0000 0100", "0000 0010", "0000 0001" on the l_5, l_6, l_7, l_8, respectively. These test responses undergo logic operations as defined in Equations 6.4, and 6.5 with the corresponding local test flit i.e., "0000 1000" already derived by I-TPG of S_j. Equation 6.4 identifies that l_6, l_7, l_8 are shorted with l_5. Thus, the transmitted test flit "0000 1000" is altered to "0000 1111". During the diagnosis of the faults, the TRA's TSG block derives 2-bit signals that report payload error if l_5 is data-wire (or handshake-wire), and misrouted (or dropping) error if l_5 is a control wire that transports packet header (or trailer) flit. The TSG block else reports the state of correctness of the wire.

6.3.2 Detection of Other Types of Channel-Faults

A channel-wire may be affected by other faults, such as packet deadlock, transient, etc. Here, it is studied whether the proposed theory can handle these popular fault models. In addition, the proposed scheme is extended to address faults that manifest transient nature.

6.3.2.1 Addressing Packet Deadlock

Beside the channel-errors that occur due to shorts, NoC channels may also endure another possible logic fault, such as packet deadlock. However, the proposed solution has ensured that no such error can occur because the proposed test algorithm transmits test packets over a single hop routing path and no data packet is allowed on the channels under test. Additionally, the size of test packets is quite small that can easily be shifted without any latency degradation.

6.3.2.2 Addressing Transient Faults

Sometimes transient faults may be experienced in channels due to some temporary conditions, such as external radiation, spurious voltage spikes, connectivity issues or service unavailability between nodes. A packet cannot be routed to its destination resulting in the need of retransmission of the packet. So, a transient fault may enhance packet loss. Therefore, this fault needs to be detected for both yield enhancement and reliability improvement. Here, the existence of a transient fault in a channel under test is determined at the receiver-TRA over the channel. In order to determine a transient fault, an additional functional circuit say, $\Re$ is added to the TRA unit. $\Re$ must be designed in the view of detecting the transient faults in the on-line mode. A response flit (signal) is given as input to a demultiplexer. The output signals from this circuit are input to both the FDM and $\Re$ blocks at the TRA for the detection of a permanent fault (e.g., short) and transient fault, respectively. The detection mechanism via logical operations shown in Equations 6.4–6.5 is used for the permanent short faults. Since a transient fault lasts for a while, an observation method [190] is used at the output signal from the $\Re$ circuit in detecting the fault. To begin with the observation method, a scheme is proposed as follows.

The scheme describes that the sender node transmits same the test pattern e.g., W1 sequence. Once a response signal (*n-bit stream*) treated as a non-zero codeword is input to the $\Re$ block, it produces new code words for predefined $2^{\kappa} - 1$ shifts. Here, κ is code-word length excluding a parity bit. Figure 6.5 represents the $\Re$ block that has fault detection capability. Every least significant bit (LSB) of the words is considered as a parity bit. The cycle is repeated after $2^{\kappa} - 1$ shifts. Any error in the $\Re$'s output signal is detected by checking the parity. The "0" and "1" in the output "F" are used to ensure the presence of a transient fault or no error, respectively. The proposed short-fault test mechanism, for instance, is illustrated with *n=16-bit* channels. Therefore, the code word length is five bits including the LSB as a parity bit. Table 6.4 provides a sample of bit-stream shifting for 15 cycles. It can be seen that the area overhead by $\Re$ block is comparatively low. The transient faults are temporary and occur infrequently, one may not necessarily repeat every response pattern to detect a transient simultaneously with manufacturing faults. In this case, a 2×1 multiplexer unit is

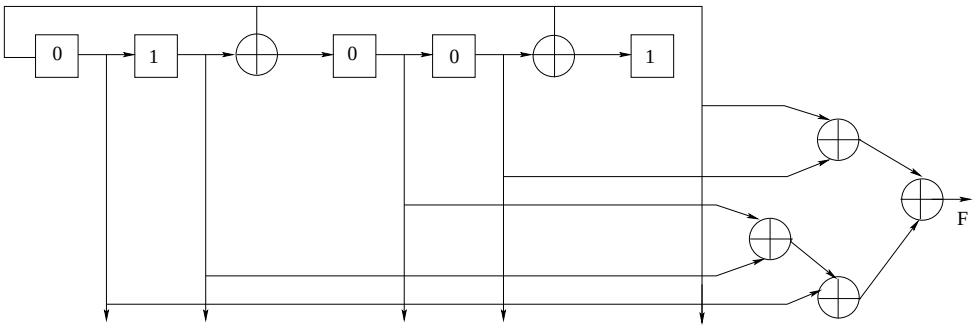


Figure 6.5: Logic design for detection of transient fault.

Table 6.4: Bit streams after different shifts.

shift	bit-stream				
0	0	0	1	0	1
1	1	0	1	1	1
–	–	–	–	–	–
14	0	1	1	1	1

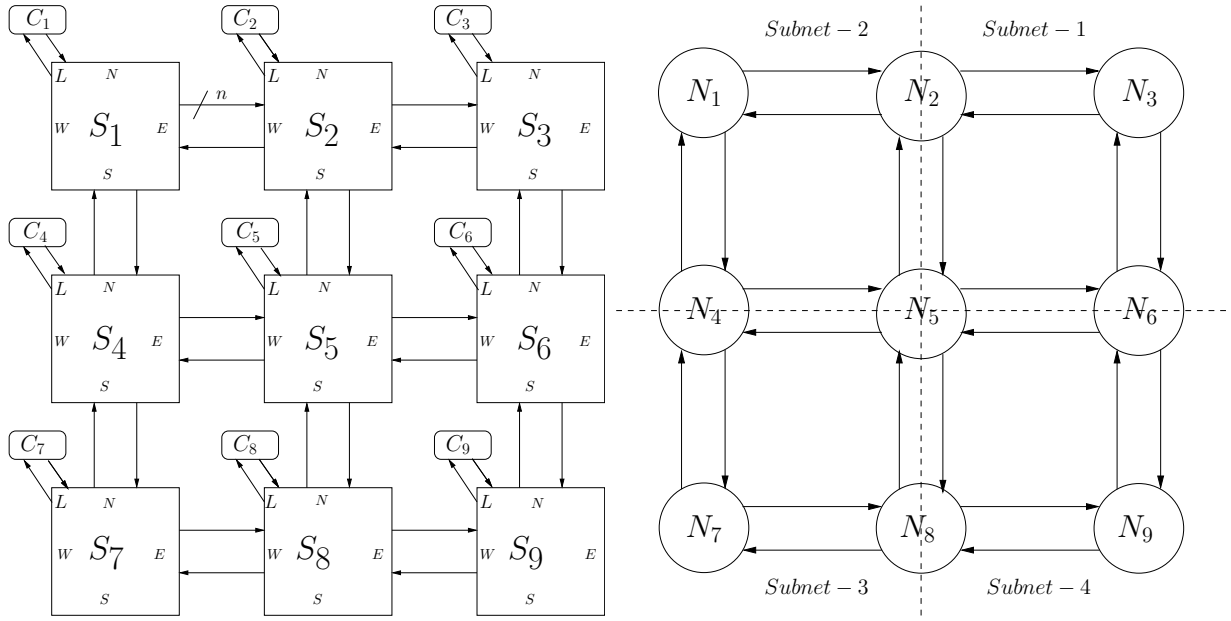
placed before the $\Re$ block. The control/selection line is activated randomly/periodically to initiate the detection of a transient fault at the $\Re$ block. Once, the transient fault is detected, the receiver node automatically sends a request message to the sender for retransmission of the packets.

6.4 Test Scheduling

One of the main criteria of the proposed solution is to provide low test time. Another criterion is similar network resource utilization, such as test energy during application of the test mechanism especially in the on-line mode, i.e., on a part of a network (subnet). Simultaneously, the proposed solution must meet adequate network performance, be scalable and adaptable to many NoC topologies. Therefore, a suitable test scheduling is needed, that fulfills the aforementioned criteria on modeling an NoC architecture.

6.4.1 Modeling an NoC Architecture

An NoC topology that organizes physical interconnection of nodes can be considered a graph $\mathbb{G} = (\aleph, \complement)$. Here, $\aleph = \{N_i; i \in \mathbb{N}\}$ represents the set of nodes, and $\complement = \{Ch_j; j \in \mathbb{N}\}$ represents the set of unidirectional interswitch and local channels. Each node $N_i =< S_i, C_i >$ is combination of a router S_i and its dedicated core C_i. Each channel is shared by a router pair or pair of a router and its adjacent core. The former channel is referred to an interswitch channel while later is referred to a local channel. A channel may be unidirectional or bidirectional. Depending on the channel category $\mathbb{G}$ is consequently treated as a directed or undirected



(a) Abstract view of a 3×3 NoC.

(b) Division of a 3×3 network.

Figure 6.6: Abstract representation and segmentation of a 3×3 network architecture into four subnets where each subnet in turn undergoes a test mode for its channels.

graph. For example, Figure 6.6a shows an abstract representation of a 3×3 mesh network with unidirectional channels. Correspondingly, the graphical view of the network is shown in Figure 6.6b.

6.4.2 Test Region Selection

As mentioned, the proposed scheme has the goal to make it test-time-independent by getting the constant number of test rounds (TRs) with respect to general NoC size and topology. Also, each test round must take same test time. Otherwise, the varying test rounds incur different test times (costs) and as a result, the overall test time becomes higher that may have the direct impact on the network performance. For example, higher is the test time taken by a test round, the latency of application packets degrades accordingly. Another goal of the proposed scheme is to provide similar resource utilization, such as test energy on the test rounds whence executed on an NoC. Therefore, a trade-off between the test time and resource utilization must be accomplished. The trade-off in this chapter is fulfilled by segmenting the NoC architecture into four parts, each is called a subnet. It can be noted that the size of each subnet must be (nearly) the same.

Let us consider $\mathbb{G}$ of a $P \times Q$ NoC (e.g., mesh and torus). This $\mathbb{G}$ is segmented into four connected components as subnets. To access similar resources in testing the channels of the subnets, each subnet has size $\lceil P/2 \rceil \times \lceil Q/2 \rceil$, $\lceil P/2 \rceil \times \lfloor Q/2 \rfloor$, $\lfloor P/2 \rfloor \times \lceil Q/2 \rceil$, or $\lfloor P/2 \rfloor \times \lfloor Q/2 \rfloor$. Each subnet is labeled following the quadrant convention in geometry of dividing the plane.

Let us take Figure 6.6b, the graphical view of a 3×3 mesh NoC architecture (Figure 6.6a). The segmentation of the network into four subnets is shown by vertical and horizontal dotted lines on Figure 6.6b. The subnets are labeled as Subnet-1, Subnet-2, Subnet-3, and Subnet-4. It shows that each subnet has the same size and looks like a 2×2 architecture. During execution of the test algorithm at a test round, only one of these subnets is in test mode and rest part is free to transmit application data. Subnet selection on a $P \times Q$ network is comparatively easy. If the NoC is not a mesh or torus, then it is modeled into $P \times Q$ matrix as proposed in Chapter 3 to find the subnets.

The decomposition of an NoC into four subnets is at least needed and explained as follows. If TR=1; the whole NoC is treated as the single subnet. Single test round is accepted on a $P \times Q$ NoC like mesh but, the proposed test approach, in this case, remains no longer in an on-line mode. Again, other NoCs like octagon cannot be tested in just TR=1. If TR=2, i.e., two subnets, the partitioning may be possible again on the $P \times Q$ NoCs but not on hybrid and octagon NoCs. With TR=2 on an octagon NoC, only two nodes can send application data where six nodes are busy in test mode. The test application is nearly an off-line mode. This issue has been discussed in Section 6.7 where the adaptability of the proposed solution on the octagon network is demonstrated. If the channels of a network are scheduled to be tested in just TR=3 i.e., on dividing the network into three subnets, it may not be easy even for $P \times Q$ NoCs to get equal sized subnets. Consequently, performance behavior and overhead, and resource utilization will become dependent on a subnet. With TR=4 i.e., four subnets, the above issues are successfully addressed. At $TR \geq 5$, increased test cost and performance degradation as the consequences are observed.

Application of the test mechanism on each subnet is treated as a test round that is completed in two test iterations. In the first iteration, the test algorithm addresses short faults[1] in the channels from the odd nodes of a subnet. The faults in rest of the channels are addressed on the execution of the test algorithm at the even nodes of the subnet. For instance, Subnet-1 in Figure 6.6b consists of four nodes N_2, N_3, N_5, N_6 where N_2, N_6 are treated as odd nodes and N_3, N_5 are treated as even nodes. The 2-color problem on a graph [170] can be employed to find a node as an odd/even. Further, one may employ the method as discussed in Chapter 3 for finding the odd/even nodes on the graphical view of an NoC architecture.

6.4.3 Test Time Evaluation

A system-wide implementation of the proposed test mechanism should conform to lower test time requirements. An NoC architecture provides an efficient realization by exploiting its highly parallel and distributed nature. In fact, NoCs with a high degree of parallelism allows testing multiple channels simultaneously. Proposed test algorithm provides high fault

[1]Detection of transient faults in a test iteration is attempted followed by the short fault detection.

coverage metrics and lower test time. The first step needs a selection of the test sequences for the detection of channel short and other faults. The algorithm exhaustively tests shorts in channels thus the W1 sequence is the preference. The second step directs the way on the application of the test sets. This functionality is accomplished by simultaneous unicasting and multicasting of test packets from core and router of a node, respectively. The advantage of transmitting test packets in this manner is that the channels from a node under test spend equal test time (clock), which is a key requirement for any test solution. The test time T_{node} at a node for its channels is the sum of the times required to derive the test set including header and trailer T_{tpg}, organize the the test set into packets T_{tpo}, transmit the test packets T_{tpt}, and analyze the test responses T_{tra}. Equation 6.6 defines T_{node} as.

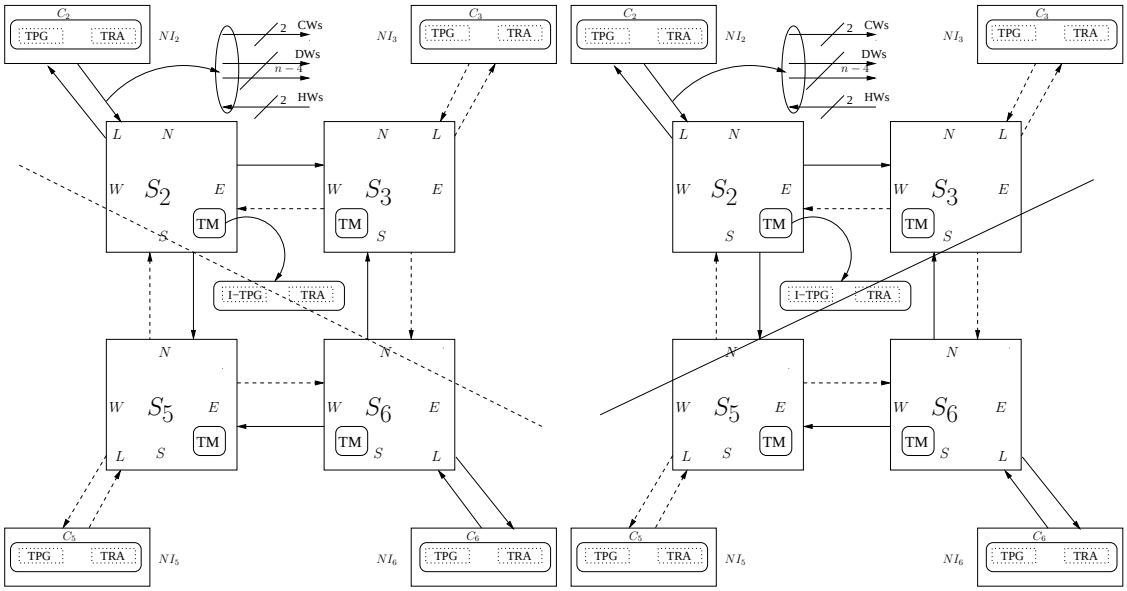
$$T_{node} = T_{tpg} + T_{tpo} + T_{tpt} + T_{tra} \tag{6.6}$$

$$T_{subnet} = 2T_{node} \tag{6.7}$$

$$T_{n/w} = 4T_{subnet} \tag{6.8}$$

Proposed test algorithm is applied at a node for its channels. Therefore, a set of nodes must be kept busy in testing their channels so that the objective of lowering total test clocks is fulfilled. The proposed test solution is applied in the on-line mode in an NoC where a part is in test mode, while the rest part can transmit application data. Accordingly, the NoC is partitioned into four parts as subnets. Partitioning of an NoC is discussed in Figure 6.6b. At a time only one subnet is in test mode and is treated as a test round. Testing of channels in a round (i.e., subnet) is completed by two iterations. In the first iteration, odd nodes send test packets which are analyzed at even nodes. The operations are reversed in the second iteration. Thus, the T_{node} is same for an iteration as well. The time T_{subnet} required to test channels of a subnet is defined in Equation 6.7. After testing the channels of a subnet, the test application is shifted to the next subnets. Then, $T_{n/w}$ that shows the time needed for testing the channels of an NoC architecture can naturally be defined as in Equation 6.8.

The implementation and analysis of the proposed test method starts with a 3×3 mesh NoC that consists of 9 routers, 9 cores, and 42 (24 interswitch and 18 local) unidirectional channels; each has $n = 16$-bit width. The network is partitioned into four subnets- 1, 2, 3, and 4 as shown in Figure 6.6. Each subnet has the same size and looks like a 2×2 mesh NoC. The proposed scheme is applied on a network in on-line mode only i.e, one subnet at a time is kept busy with testing its channels. Other subnets, on the other hand, can communicate with each other using application packets i.e. they are in functional mode. Any data packet that enters into the subnet in test mode has to wait at a corner/border router of the subnet. A subnet in test mode is termed as a test round which is completed in two iterations. The channels from all odd nodes in the first iteration (I_1) while channels from all even nodes



(a) Test Round I (TR=1), Test Iteration I (I_1)

(b) Test Round I (TR=1), Test Iteration II (I_2)

Figure 6.7: Application of the proposed test scheme in first subnet of the 3×3 network. (a) Execution of the test algorithm at odd nodes and analysis of test responses at even nodes. (b) Execution of the test algorithm at even nodes and analysis of test responses at odd nodes.

in next iteration (I_2) are undertaken for testing. Figure 6.7 illustrates the application of the test solution on the first subnet S_1 (Subnet-1) and is treated as the first round (R_1 i.e., TR=1) test application. Figure 6.7a illustrates I_1 test application in R_1. The odd nodes $N_2 =< S_2, C_2 >, N_6 =< S_6, C_6 >$ execute the test algorithm to detect shorts followed by other faults (e.g., transient faults) related to the channels (denoted by solid arrows) from these nodes. On completion of I_1, the test iteration is shifted to I_2 of R_1. Figure 6.7b illustrates this. Even nodes $N_3 =< S_3, C_5 >, N_5 =< S_5, C_5 >$ like odd nodes execute the test algorithm to address shorts in rest of the channels (denoted by dotted arrows) in the subnet. Other test rounds R_2, R_3, R_4 can be iterated by sliding the test test application in R_1 just anticlockwise for detection of short and other faults in channels of the subnets S_2, S_3, S_4, respectively.

6.4.4 Test Energy Model

The test method is driven by the TM units placed in the nodes. It is desirable for a test technique that it should be energy efficient along with the low test time as well as the small area overhead of the TMs. The proposed test scheduling has segmented an NoC architecture into four equal subnets as the test regions. The goal is to satisfy the trade-off between the test time and resource utilization, such as energy dissipation during testing of the channel-shorts in a subnet. The proposed test energy model is now described like [52] so that the proposed test solution is able to quantify the test energy metric.

In the basic working principle of the proposed mechanism, the source TMs transmit the test packets and the receiver TMs analyze the responses over the channels under test. When the test flits (W1 sequences) travel on the channels, both the channel-wires and the logic gates in the TM units result in energy dissipation. Here the dynamic energy dissipation as the test energy is considered. It is noted that this test energy is consumed by the communication process established by the TMs in the subnet whose channels are under test. The amount of test energy E depends on the number of hops, that a test flit from a source node needs to traverse to reach the destination node. As described, the test flits in the proposed method traverse only one hop. The E is defined in Equation 6.9. E_{tm} and E_{ch} are energy dissipated per flit by the TMs and by a channel under test, respectively.

$$E = 2E_{tm} + E_{ch} \tag{6.9}$$

Frequently, E_{tm} depends on the total capacitances and signal activity of the TM unit. Similarly, the factors influence the E_{ch} for the wires of the channel. E_{tm}, E_{ch} are determined in Equations 6.10 and 6.11, respectively.

$$E_{tm} = \alpha_{tm} \times C_{tm} \times V^2 \tag{6.10}$$

$$E_{ch} = \alpha_{ch} \times C_{ch} \times V^2 \tag{6.11}$$

Here, α_{tm}, α_{ch} are the signal activities of a TM and a channel, respectively. C_{tm}, C_{ch} represent the total capacitances of the TM and the channel,respectively. If a test packet consists of υ flits then the energy dissipated in transporting the packet over a channel can be calculated as defined in Equation 6.12.

$$E_{pkt} = \upsilon \times E \tag{6.12}$$

The test sequences are organized into several small packets. Let $\wp$ be the total number of test packets transported on a channel. Then, Equation 6.12 can be treated as the energy dissipated by the $i^{th}(1 \leq i \leq \wp)$ test packet. Thus, the average energy per test packet $\overline{E_{pkt}}$ is calculated as given in Equation 6.13.

$$\overline{E_{pkt}} = \frac{\sum_{i=1}^{\wp} E_{pkt}}{\wp} = \frac{\sum\limits_{i=1}^{\wp} \upsilon_i \times E}{\wp} \tag{6.13}$$

6.5 Simulation Results

The network architectures, in general, furnish multiple known advantages like concurrent data transmissions, regularity, and controlled electrical parameters [191]. Among many NoC architectures, the mesh is the most widely preferred architecture. The proposed test model

is evaluated with a series of $P \times Q$ mesh-based networks. A subset of mesh networks that includes $2 \times 2 - 8 \times 8$ mesh NoC is selected here for testing of their channels. These networks are characterized by Table 6.5. The characteristics are defined in terms of the number of routers $|R|$, cores $|C|$, channels $|\mathbb{C}|$, channel-wires #W, interswitch channels #R-R, and local channels #R-C. Every router is linked to one core and every channel has the width, for instance, $n = 16$-bit. In Section 6.6, the networks of 32-bit channels are considered to evaluate the scaling property of the proposed solution. Further, an octagon network of 16-bit channels is considered in Section 6.7 to illustrate the solution-adaptability.

Table 6.5: Characteristics of $P \times Q$ NoCs.

N/w Size	$\|R\|$	$\|C\|$	$\|\mathbb{C}\|$	#W	#R-R	#R-C
2×2	4	4	16	256	8	8
3×3	9	9	42	672	24	18
4×4	16	16	80	1280	48	32
5×5	25	25	130	2080	80	50
6×6	36	36	192	3072	120	72
7×7	49	49	266	4256	168	98
8×8	64	64	352	5632	224	128

Table 6.6: Basic simulation parameters

Parameters	Value
Channel-width	16
Switching	Wormhole
Routing	XY
Traffic	Random
Application Data	W1
PIR	0.01 – 0.1
Simulation Cycles	10000

6.5.1 Simulation Setup

For testing of the channels in networks, the simulation setup built in Chapter 5 is extended here. The Xilinx 10.1 ISE Design Suite. The Spartan3E FPGA family is used with the XC3S250E device and CP132 library package with a fault injection mechanism in channels. The channels in an iteration are placed under test. The respective TPGs send the test packets and TRAs analyze the responses. The XST [VHDL/Verilog] as synthesis tool is used to analyze hardware area overhead of these test modules. Further, the fault simulation is extended with the Modelsim PE 10.3c simulator integrated with the Xilinx ISE Design Suite to measure test (link) and fault coverage metrics. The interconnect shorts put a network into many system-level failures. To see the effect of the faults on network performance, the system simulation is performed using the Noxim simulator [22]. The simulator is well known to be a cycle accurate simulator. Table 6.6 gives basic simulation parameters configured in the simulation. The wormhole switching is used to packetize the W1 sequences as the application as well as test packets. Each packet has four flits including header and trailer flits. Test packets are transmitted by TPGs and analyzed by TRAs at nodes of the subnet in test mode while the application packets are transmitted using XY routing. The packet injection rate (PIR) i.e., application data packets are injected in the range of 0.01–0.1 in a test round in the subnets not in test mode. Each time, the simulation is continued for 10000 cycles out of

Table 6.7: Synthesis Results for the TM on a node with 16-bit channels.

n	TM Unit	#LGs	RASoC(%)	Xpipe(%)	Core(%)
	C-TPG	118	6.99	7.74	2.5
	C-TRA	84	4.98	5.51	1.5
	C-TM	202	11.97	13.25	4
16	R-TPG	152	9	9.97	–
	R-TRA	114	6.75	7.48	–
	R-TM	266	15.75	17.45	–

which 1000 cycles are used as the wake-up period needed to initiate the simulation and collect its statistics.

6.5.2 Hardware Area Overhead

Each proposed TM that consists of a pair of TPG and TRA units are the driving components of the test mechanism. As mentioned, there is only one TM component for a whole router. Also, another TM component is needed for the dedicated core of the router. The TMs are implemented using Verilog of the Xilinx ver. 10.1 ISE Design Suite together with a short-fault injection mechanism in channels. Two well-known router RASoC [65], and Xpipe [66] are considered to see area taken by the proposed TMs in these routers. Table 6.7 provides the synthesis results towards area overhead in terms of logic gates (LGs) of the TMs at a core and router of a node. A router has five input/output (I/O) ports that share 16-bit channels. As can be seen, a core TPG (C-TPG) takes 6.99–7.74% router area while a core TRA (C-TRA) takes 4.98–5.55% router area. In other words, a core TM (C-TM) takes nearly 4% area in the core. Similarly, it is seen that a router TPG (R-TPG) and router TRA (R-TRA) individually occupies 15.75–17.45% router area. As only channel-shorts are targeted, the area overhead is quite small and acceptable. The area overhead is reduced on the border/corner routers because the number of I/O ports at these routers are less by one (for border routers) and two (for corner routers) compared to routers with five I/O ports. In other words, a border and corner router respectively have three and two neighbor routers. It is seen that area of TMs in these routers are 2–4% less as compared to the router with five I/O ports. Therefore, size of a TM depends on the ports of a router and the channel-width. Further, an area of a TPG at a core is smaller than that at a router in general because of the fact that the TPG at the core generates the test sequences and verifies the responses only for its fixed local channel. Whereas, the TM component in a router does the job for its fixed local as well as varying interswitch channels. Thus, extra circuits in the logic are needed. Furthermore, an IP core is much larger (approximately five times) than a router size. Consequently, the area occupied by a TM at a core is much smaller than that at a router of a node.

6.5.3 Solution Evaluation

Evaluation of the proposed test solution is done to measure test clocks and coverage metrics, and to observe the effect on network performance for the channel shorts. Here, the proposed solution is evaluated at $n = 16$. So, every channel can be treated as a 16-bit bus. Figure 6.8 gives the size of intra-shorts #S1, and inter-shorts #S2 injected in 16-bit channels of the networks.

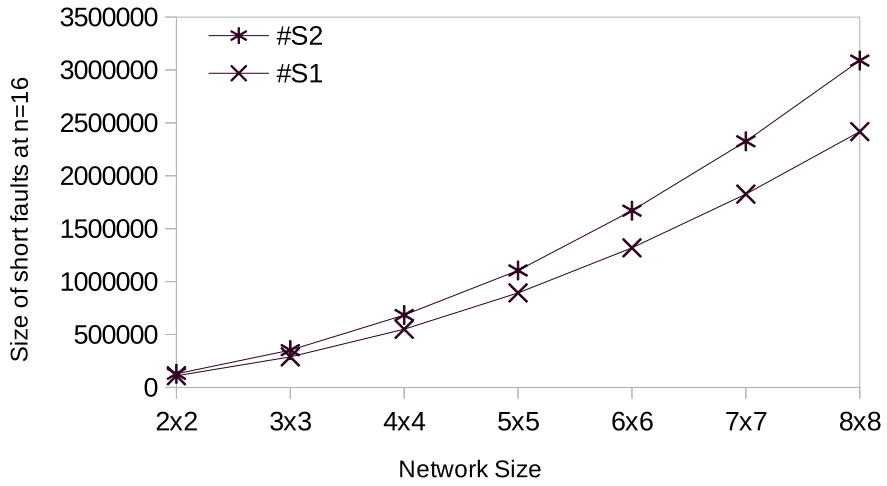


Figure 6.8: Size of intra- and inter-shorts injected in channels of the networks for $n = 16$.

6.5.3.1 Test Clock Analysis

The shorts injected in NoC channels are considered for testing in four rounds. Each round is completed in two iterations. For example, consider Figure 6.6 and its first-round test application for channel-shorts as shown in Figure 6.7. The set of channels in an iteration and rest of the channels in another iteration of the test round are activated consequently. Respective TPGs are placed at one side to shift the test packets and TRAs are placed at another side of the channels for the analysis of test responses. During the experiment, it is seen that a TPG at core or router needs 5 clocks (i.e., $T_{tpg} = 5$) to derive the required W1 sequences as the test set including header and trailer information. One clock i.e., $T_{tpo} = 1$ is needed to organize these raw test set into packets. Each test packet has 4 flits including header and trailer flits. So, the W1 test set is organized into 8 packets. Every test flit from the test packet is transmitted at a clock. Then, T_{tpt} equals to 32 clocks. On receiving first test packet, the TRA at a receiver starts diagnosing the received test flits. The operation is continued along with receiving test flits from a neighbor' TPG. Thus, additional 2 clocks after the last received test packet are needed by the TRA to complete its current test iteration.

Therefore, the test time T_{node} of an iteration equals 40 clocks. Since each round goes through two iterations and each iteration takes the same amount of test clocks, testing of the channels of a subnet is completed in just 80 clocks. The similar test configuration is iterated for other subnets of the underlying network. Thus, the $T_{n/w}$ equals 320 clocks (Table 6.8) since the proposed model assumes a network of four subnets. The test clocks for detecting channel-shorts are same irrespective of the network size and type till the width n of a channel remains alike. It may be noted that this is the biggest achievement of the proposed solution.

6.5.3.2 Coverage Metrics Analysis

Considering a subset of channels in a test round partially covers both link (test) and fault coverage metrics. In I_1 (Figure 6.7), 4 interswitch and 4 local i.e., 8 channels from nodes N_2, N_6 are tested. Rest 8 channels i.e., 4 interswitch and 4 local channels from nodes N_3, N_5 are tested in I_2 (Figure 6.7). Therefore, link coverage metric (LCM) in each iteration is 8/16*100=50%. Hence, cumulative LCM in this round is 100%. On shifting the test round to another subnet subsequently, all channels of the network are covered. Thus, the LCM achieved for the network is 100%.

Table 6.8: Fault coverage analysis on $P \times Q$ NoCs at $n = 16$.

N/w Size	$T_{n/w}$ (Clocks)	PE (%)	TE (%)	ME (%)	Total FCM(%)
2x2	80	71.63	14.31	14.06	100
3x3	320	74.54	17.75	7.71	100
4x4	320	69.30	14.34	16.36	100
5x5	320	76.53	14.71	8.76	100
6x6	320	71.98	14.67	13.35	100
7x7	320	75.46	16.10	8.44	100
8x8	320	77.50	16.71	5.79	100

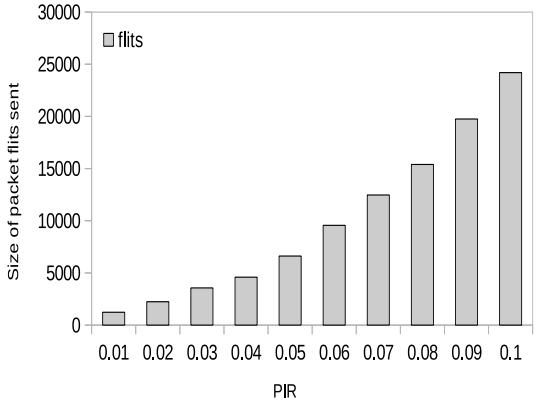
Like the LCM, the fault coverage metric (FCM) is another important metric achieved during fault simulation. In the first round, 16 channels are put in the test mode and the subnet looks like a 2×2 network. Sufficient W1 sequences are shifted by TPGs and respective test responses are analyzed by TRAs to detect shorts on the channels. The shorts are detected in the forms of their occurrences on data, control, and handshake wires of the channels. First, 25168 intra-shorts and 10224 inter-shorts i.e., 35392 shorts are exhaustively tested on data-wires (DWs) on shifting respective test packets. Then, the fault simulation achieves the FCM (35392/128128)*100%=27.62%. Second, the fault simulation is performed on control-wires that include "bop" and "eop" wires. When considering these wires, the experiment is expanded to data-wires resulting in the detection of 55328 intra-shorts and 13944 inter-shorts. Thus, the FCM is 54.06%. Finally, the fault simulation is performed on handshake-wires that include

"ack" and "val" wires. The experiment is extended to data- and control-wires resulting the detection of 549440 intra-shorts and 134144 inter-shorts i.e., 683584 shorts. Thus, the FCM is 100%. Shifting the current test round to another subnet, same fault coverage metric can be achieved. Thus, the fault simulation campaign in the network detects all modeled faults resulting the FCM to be 100%. The shorts detected over channels by TRAs are manifested as packet payload, misrouting, and timeout errors. Table 6.8 provides the fault coverage analysis of current fault simulation campaign on the 3×3 network.

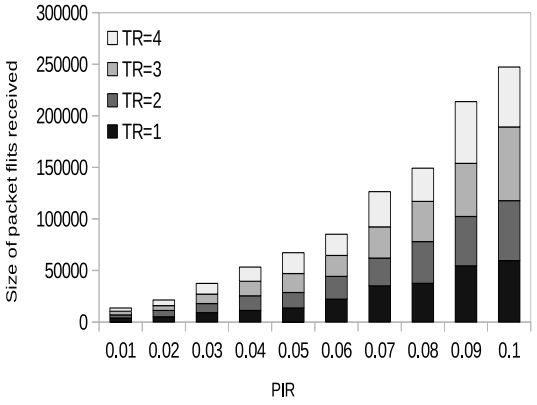
6.5.3.3 Network Performance Analysis

So far the evaluation, the proposed model finds the test time and coverage (test and fault) metrics on a 3×3 network. Now deeply study the model when application data are transmitted in the network where a part is under test. The channel-shorts put the network into following system-level failures- overloading, misrouting, delaying, and dropping of packets. Consequently, the performance of the network is significantly affected. The effect of shorts on network performance metrics is focused on the following performance metrics- throughput, latency, and power (energy) consumption. Rigorous simulations are performed on an NoC to measure the performance metrics. Figure 6.9 reveals the performance of a 3×3 network at $n = 16$ as an on-line evaluation of the proposed solution. The amount of traffic in terms of packet flits injected in the network is given in Figure 6.9a. When traffic traverses a faulty channel then many packet flits get duplicated that increase the amount of received packet flits at a node over the channel. The duplicate packet flits may further be increased if multiple faulty channels are encountered in a routing path. Therefore, destination node receives many duplicate flits. Again some packets may be misrouted to an unintended receiver due to a short on a control wire that has modified the packet header. Figures 6.9b and 6.9c demonstrate the amount of packet flits received and dropped in the network. It is seen that ≈2.05-3.17X i.e., 2.61X of the injected flits on average are received after a test round in the network. Such large received flits are naturally summed the original, overloaded (duplicated), and misrouted flits. If a node does not receive the "eop" signal i.e., packet trailer within a scheduled time interval, the packet is assumed to be lost due to "timeout" error. In the normal mode of a network, the timeout error is ≈3-7% of the injected traffic. This error is enhanced due to shorts on the control wire that carries packet trailer.

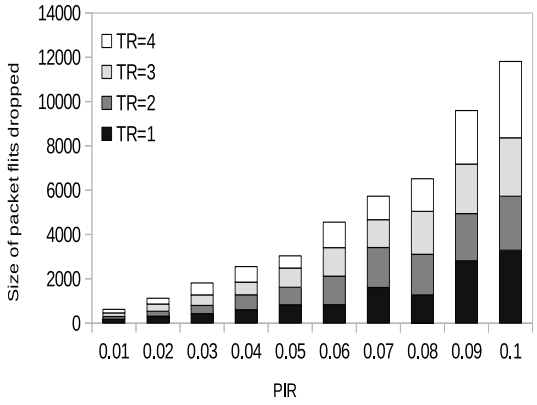
When channel shorts are considered in the network then packets get delayed due to overloaded traffic resulting enhanced packet timeout. As result, more flits are dropped. Many duplicate flits forwarded by a router in the underlying routing path may be dropped later in the path. Thus, the dropping as observed raises to ≈8.31-14.47% of the injected traffic flits. With such amount of packet flits received and dropped, the behavior of performance metrics-throughput at a node, average latency of a packet, and energy consumed by a flit at different test rounds are observed and demonstrated in Figures 6.9d, 6.9e, and 6.9f, respectively.



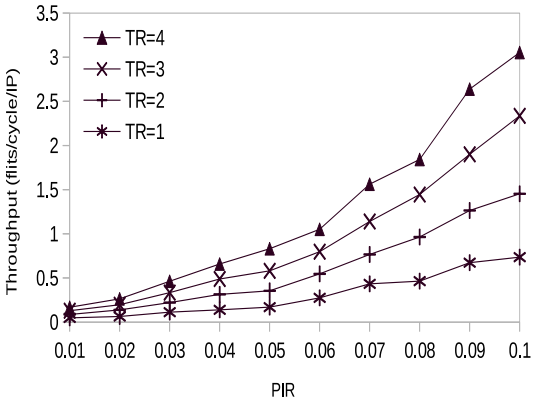
(a) Amount of packet flits injected in the 3×3 NoC.



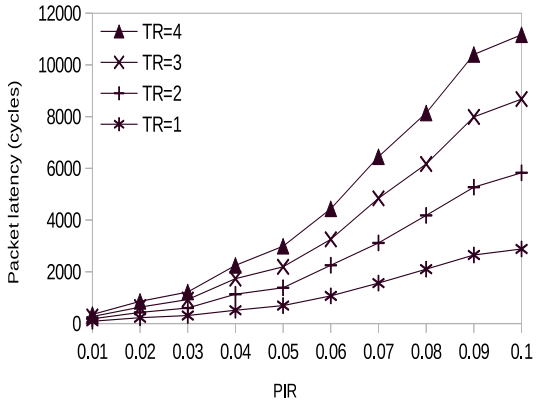
(b) Amount of packet flits received in the 3×3 NoC.



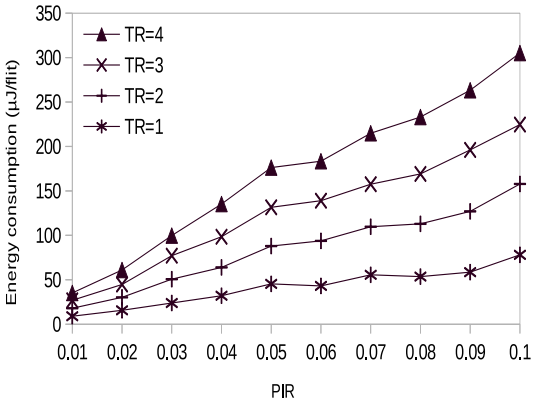
(c) Amount of packet flits dropped in the 3×3 NoC.



(d) Throughput behavior observed in the 3×3 NoC.



(e) Avg. Latency behavior of a packet in the 3×3 NoC.



(f) Energy consumed by a flit in the 3×3 NoC.

Figure 6.9: On-line evaluation of the proposed solution to observe the behavior of various performance metrics in the 3×3 NoC with 16-bit channels.

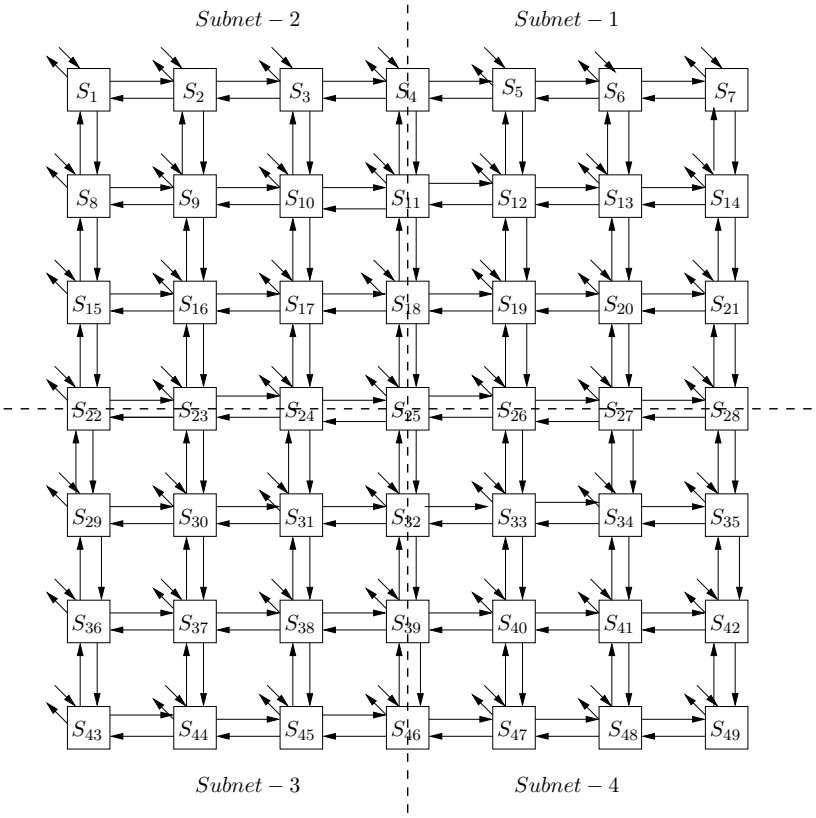


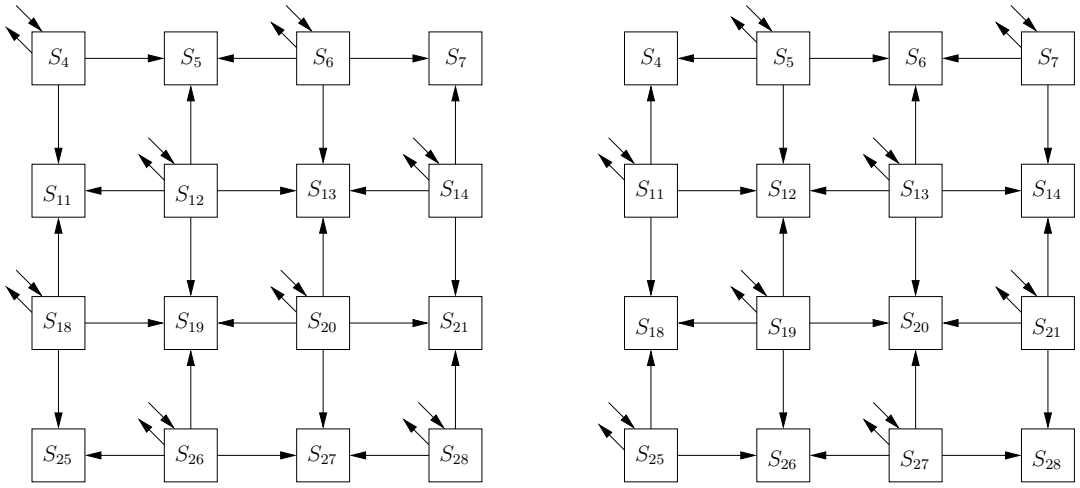
Figure 6.10: The partitioning of a 7×7 NoC to four subnets and respective test rounds for an application of the proposed test solution. The solution duly illustrates the scalable behavior with a larger network.

6.6 Solution Scalability

The proposed test solution scales to NoCs independent of its size and type. Taking the advantage of the proposed test scheduling scheme already discussed in Section 6.4, the scalable behavior of the proposed test solution is described irrespective of the size and channel width of all the NoCs included in Table 6.5.

6.6.1 With NoC Size

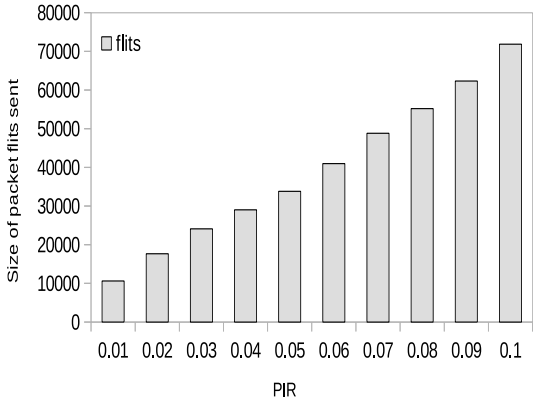
If the faults are to be detected in larger NoCs the test method must be executed concurrently in such way that the test time overhead is reduced. It will help the incoming application packets to wait for less time when they have entered into a node involved in testing the channels. The proposed test scheduling scheme segments an NoC architecture into four test regions independent of the size of the NoC. The regions (subnets) are named as per the quadrant in the XY plane. Subsequently, the proposed solution model is named as a quadrant model (Q-Model). The natural scaling of the Q-Model here is illustrated with a 7×7 mesh NoC. The partitioning of the network into four subnets is shown in Figure 6.10. Characteristics of the network are provided in Table 6.5. As can be seen that the network has 168 interswitch



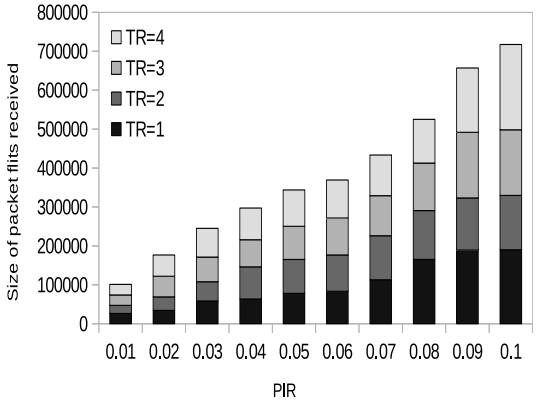
(a) Round I (TR=1), Iteration I (I_1). (b) Round II (TR=2), Iteration II (I_2).

Figure 6.11: Application of the test algorithm in Subnet-1 of the 7×7 NoC i.e., first test round. (a) and (b) demonstrate the execution of the test algorithm at odd and even nodes, and subsequently analysis of test responses at even and odd nodes in first and second iterations, respectively.

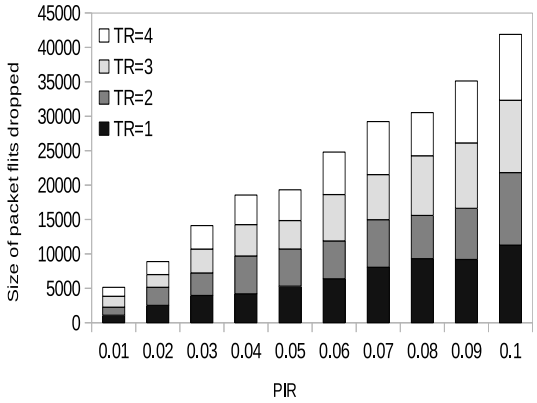
and 98 local i.e., 266 channels; each has width $n = 16$-bit. Since the bit-width of the channels remains fixed, area overhead of the TMs in a node is same as earlier (Table 6.7). Thus, the area overhead of the TMs in nodes remain same for the node with 16-bit channels. The channels of the 7×7 network are tested by applying the test algorithm on the nodes of a subnet treated as a test round and shifting of the test application to next subnets. Figure 6.11 for instance, demonstrates the first round test application that tests 80 channels in the subnet. The first iteration of the first round is shown in Figure 6.11a while Figure 6.11b presents the next iteration on the same round. Note that the subnet is treated like a 4×4 network. As computed previously, $T_{subnet} = 80$ and $T_{n/w} = 320$ clocks. Thus, it states that the test clocks are same for all the networks (Table 6.8). In the round, testing of 40 channels in the first iteration results in 50% LCM and rest 40 channels in next iteration results in 50%. Thus, the round achieves 100% LCM. The test round is shifted to other subnets in the network and the application results 100% LCM for the network. The TPGs at one end of the channels transmit test packets containing required W1 sequences. The TRAs at the other end of the channels analyze test responses and detect shorts in these channels in phases. First, the experiment detects 125840 intra-shorts and 75264 inter-shorts on data-wires of the channels. It results in 29.41% FCM. Next, the experiment undertakes control-wires and extends the test region to data-wires. Note that, 276640 intra-shorts and 102592 inter-shorts are detected resulting in 55.47% FCM. Furthermore, the experiment undertakes handshake-wires and extends the test region to both data- and control-wires; 549440 intra-shorts and 134144 inter-shorts are detected. Thus, the FCM reaches to 100%. Since similar test configuration is iterated in other subnets, all modeled shorts in channels of 7×7 network are addressed. Thus, the fault



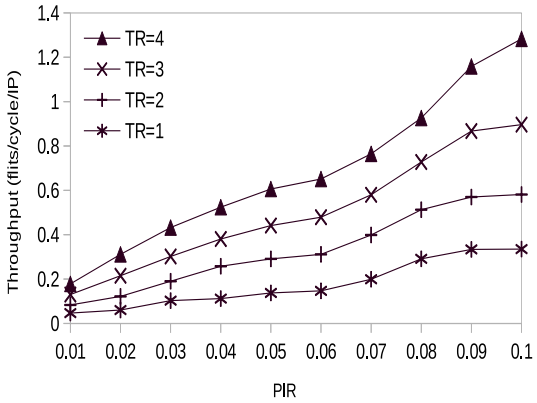
(a) Amount of packet flits injected in the 7×7 NoC.



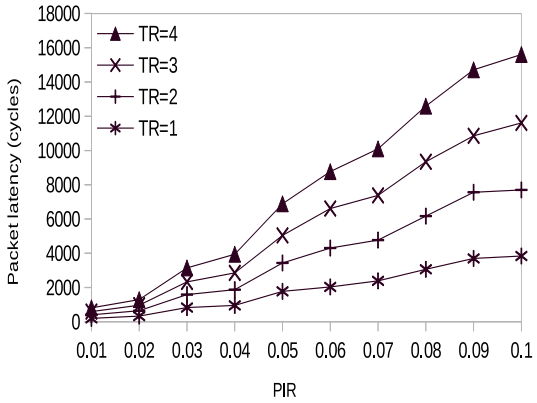
(b) Amount of packet flits received in the 7×7 NoC.



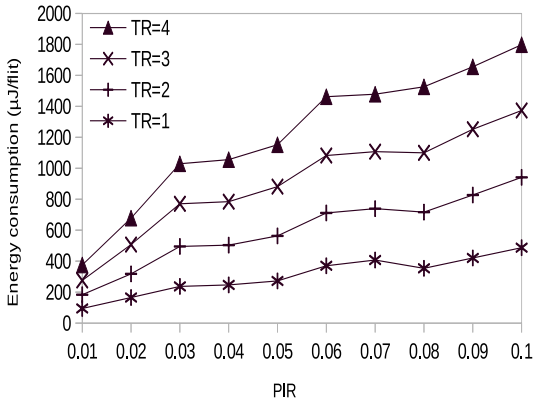
(c) Amount of packet flits dropped in the 7×7 NoC.



(d) Throughput behavior observed in the 7×7 NoC.



(e) Avg. Latency behavior of a packet in the 7×7 NoC.



(f) Energy consumed by a flit in the 7×7 NoC.

Figure 6.12: On-line evaluation of the proposed solution to observe the behavior of various performance metrics in the 7×7 NoC with 16-bit channels while the channels in a subnet are kept in test mode and rest part is in functional mode.

simulation results to 100% FCM. As the faults are detected on channel-wires, TRAs announce the type of errors which are already provided in Table 6.8.

Continuing with the fault simulation campaign, the on-line evaluation of the proposed solution is performed in the network to observe the effect of shorts on performance metrics. Figure 6.12 demonstrates performance evaluation at all four test rounds for $n = 16$. The simulation setup is extended to observe the effect of channel-shorts on the network. It is observed that the amount of received flits is 1.93–3.09X of the number of flits injected in the network. Also, it is seen that the amount of dropped flits varies in the range of 10.29–18.95% of the injected flit size.

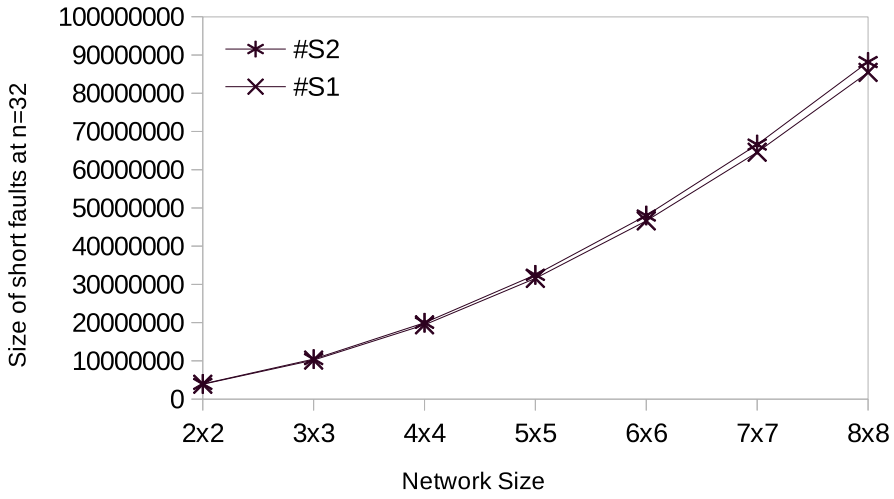


Figure 6.13: Size of intra- and inter-shorts injected in networks at $n = 32$.

6.6.2 With Channel Width

The width of a channel has the direct effect on performance metrics especially in minimization of packet latency. The channel width of a network varies with topology, applications, and so on. Moreover, the same topology may have a channel with varying width e.g., fat-tree network. Accordingly, a test solution must scale with such channel configurations. The proposed Q-model scales to networks with larger channel width. Here, the networks characterized in Table 6.5 are assumed for $n = 32$-bit instead of $n = 16$-bit. So the characteristics of the NoCs remain same except for the wire set. The new size of the channel-wires #W is mentioned in Table 6.9. On these wires both intra- and inter-shorts are modeled and simulated as shown in Figure 6.13. As the channel-width is doubled, the TMs in a node must generate the required W1 sequences and diagnose the channel-shorts by analyzing the responses. Some routers like CHAIN, MANGO, SPIN [10] have more size ($\leq 1.5\times$) than RASoC and Xpipe routers with

Table 6.9: Fault coverage analysis on $P \times Q$ NoCs at $n = 32$.

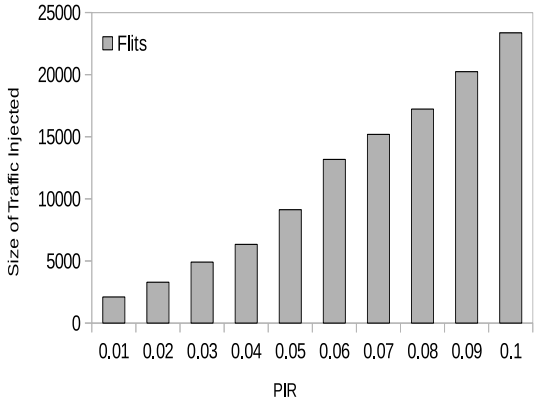
N/w Size	#W	T_{NoC} (clocks)	PE (%)	TE (%)	ME (%)	Total FCM(%)
2x2	512	156	70.44	16.75	12.81	100
3x3	1344	624	61.21	15.24	23.55	100
4x4	2560	624	63.89	18.01	18.10	100
5x5	4160	624	68.84	17.35	13.81	100
6x6	6144	624	69.70	14.85	15.45	100
7x7	8512	624	74.52	15.37	10.11	100
8x8	11264	624	61.31	16.36	22.33	100

Table 6.10: Synthesis Results for the TM in a node with 32-bit channels.

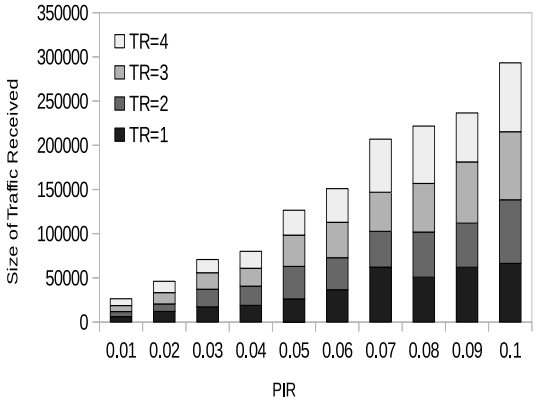
n	TM Unit	#LGs	RASoC(%)	Xpipe(%)	Core(%)
	C-TPG	146	5.97	6.6	1.83
	C-TRA	105	4.29	4.75	1.21
	C-TM	251	10.26	11.35	3.04
32	R-TPG	219	8.95	9.9	–
	R-TRA	135	5.52	6.11	–
	R-TM	354	14.47	16.01	–
		251-354	19-22	22-25	5

32-bit channels. Area overhead of TMs in these routers is less. Table 6.10 provides the area overhead in these networks for a node with 32-bit channels. As can be seen that a C-TM takes 10.26–11.35% router area (or 3.04% are in the core) while an R-TM takes 14.47–16.01% router area only. In reference to a 16-bit channel, these areas are nearly 19–22% and 22–25% of RASoC and Xpipe routers, respectively while 5% area is seen to be reserved in the core. Though the channel-width becomes double the TM area increases only by few gates. Thus, one can conclude that the area overhead incurred by a TM increases slowly with the rapid increase of the channel width.

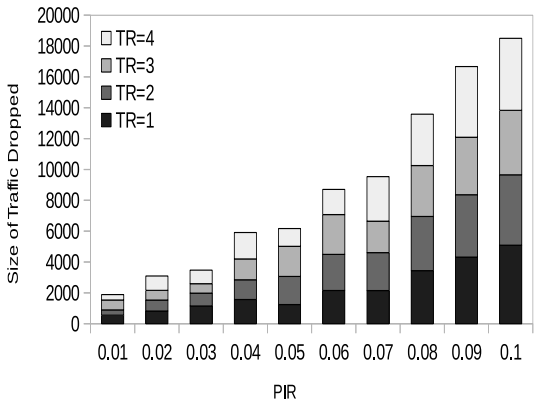
Though shorts increase exponentially with channel width the T_{node} varies almost linearly. It is because of the larger TM block in a node. For instance, consider the testing of 32-bit channels on the networks characterized in Table 6.5. In a node, a TPG generates 32 W1 sequences and organizes into 16 test packets. The TPG needs 10 clocks to generate the W1 set and 2 clocks to packetize. The test packets are shifted in 64 clocks. Since a TRA over a channel under test starts analyzing the test responses on immediate receiving a test packet, additional 2 clocks are sufficient to finish the ongoing testing. Thus, $T_{node} = 78, T_{subnet} = 156, T_{n/w} = 624$ clocks (Table 6.9). This experiment as earlier has detected all modeled shorts on the data, control, and handshake wires. The coverage metrics (both test and fault) have



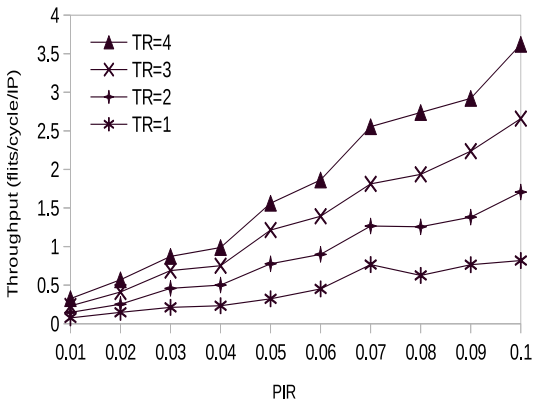
(a) Amount of packet flits injected in the 3×3 NoC.



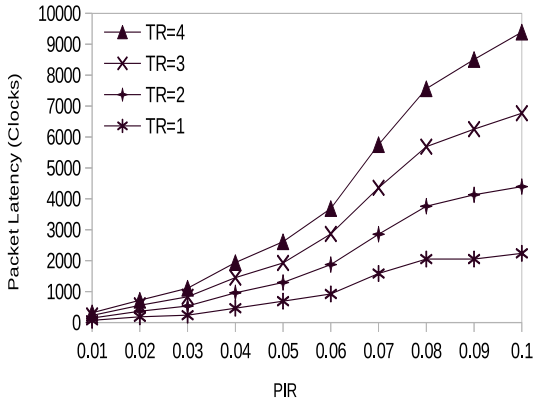
(b) Amount of packet flits received in the 3×3 NoC.



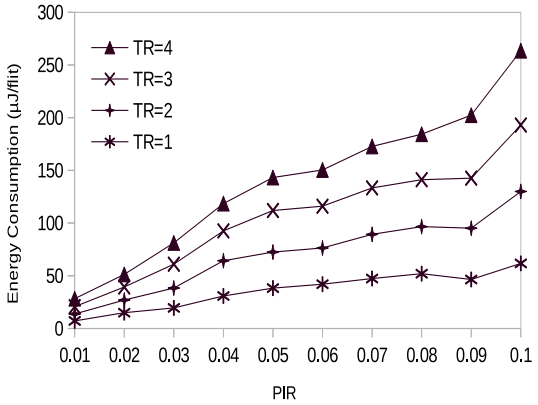
(c) Amount of packet flits dropped in the 3×3 NoC.



(d) Throughput behavior observed in the 3×3 NoC.



(e) Avg. Latency behavior of a packet in the 3×3 NoC.



(f) Energy consumed by a flit in the 3×3 NoC.

Figure 6.14: On-line evaluation of the proposed solution to observe the behavior of various performance metrics in the 3×3 NoC with the 32-bit channels.

achieved 100% as earlier. Consequently, the manifestation of the short faults in terms of channel errors is observed and provided in Table 6.9.

Now, the effect of channel-shorts on the well-known performance metrics is investigated. For instance, the 3×3 mesh network with a 32-bit channel-width is considered. The simulation setup is extended to observe the behavior of the performance metrics under the proposed solution. The W1 sequences as application data are used and packeted using the wormhole switching technique. Figure 6.14 presents the on-line evaluation of the proposed Q-model. As the size of faults in channels increases with the channel-width, more and more packets flits may be duplicated due to payload error. At the same time, many packet flits will be dropped due to the timeout error. It is seen that the variation in the size of received as well as dropped packet flits depends on the size of packet flits injected and the size of channel-shorts experienced in a routing path. It is seen that nearly 2.5–4.25X (Figure 6.14b) and 11.57–16.86% (Figure 6.14c) of the injected traffic (Figure 6.14a) are respectively, received and dropped in the network. Subsequently, the performance metrics- throughput, packet latency, and energy consumption are reported in Figures 6.14d, 6.14e, 6.14f.

6.7 Solution Adaptability

With the ability of natural scaling of the proposed test technique established in previous sections, it would be additionally advantageous for the technique whence it shows the portability on topologies other than meshes. In this section, the efficiency of the Q-Model is evaluated on the Octagon network with 16-bit unidirectional channel configurations. The abstract representation of an Octagon network is shown in Figure 6.15. The proposed test scheduling scheme segments a network into four subnets which undergo testing of channel-shorts in turn. Without loss generality, a designer can think of the test application in the Octagon network as shown in Figure 6.16 in order to the trade-off between test clocks and performance overhead.

The octagon network consists of 8 nodes and 40 channels, each has 16-bit width. Therefore, the fault injection campaign needs to inject #S1=274720, #S2=65024 i.e., #S=339744 short faults for testing. Each node in the Octagon network has neighbor nodes, which implies that every node is a border node. The area overhead of the TM block can be reconfigured and has reduced up to 2–3% on the size provided in Table 6.7 for 16-bit channels. Considering the test time evaluation method discussed earlier, each iteration needs 40 clocks to detect shorts in 16-bit channels. Therefore, $T_{n/w}$=320 clocks since testing of all channels can be completed in just four rounds or eight iterations. Let us consider the first subnet (Figure 6.16a) and put the channels under test. The fault simulation campaign is conducted in the sequence of testing DWs, CWs, and HWs in a test round. Conducting the fault simulation on the DW wire set, #S1=25168 as intra-shorts, #S2=10224 inter-shorts i.e.,

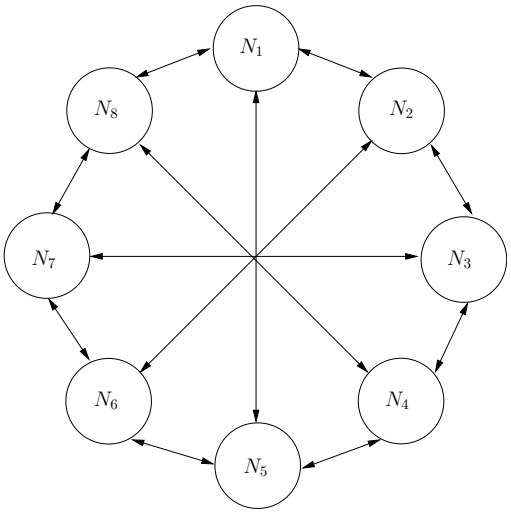
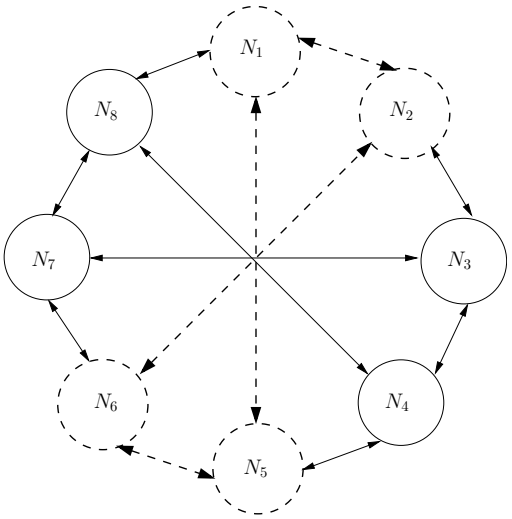
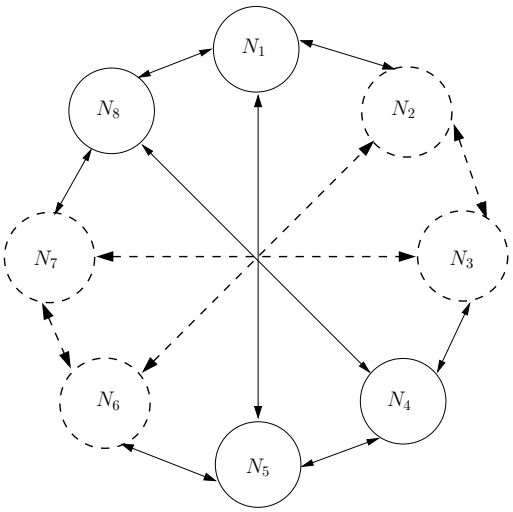


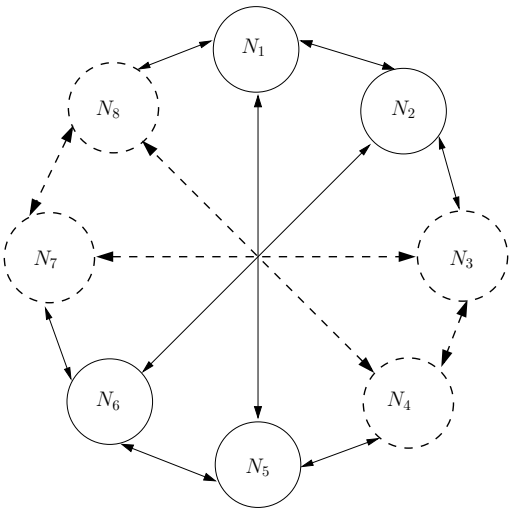
Figure 6.15: Abstract representation of an octagon network.



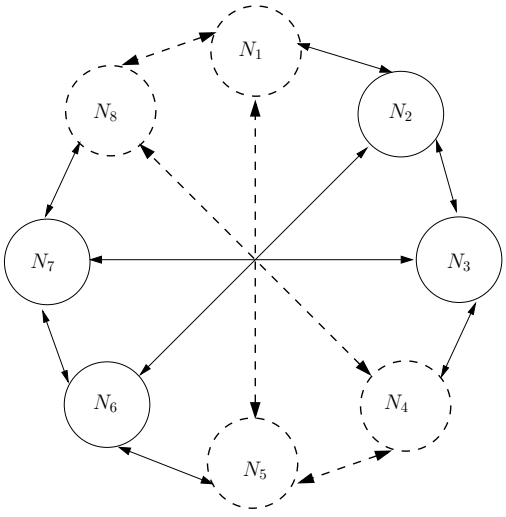
(a) First round (TR=1) test application.



(b) Second round (TR=2) test application.



(c) Third round (TR=3) test application.



(d) Fourth round (TR=4) test application.

Figure 6.16: Application of the test algorithm at various test rounds on an Octagon network.

#S=35392 shorts are detected. Subsequently, the fault simulations have detected #S1=55328, #S2=13944 i.e., #S=69272 shorts and #S1=109888, #S2=18240 i.e., #S=128128 shorts on testing the CWs and HWs. In the current test round, 16 channels are tested resulting 40% achievement on the LCM and 37.71% achievement on the FCM. The fault simulation must be repeated by shifting the test application anticlockwise as shown in Figures 6.16b to 6.16d to detect shorts in rest of the channels. All modeled short faults in channels are detected resulting 100% LCM and FCM. These shorts in channels are treated in the form of channel errors. As observed, the fault analysis reports 70.76%, 16.52%, and 12.72% as payload, misrouting, and dropping error, respectively which again cumulatively ensure 100% FCM.

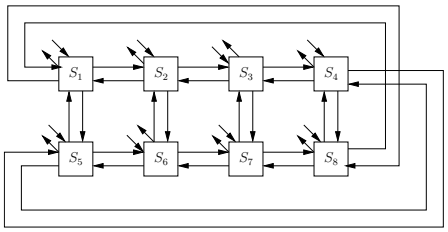
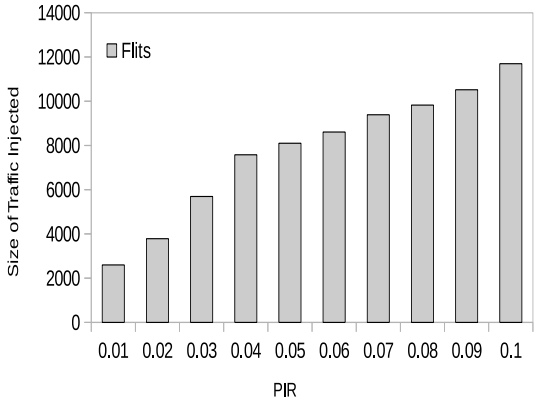


Figure 6.17: Modeling of an Octagon network (Figure 6.15) into a 2×4 network.

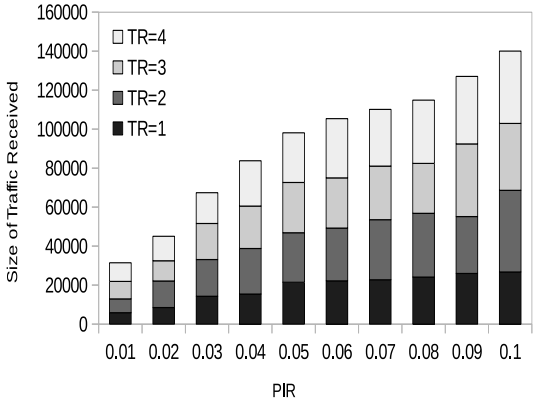
Now, three basic performance metrics- throughput, latency, and power (energy) consumption are evaluated for the proposed test model on the Octagon network. The simulation setup is extended to the Octagon network. The network is modeled to a 2×4 mesh network with the interconnection between opposite diagonal routers (Figure 6.17). Figure 6.18 demonstrates the online evaluation of the proposed test model in a test round. In other words, the significant effects of the channel shorts on the above performance metrics on the Octagon network with a 16-bit channel width are provided in Figure 6.18. The traffic in terms of packet flits is injected into the network and is shown in Figure 6.18a. Corresponding received and dropping flits observed in the network are provided in Figures 6.18b and 6.18c, respectively. Note that the received flits comprise the original and duplicated flits. A fraction of injected traffic is lost due to packet timeout fault. However, this fault is enhanced when CWs that carry packet trailers experienced shorts. Generally, 4–7% of injected traffic is lost due to timeout. But, with faulty channels, this dropping is enhanced to 10–16%. Consequently, performance metrics get changed and are provided in Figures 6.18d, 6.18e, and 6.18f.

6.8 Comparative Study

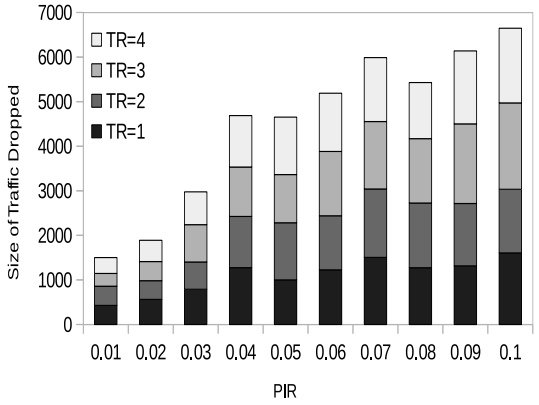
A test methodology becomes widely accepted whence it is complemented by comparison with existing mechanisms. In this work, the quality metrics used for comparison are hardware area overhead, test clocks, fault coverage and consequent channel errors, and performance overhead, such as packet latency and energy consumption by a packet flit. These quality metrics are influenced by the related parameters like test rounds/iterations, channel width,



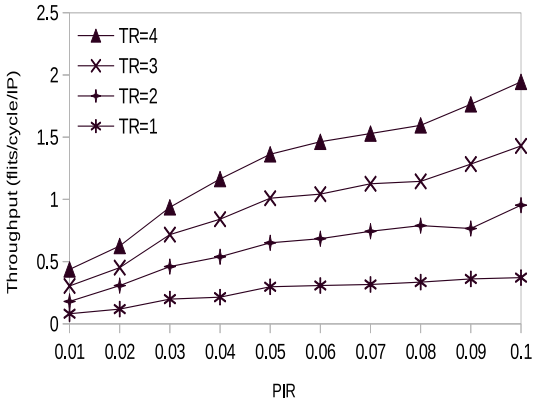
(a) Amount of packet flits sent in octagon network.



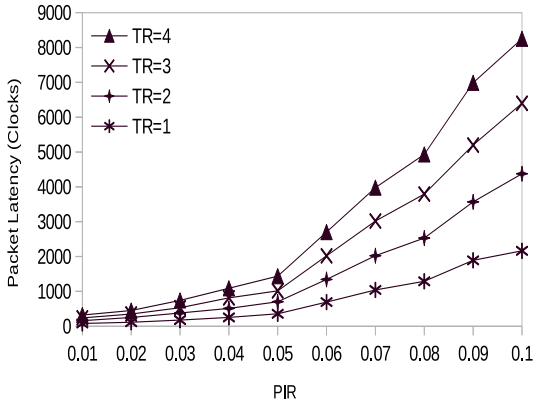
(b) Amount of packet flits received in octagon network.



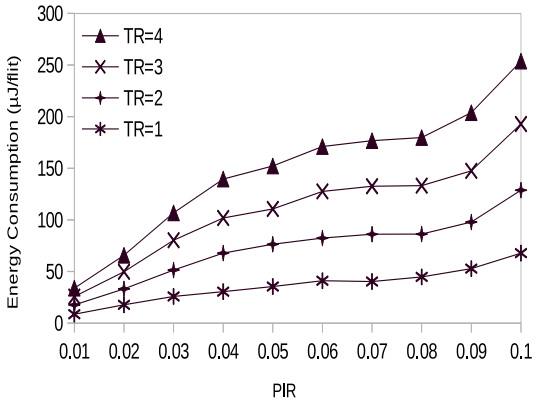
(c) Amount of packet flits dropped in octagon network.



(d) Behavior of throughput in octagon network.



(e) Behavior of packet latency in octagon network.



(f) Behavior of energy consumed by packet flit.

Figure 6.18: On-line evaluation of various performance metrics for the application of the proposed test model in an octagon network with 16-bit channels.

fault size, fault type, etc. In this chapter, one can see the benefits of the proposed Q-Model with a set of prior test models, such as the 2×2 basic mesh-based neighborhood model (2×2-Model) [37–39, 168], sequentially router selection model (S-Model) [6, 7], token-based two hop model (2hop-Model) [RPC-13], iterative 4×4-subnet selection model (I-Model) [RPC-12, RPC-14], and diagonal node activation model (D-Model) [Chapter 3] in terms of the above-mentioned quality metrics.

6.8.1 Benefits in Area Overhead

At first, the comparison analysis undergoes among the models with the area overhead of a TM architecture. In 2×2-Model, the TM architectures are placed in the IP cores of the basic 2×2 network neighborhood. The TM from a core transmits test packets to the TM in another core which is on the XY path. Each path has four hops i.e., channel length. Therefore, the number of received packets during analysis is larger while channels are affected by shorts. In order to tackle these faults, the size of a TM is naturally high. It is given that each TM takes around 40–50% of the router area depending on the 16-bit or 32-bit channels. It is naturally undesirable as the area of the TM unit depends on the channel-width. In S-Model [6, 7], the shorts are taken in 16-bit interswitch channels only. The TM takes 17% area in the router. The area overhead is increased to 21% while the shorts are extended to local channels. On considering the test mechanism to the 32-bit channels, the area overhead raises to 23% when the shorts are assumed on only the interswitch channels and 29% when the local channels, in addition, are considered. Therefore, the TM area by the S-Model ranges from 17–29%. Thus, the S-Model has saved 21–23% test area overhead resulting 42.00–57.50% improvement over the 2×2-Model. In the proposed work, the test mechanism is run at a node. From Tables 6.7 and 6.10, it is seen that a TM on average takes 12–25% (12–18% and 19–25%) area on a router for 16-bit to 32-bit channels. Thus, the proposed approach saves router area by 4–5% over the S-Model, and 25–28% over the 2×2-Model. As a result, the improvement becomes 13.79–29.41% and 50-70%, respectively as compared to S-Model and 2×2-Model.

6.8.2 Benefits in Test Clocks

Next, the comparison analysis is done for the benefits of test clocks incurred for a 16-bit channel by the test mechanisms. The test clocks directly depend on the number of test iterations (Tits). Figure 6.19 gives the *Tits* needed by a test model to account for the channel-shorts on a network. Correspondingly, the test clocks incurred by the models are provided in Figure 6.20. In order to detect the shorts in channels of a larger $P \times Q\,(P, Q > 2)$ network by the 2×2-Model, multiple 2×2 test configurations are applied in a round. Each round takes 250 clocks. It is seen that all channels can be tested in just four rounds and costs 1000 clocks irrespective of the size of the network. However, testing of the channels in

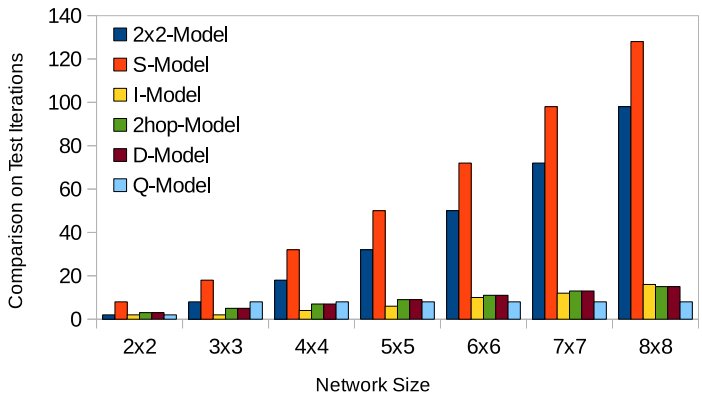


Figure 6.19: Comparison among test solutions with test iterations.

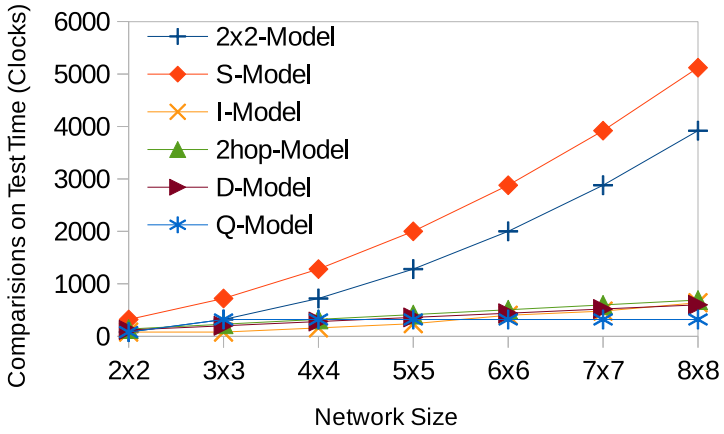


Figure 6.20: Comparison among test solutions with test clocks.

the on-line mode and putting a 2×2 subnetwork in a $P \times Q$ network as the test region in a round, the number of test iterations ($Tits$) can directly be determined from the network size as $Tits = (P-1) \times (Q-1)$. Correspondingly, the test time will be $250 \times (P-1) \times (Q-1)$ clocks which are extremely high resulting in high test cost and performance degradation. On the other hand, if one employs the proposed test mechanism, it costs $80 \times (P-1) \times (Q-1)$ clocks. Therefore, 250-80=170, i.e., 68% clocks per test round are saved. In the S-Model, only one router is selected at a time in order to test its I/O channels for shorts. Therefore, the test rounds needed by the model vary with the size in terms of the number of routers/nodes in a network. If one brings the test mechanism, it is executed in two phases- one for input channels and another for output channels that cost 80 clocks. Therefore, overall test cost will be $80 \times P \times Q$ which is comparatively very high. The S-model, as a result, cannot be economic with network size. In the I-Model, a subnet of 16 nodes at a time is put in a test

mode in a $P \times Q$ network. The model works well for a network where the number of nodes $|P \times Q| > 16$. Otherwise, the approach makes the test application like in an off-line mode. In the model, the test mechanism is executed twice and costs 80 clocks. Therefore, the I-Model at least takes $80 \times \lceil |P \times Q|/16 \rceil$ clocks as the test time in a $P \times Q$ network. Though the I-model apparently looks to be time efficient but takes higher test time on the increasing size of the $P \times Q$. This drawback is overcome by the D-model where the $Tits$ vary with the order of $(P + Q - 1)$ which seems to be much lower than $\lceil |P \times Q|/16 \rceil$ for larger $P \times Q$ networks. Thus, the D-Model takes $40 \times (P+Q-1)$ clocks on a $P \times Q$ network. So far, from the analysis on the test time metric, it is observed that the test time linearly increases with the number of test iterations that depends on the network size. On the contrary, the proposed Q-Model presents the constant number of test iterations resulting same test clocks irrespective of the network size. Now, the Q-Model takes at most $Tits = 8$ iterations to detect channel-shorts. Consequently, the test cost becomes 320 clocks. It shows that Q-Model reduces the test time significantly. Consequently, the Q-Model becomes much faster and it is up to 16× for the $P \times Q$ networks studied here. Subsequently, the Q-Model saves test clocks up to 11.11–93.75% over the existing models for the $P \times Q$ networks.

6.8.3 Benefits in Coverage Metrics and Channel Errors

Next, the analysis of another quality metric namely, channel error as the realization of the short faults experienced on the channels of the network goes for. The fault simulations are conducted on the networks characterized in Table 6.5. The channel-shorts are injected into the networks and are simulated as per the subnet selection by a test model. The shorts are analyzed as mentioned in the sequence of the DWs, CWs, and HWs. Each approach has achieved 100% test as well as fault coverage metrics on the test applications. Consequently, Tables 6.11, 6.12, and 6.13 reflect the payload error, misrouting error, and timeout error, respectively as caused by the channel-shorts on conducting the fault simulations by the set of test models selected here. For example, the 2×2-Model on the 3×2 network senses 70.36%, 13.34%, and 16.30% as the payload, misrouting, and timeout error, respectively.

6.8.4 Benefits in Performance Overhead

The analysis on the performance overhead among the test models on the networks of 16-bit channels is completed. As the test iterations in prior models are higher, it is more likely that many application packets get passed through multiple test iterations. It means the packets may be routed through multiple faulty channels. Since, the channel shorts cause packet duplication, the size of received packets in a network are more. Figure 6.21 reveals the traffic (packet flits) size received in the networks. For example, the 2×2-Model observes 4.27–7.83X of the injected traffics while the Q-Model observes up to 4.5X of the injected

Table 6.11: Comparison of payload error (%) analysis in $P \times Q$ NoCs.

N/w	Test Models					
Size	2x2-Model [38, 39]	S-Model [6, 7]	**I-Model**	**2hop-Model**	**D-Model**	**Q-Model**
2×2	70.44	70.56	69.84	69.19	69.88	70.60
3×3	70.36	69.29	73.97	71.36	70.43	69.30
4×4	70.19	72.27	69.36	71.05	70.37	70.08
5×5	71.31	71.81	71.21	70.93	70.81	70.97
6×6	70.64	72.89	68.76	70.21	71.37	69.89
7×7	70.27	69.84	72.77	70.56	70.56	71.15
8×8	69.63	70.31	73.35	70.84	71.04	70.20

Table 6.12: Comparison of misrouting error (%) analysis in $P \times Q$ NoCs.

N/w	Test Models					
Size	2x2-Model [38, 39]	S-Model [6, 7]	**I-Model**	**2hop-Model**	**D-Model**	**Q-Model**
2×2	13.31	13.38	14.88	15.24	13.65	13.98
3×3	13.34	14.07	11.81	14.02	13.75	16.36
4×4	13.78	13.39	14.85	13.04	17.09	13.65
5×5	14.12	14.04	15.34	13.98	13.23	14.78
6×6	17.17	13.92	15.11	14.02	14.11	15.14
7×7	14.56	14.63	13.29	15.09	15.27	13.48
8×8	13.26	14.54	13.92	14.29	13.13	13.53

Table 6.13: Comparison of timeout error (%) analysis in $P \times Q$ NoCs.

N/w	Test Models					
Size	2x2-Model [38, 39]	S-Model [6, 7]	**I-Model**	**2hop-Model**	**D-Model**	**Q-Model**
2×2	16.25	16.06	15.28	15.57	16.47	15.42
3×3	16.30	16.64	14.22	14.62	15.82	14.34
4×4	16.03	14.34	15.79	15.91	12.54	16.27
5×5	14.57	14.15	13.45	15.09	15.96	14.25
6×6	12.19	13.19	16.13	15.77	14.52	14.97
7×7	15.17	15.53	13.94	14.35	14.17	15.37
8×8	17.11	15.15	12.73	14.87	15.83	16.27

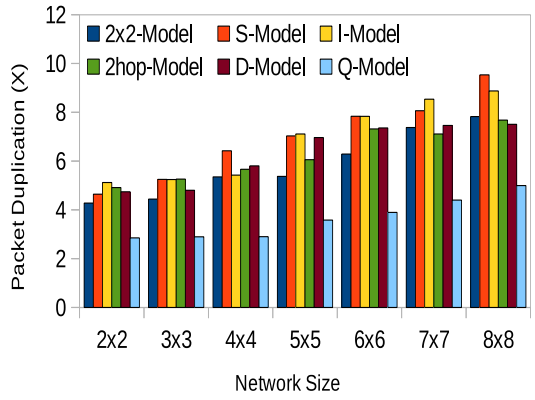


Figure 6.21: Comparison on the size of received traffics by a test model on the networks.

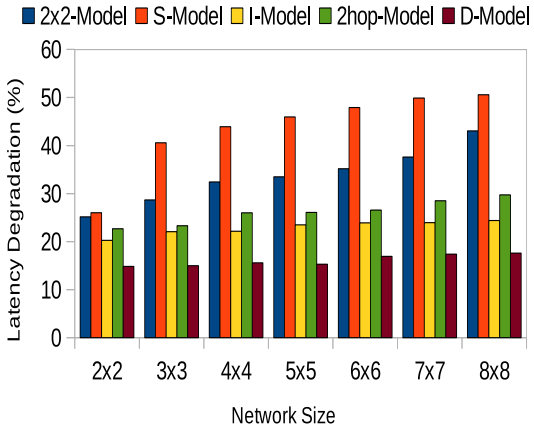


Figure 6.22: Packet latency degradation by a set of test models over the Q-Model.

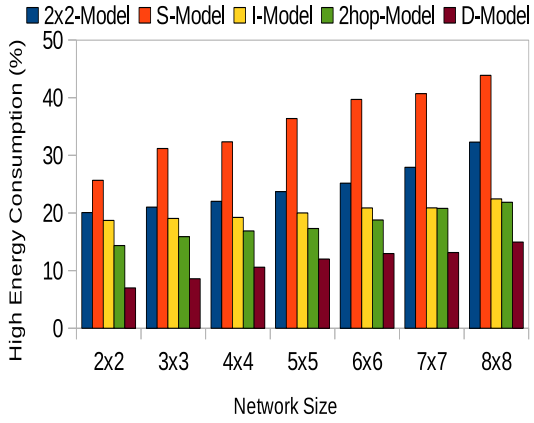


Figure 6.23: Extra packet energy consumption by a set of test models over the Q-Model.

traffic. Also, it is noticed that the S-Model receives more traffic (4.64–9.53X) as compared to other models. At the failure modes caused by the channel-shorts, the duplicated packets are transported on the routing paths before reaching the destination. Consequently, packet latency and energy consumption are increased. Figure 6.22 demonstrates the packet latency degradation of the prior models over the Q-Model. For example, the latency degradation by the 2×2-Model, S-Model, I-Model, 2hop-Model, and D-Model are respectively observed as 25.16% 26.01%, 20.25%, 22.67%, and 14.98% on the 2×2 mesh network when on executing the proposed test mechanism. The latency degradation by the models on the 8×8 mesh network is observed as 43.04%, 50.67%, 24.39%, 29.73%, 17.58%. Thus, the Q-Model reduces packet latency up to 14.98–50.67% as compared to the prior models on the set of networks. Figure 6.23 demonstrates the energy consumption by a packet flit by the prior models over the Q-Model. For example, the 2×-Model, S-Model, I-Model, 2hop-Model, and D-Model

respectively, consumes 20.07%, 25.66%, 18.69%, 14.34%, and 6.83% more energy than the Q-Model on the 2×2 mesh network. On the other hand, the prior models need 32.29%, 43.89%, 22.44%, 21.86%, and 14.93% more energy to transport the packet flits. As a result, the Q-Model improves packet energy consumption by 6.83–43.89% for the NoC set included here. Note that, the improvements grow with NoC size.

6.9 Limitations

Today's multicore, multiprocessor SoCs (MPSoCs) require a complex network-on-chip (NoC) as communication channels, which often suffer from various manufacturing faults. Such physical faults cause system-level failures and subsequent degradation of reliability, yield, and performance of the computing platform. These faults are exclusively addressed in this dissertation through Chapter 3–5. In addition to a permanent fault, on-chip channels can experience a temporary fault, such as the transient fault. In this contribution both permanent and temporary faults are detected. In addition to the detection of channel faults, a theoretical test energy model is presented in order to make the trade-off between the test and network resource utilization in terms of dissipated energy. This is done by providing a new test scheduling scheme that segments a network into four (almost) equal parts followed by the application of the test algorithm for channel faults. For the illustration of the proposed partitioned based test solution, channel-short faults are taken in the analysis. The main drawbacks of this work are provided below.

- **Coexistent Channel Faults:** One disadvantage of this work like other works discussed in Chapter 3–5 includes that single fault model, i.e., only stuck-at, only open, and short faults, are accounted in the channels one at a time. These permanent faults can, however, appear simultaneously in the channels. Subsequently, the reliability, network performance, etc. are noticeably affected by the coexistent nature of the faults.

- **Scope:** The test solutions discussed in Chapter 3–6 have overcome the scope and scalability limitations of most of the commonly practiced test approaches that consider mesh-based NoC channels only, and do not perform well for other topologies such as the octagon, κ-octagon, and spidergon networks with regard to testing time and overhead issues. However, a little emphasis is given on the untraditional NoCs by applying the proposed solutions on the octagon NoC only. The untraditional networks have also shown that these networks can satisfy high demand of performance requirements in many multiprocessor architectures.

As a result, bridging the gap of testing the channel faults between traditional and untraditional NoCs should be reduced by addressing the issues of co-existent channel-

faults and advancing the scope of the test mechanism beyond mesh and octagon to other untraditional networks.

6.10 Conclusion

This chapter has presented a cost-effective, time-optimized test solution with the capability of detecting and diagnosing channel-shorts. The intra-shorts beyond pairwise occurrences and pairwise inter-shorts have been addressed in both interswitch and local channels. The fault detection mechanism has stated faultiness/non-faultiness of the channel-wires. The diagnosis mechanism, on the other hand, has identified the set of faulty channel-wires. Additionally, detection of other faults, such transient faults in channels has been theoretically studied by the proposed scheme. A convenient test scheduling scheme has ensured constant test time for all NoCs at fixed channel width and made the proposed model to scale for larger NoCs. Experiments have shown little area overhead to execute the test mechanism and achieved 100% coverage metrics. Simulation results observed during on-line evaluation of the proposed test solution, have revealed significant effects of channel-shorts on network performances. The partitioned based test solution presented in this chapter shows that best results can be obtained by using this solution for the analysis of co-existent channel-faults and advancing the scope of the test mechanism beyond mesh and octagon to other untraditional networks. These aspects are studied in the next chapter.

Chapter 7

Optimal Detection and Diagnosis of Coexistent On-Chip Channel-Faults

7.1 Introduction

With continuous silicon technology scaling, chip multiprocessors (CMPs) over the last decade are witnessing the demand for high-performance computing as well as faster communication to support parallel applications. In today's market, SoC-based complex designs, such as Tilera CMPs and Intel's SCC, embed a number of IP cores in the range of 48-100 on a chip [192]. However, the seamless integration of many on-chip IP cores, such SoCs often fail to sustain high-volume and high-speed communication among their components due to the use of global buses as the interconnects. In order to provide better resource utilization, high bandwidth requirements, and high interconnect performance, NoC designs include multiple complex components, newer interconnection topologies, and many advanced features, resulting in increased design complexity and size [193, 194]. Consequently, checking the functional correctness of the NoCs has become challenging, particularly in the presence of various faults in the communication architectures arising due to many physical phenomena such as process variation, misalignment, defects, and wear-out. The typical fault model used for an NoC includes shorts and stuck-at faults on communication channels [36].

Unfortunately, the manufacturing channel defects (short and stuck-at faults) bring the network into notable physical failures. For example, short faults act as an agent to packet duplication, misrouting, and dropping while stuck-at faults (SAFs) act as an agent to packet corruption, misrouting, and dropping. When these faults occur at the same time in a channel, combined effects of these faults may severely degrade network performance. The manufacturing defects are the physical faults that last forever and may render the chip useless. Simultaneously, the size of permanent defects in channels is ever increasing due to the advanced silicon manufacturing processes [155]. The increasing fault rates, therefore,

may cause a major concern to tackle while ensuring the desired reliability, yield, performance of the system.

The increasing complexity and network size deservedly support both homogeneous and heterogeneous applications but worsen the reliability and related issues like yield improvement when permanent faults exist in channels. To address these issues via ensuring the correctness of the network channels and protect the network from the possible physical failures, a set of mechanisms, such as replacement method [145], fault-tolerant routing [152, 154], etc. is practiced in the state-of-the-art. Either of the techniques (replacement or fault-tolerant) can be adapted to make the network functional and maintain network performance only when they have the knowledge of the health status of the channels. In other words, all faulty channels must be identified prior to application of the techniques. One basic shortcoming of these techniques is that such schemes do not usually comprise a test mechanism [6]. To ensure the ability to accommodate manufacturing channel-faults and maintaining post-manufacturing operations, a key requirement for modern NoC architectures is to design a suitable test method that can as frequently as possible detect and diagnose the channel faults within the *incubation period* [33]. The incubation period as defined by Huang *et al.* [33] is the time duration between the occurrence of a physical failure and the first observation of a fault in the network. If it is possible to detect the faults (here channel's faults) within this incubation period, the diagnosis information can be purposely reused by the replacement and fault-tolerant approaches. Thus, an effective test solution is highly needed to integrate into an NoC-based system. In this chapter, a cost-effective test scheme is presented for the detection and diagnosis of coexistent short and stuck-at faults in NoC-channels and evaluate the system-performance during runtime. The proposed scheme is applicable to general NoCs that range from conventional (e.g, 2D mesh) to unconventional interconnection topologies such as octagon or spidergon.

The rest of the chapter is organized as follows. Motivation and contributions to this work are mentioned in Section 7.2. Section 7.3 describes the proposed test mechanism. The proposed test-scheduling is discussed in Section 7.4. Section 7.5 reports simulation results. The scalability and adaptability of the proposed test solution are described in Section 7.6 and Section 7.7, respectively. Section 7.8 presents benefits of the proposed solution over the prior schemes. Concluding remarks appear in Section 7.9.

7.2 Motivation and Contribution

Most of the existing test approaches have been discussed with mesh networks. However, there are many other unconventional networks, e.g., octagon that can meet performance requirements of network processor SOCs. Advantages offered by octagons include low design cost, fair performance, and scalability. These parameters make the architecture suitable

for supporting aggressive on-chip communication demand of networking SOCs and related domains [87]. Unlike on a mesh, an octagon network is designed to establish communication at most two hops between any two nodes. Like a mesh, manufacturing faults can be experienced in channels of the octagon and similar NoCs, such as Spidergon [89] resulting in significant performance degradation. The motivation behind this work is to meet the demand of devising and analyzing a quality of reliability and performance (QoRP) aware, distributed test solution that addresses the coexistent short and stuck-at faults (CSSAFs) in channels of and investigates the impact of the faults on various performance metrics on unconventional and conventional NoCs. A fixed cost and fast general test algorithm is presented at a node (router and its core) to detect and diagnose the CSSAFs in channels of an NoC. Additionally, the algorithm is used to detect temporary faults, such as transient and aging faults. Then a test module (TM) is designed at the node to execute the test algorithm. However, to reduce overall test time and subsequent performance overhead, the availability of TM is insured at both router and its dedicated core. It is seen that coexistent nature of short and stuck-at faults has made some faults to be undetected by the test algorithm. An attempt investigates this fault non-diagnosability. The test-scheduling methodology that can trade-off the overall test time with resource utilization while handling faults. One clear advantage of the scheduling scheme is that it does not only scales the test solution with atypical NoCs like the octagon, spidergon but can also be adapted to any other typical topologies, such as the mesh. The scalability can be shown with respect to network size, channel width, and network type.

Synthesis result shows that hardware area overhead of a TM block implemented on FPGA hardware is 16.54–25.97%. Experimental results reveal 100% (and near-92%) fault coverage when shorts and SAFs are simulated independently (and together). The loss of fault coverage in the second case (while short and stuck-at are coexistent) arises because of few short and stuck-at faults become undetectable which are treated as the non-diagnosable faults. However, 100% link coverage is always achieved. Simulation results for on-line evaluation of the proposed scheme reveal the impact of the CSSAFs on common performance metrics under large traffic scenarios. Side by side, it is observed that the proposed scheme reduces hardware area overhead up to 37% showing $\approx$ 60% improvement, becomes $\leq 55\times$ faster resulting up to 98% reduction in test time, and lowers fault non-diagnosability by 13%. Consequently, the performance overheads in terms of packet latency and energy consumption are improved by 67.05% and 54.69%, respectively.

The major contributions to this work can thus be summarized as follows:

(1). Detecting CSSAFs in channels. It indicates the health status of a channel-wire in terms of its state of faultiness/non-faultiness.

(2). Diagnosing the CSSAFs to identify faulty channel-wires so that a fault-tolerant routing may by-pass them during transmission.

(3). Identification of diagnosable and non-diagnosable faults.

(4). Generalization of the test scheduling scheme towards lowering the test cost in terms of the fixed test time and resource utilization, and widening its scope from unconventional to conventional networks.

(5). Studying the effectiveness of the proposed test mechanism in terms of many quality metrics, such as hardware area overhead, test time, coverage and performance metrics on the basic octagon network.

(6). Establishing the solution scalability with respect to a larger network which may consist of multiple basic octagon networks or spidergon networks.

(7). Establishing the solution adaptability with a mesh network.

(8). Observing the direct benefits incurred by the proposed solution with respect to many quality characteristics over a set of notable works.

7.3 Proposed Test Model

A built-in-self-test (BIST) based test mechanism is a common practice to detect and diagnose the permanent faults in digital circuits. In this section, an on-line BIST test algorithm is presented for the CSSAFs in interswitch and local channels in NoCs. The BIST structure called the test module (TM) drives the test mechanism by exercising specific test input sets in order to cover the potential faults (CSSAFs) in the channels under test. The faults in the channels can be diagnosed on the basis of the test responses of the test input sets. Though the algorithm can detect all short and stuck-at faults in the channels when they are supposed to exist alone, in practice, these faults may, however, be coeval. It means these faults may be experienced simultaneously in the channels. This nature of the faults raises the issue of fault non-diagnosability. It is noted that during testing, a high-level fault model for the channels is used. Thus, it is essential to represent the concerned fault model and TM architecture before attempting the testing of the channels and the related issue of fault non-diagnosability.

7.3.1 Coexistent Short and Stuck-at Fault Model

The fault model of interest must be based on the likely failures observed due to manufacturing reasons in NoC channels. The most manufacturing defects in channels are transformed into permanent faults and realized as common logic-level short and stuck-at faults. The shorted wires either belong to same or distinct channels. Accordingly, short faults are categorized to *intra-channel* and *inter-channel* shorts. A stuck-at fault is described at two states: stuck-at-0 (SA0) and stuck-at-1 (SA1).

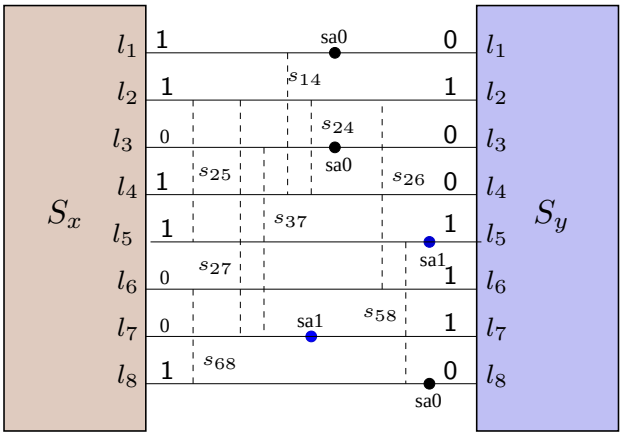


Figure 7.1: Example of CSSAFs in a channel (S_x, S_y).

Definition 7.1. ***Short Fault:*** *When a group of $k(k \geq 2)$ independent channel-wires accidentally suffer from electrical interconnection, they are said to be "shorted"; the corresponding defect is modeled as a channel-short fault, whereas in the normal mode, they should be electrically disconnected from each other. For example, a short fault between two (k=2) wires l_i, l_j is denoted by s_{ij}.*

Definition 7.2. ***Stuck-at-0 Fault:*** *It is a manufacturing defect on a channel wire l_i that assumes to be stuck at the logic-0 for any information on the wire. The fault is labeled as sa0 and its occurrence is termed as an instance y_i.*

Definition 7.3. ***Stuck-at-1 Fault:*** *It is a manufacturing defect on a channel wire l_i that assumes to be stuck at the logic-1 for any information on the wire. The fault is labeled as sa1 and its occurrence is termed as an instance z_i.*

Figure 7.1 shows a high level representation of the CSSAFs in an interswitch channel shared by routers S_x and S_y. One can assume intra-channel shorts in the best and worst cases. In the best case, pairwise $(k = 2)$ shorts are followed by a set of prior works [7, 39]. It will be a too simplistic case due to the high closeness among the wires in a channel since the demand of high bandwidth requirement is satisfied on implanting multiple wires in the channel. Oppositely, it is similarly pessimistic to assume all n wires in the channel are affected by short faults. The occurrence of intra-channel short faults should legitimately be considered in order to properly observe their influences on the system performance. As the size of k increases, occurrences of *intra-channel* shorts decrease. It is observed that this short beyond $k > 4$ is negligible. Further, if two channels in an NoC are sufficiently separated, the probability of *inter-channel* shorts is also negligible. It thus suffices to consider intra-channel shorts for $k \leq 5$ and inter-channel shorts for $k = 2$ at a node in order to closely observe their impact alongside the SAFs on NoC performance. Thus, the size of intra-channel shorts #S1, inter-channel shorts at a node #S2, and total short faults #S in an NoC architecture are

$$\#S1 = \sum_{2 \leq k \leq 5} \frac{n!}{k!(n-k)!} \times x_k \tag{7.1}$$

$$\#S2 = \frac{(mn)!}{2!(mn-2)!} \tag{7.2}$$

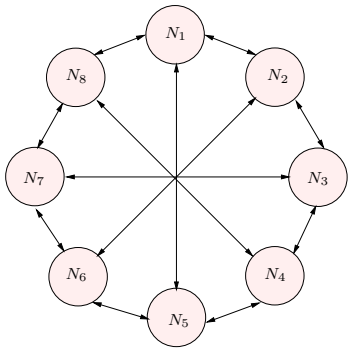
$$\#S = \begin{cases} \complement \times \#S1 \\ + \\ \aleph \times \#S2 \end{cases} \tag{7.3}$$

$$\#SA0 = \sum_{i=1}^{n} l_i \times y_i \tag{7.4}$$

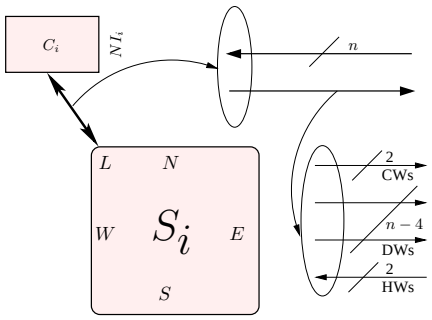
$$\#SA1 = \sum_{j=1}^{n} l_j \times z_j \tag{7.5}$$

$$\#SA = (7.4) + (7.5) \tag{7.6}$$

$$\#SAF = \complement \times \#SA \tag{7.7}$$



(a) Abstract view of an octagon NoC.



(b) General view of a 5-port node. CWs: control wires, DWs: data wires, HWs: handshake wires.

Figure 7.2: General representation of an octagon NoC and a node with five I/O ports.

defined in Equations 7.1, 7.2, and 7.3, respectively. Here, $m, \complement, \aleph$ are the number of channels of a node, the number of channels in an NoC, and the number of nodes in the NoC, respectively. In each *n-bit* channel, #SA0, #SA1 have been defined in Equations 7.4 and 7.5, respectively. Now, each wire either suffers from an SA0 or an SA1 fault. So, one must go for testing #SA (Equation 7.6) faults in the channel. Likewise, #SAF (Equation 7.7) faults in the NoC must be tested. In this work, it is treated that a channel has experienced intra-channel shorts up to $k \leq 5$. So, an SA0 (or SA1) fault can appear before/after a short fault on a wire in the channel.

In this work, a demonstration of the scheme starts with a basic octagon NoC shown in Figure 7.2. The network consists of $\aleph = 8$ nodes (8 routers and 8 dedicated IP cores), and $\complement = 40$ unidirectional channels. Each channel has single bit 16 wires (i.e., channel width *n=16-bit*). Each node N_i (Figure 7.2a) is the pair of router S_i and its dedicated core C_i. Each router has five active I/O ports (Figure 7.2b). Input and output ports are connected to incoming and outgoing channels. Three I/O ports are used for interswitch channels in the north (N), south (S), east (E), or west (W) direction while the fourth I/O port is dedicated for the local channel in the "L" direction. Another I/O port is kept for future use, such as to build a larger octagon network using multiple basic octagon units or to built a square octagon

NoC (SONoC) [195]. The channel wires are categorized into three sets: CWs, DWs, and HWs. Both CWs and HWs are assumed to be of 2-bit width while rest of the channel width i.e., the *n-4* bit is assigned for the DWs. The potential faults are modeled on these wires.

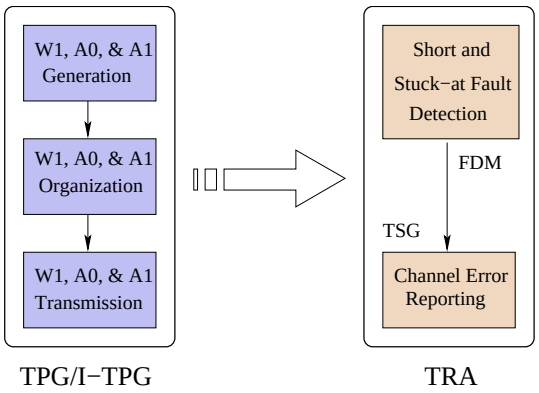


Figure 7.3: A general view of test module for addressing CSSAFs in channels.

7.3.2 Test Module and Packet Format

The key element for driving the test mechanism is a typical BIST structure, a small hardware unit known as test module (TM) (Figure 7.3). The TM is structured with different components. These components are used to generate the test sets and analyze the received test responses. The components are categorized into two blocks: test pattern generator (*TPG*) and test response analyzer (*TRA*). To have minimum performance overhead and lower test time, one may take advantage of a TM at routers and cores. The *TPG* derives raw test sequences including header and trailer and organizes them into packets. The header and trailer are mandatory as they carry important routing information. In addition to its assigned functionalities, the TPG unit has the capability to send test packets. The TM blocks in the cores and the routers are built in such a way that the channels from a node can undergo the test in parallel by the proposed test mechanism. Now, an IP core shares a local channel with its local router. On the other hand, the router in the node does not only share this local channel but also shares interswitch channels with its neighbor routers. Thus, the core-TPG is made to unicast the test packets on the local channel while the router-TPG multicasts the test packets to local and interswitch channels. Multicasting is done by integrating extra circuits (*Xckt*) to the router-TPG. This block is also referred as the integrated *TPG* (*I-TPG*). A *TRA*, whether in the core or router, extracts the test patterns as test responses and diagnoses each received test pattern. This extraction followed by diagnosis is done using its inbuilt fault diagnosis module (FDM). Further, on the basis of the diagnosis information which has identified a faulty channel wire, another in-built unit referred to as TRA signal generator (TSG) reports a channel error. As mentioned earlier, short and stuck-at faults put a network into many system level failure modes. These failure modes are realized in terms of errors in

Table 7.1: Typical packet format using W1, A1, A0 test patterns for the detection of coexistent short and stuck-at faults in n-bit channels.

Header	Payload Flits			Trailer
Flit	$W1$	$A1$	$A0$	Flit

the channels. Generally, the channel errors lead to payload error (PE) due to faults on data and handshake wires, misrouting error (ME) and dropping/timeout error (TE) due to faults on control wires that carry packet header and packet trailer, respectively.

For detecting permanent faults in NoC channels, the test mechanism exercises the input test patterns. It is necessary to carefully select a test pattern not only from the testing viewpoint but also from the fault coverage viewpoint because sufficient and fitting input test sets can improve the fault coverage. The shorts and SAFs are two different faults that subsequently need separate test patterns. Without loss of generality, well known Walking-One ($W1$) sequence is exercised for shorts, All-One ($A1$) and All-Zero ($A0$) sequences for $SA0$s and $SA1$s, respectively. NoCs are packet switched networks. The test input sets like any application data must be sent in packets. A typical test packet format is shown in Table 7.1. Each packet has three parts called header, payload, and trailer flits. Sufficient test sequences are placed in these parts whenever CWs, DWs, and HWs are under test. To begin the illustration, the channel-width is *n=16*-bit, which can equivalently be treated as a 16-bit single bus. Therefore, to cover the CSSAFs modeled in a channel, one needs 18 test flits ($|W1| = 16, |A1| = 1, |A0| = 1$). The test packet size is kept as small as possible so that other logic faults like packet deadlock [196] can be avoided at the run-time. The 16 *W1* sequences are organized into 8 test packets while each of *A1,A0* are placed in two separate packets. The proposed test mechanism is applied in the on-line mode where a part of the NoC undergoes testing of channels and the rest part allows an application. Therefore, it becomes a necessity to differentiate a test packet and an incoming application packet at a router. It is the responsibly of the arbiter in the router to do the task. The arbiter is connected with the *Xckt* of the router-TPG and gives preference to test packets over application packets. Therefore, arbiter does not allow an incoming application packet on the outgoing channel currently in test mode, though the packet needs forwarding by the router over the channel. The arbitration logic in the arbiter monitors, and recognizes incoming packets as test or application type. The inputs of the logic are controlled by multiplexers that switch between normal (application) packets or test packets.

7.3.3 Testing of CSSAFs in Channels

To address the faults in channels, one must need a test procedure that aids in detecting and diagnosing the faults. In this chapter, the proposed test mechanism targets contemporane-

ousness shorts and SAFs in channels and is applied in the on-line mode. Here a test method is presented for the CSSAFs using functional mode of a network except the channels taken for testing. Further, the faults are assumed to be present on the DWs, CWs, and HWs which undergo the testing in sequence. The proposed test method at a node is primarily executed in two stages: test packet transmission and received packet analysis.

7.3.3.1 Packet Transmission

The test packet transmission as the first working stage in the test mechanism is executed by sender-TPGs (core-TPGs, router-TPGs) which are treated as *test sources*. W1, A1, A0 as test input patterns are derived and packeted at the test sources. Once a test packet becomes available, the TPGs initiate transmission of the packet simultaneously. As mentioned earlier, the *core-TPG* unicasts each test packet to its neighbor router only while the *router-TPG* multicasts each packet to its authorized core and neighbor routers. The content of the test packets at the core and router are same except the header part.

7.3.3.2 Test Packet Analysis

The test packet analysis as the second working stage in the test mechanism is completed by receiver-TRAs (core-TRAs, router-TRAs) treated as the *test destinations* that basically do extraction and pattern checking of response flits with a corresponding local test flit. Note that the routing distance between a test source and destination is single hop. When a TRA receives a test flit (treated as response flit) from its neighbor TPG over a channel under test, the status of the flit may be affected with a fault, if any on a channel-wire. Then the health status of the wire is ensured by faultiness. The correctness in terms of the health condition of channel wires is exploited as the basis to partially ensure reliability in the NoC. The improvement on the issue can be done by employing a fault-tolerant scheme that requires the knowledge about the set of faulty wires in channels and copes with the emerging faults (short, stuck-at, transient) in the on-line mode. Additionally, the identification of faulty channel wires essentially helps to increase the production yield of the network in the off-line mode [197]. The test packet analysis in terms of response flit analysis focuses on the faultiness/correctness of a channel wire by the fault detection and identification of faulty wires by fault diagnosis. This concept is illustrated through examples for short and stuck-at faults. Figure 7.1 shows a short fault involving wires $\{l_2, l_4, l_5, l_6, l_7\}$ and five stuck-at faults in the channel (S_x, S_y).

7.3.3.2.1 Short Fault Analysis Consider a group of k=5 wires $\{l_2, l_4, l_5, l_6, l_7\}$ in the channel (S_x, S_y) without any stuck-at fault and attempt a test for short fault on l_2 with other wires in the group. To ensure that only l_2 is under test, it is set to "1", i.e., S_x'TPG transmits a W1 test vector as the test flit containing a single bit "1" at position l_2. Then, 0s are set for other wires in the channel and included in the test flit. Thus, the W1 test flit for l_2 is "0100 0000" (Figure 7.4). When the bit "1" is received by S_y's TRA over l_2, the response flit for this

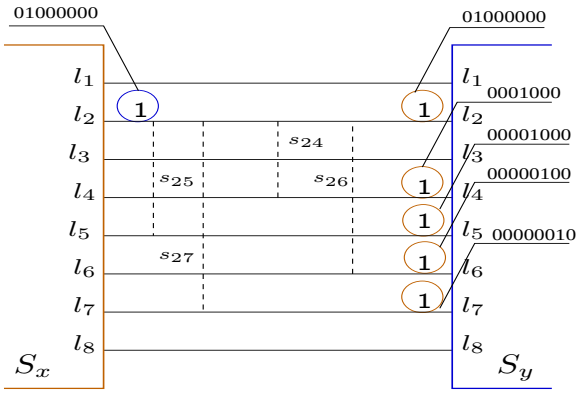


Figure 7.4: Transition of a test flit for shorts fault in l_2, l_4, l_5, l_6, l_7 of channel (S_x, S_y).

$$\oplus \begin{cases} 01000000 \\ 00010000 \\ 00001000 \\ 00000100 \\ 00000010 \\ 01000000 \\ - - - - \\ 00011110 \end{cases} \tag{7.8}$$

wire is "1" where 0s are expected for other wires. The TRA thus extracts the response flit as "0100 0000" which is same as the test flit. Since five wires $\{l_2, l_4, l_5, l_6, l_7\}$ are considered in the fault example, the TRA collects five response flits corresponding to these five wires. For example, if there is any short between l_2, l_4 wires, the bit "1" transmitted on l_2 is also received over l_4. S_y's TRA extracts the response flit "0001 0000" at l_4. The TRA extracts the received value at l_4 and for other positions 0s are explicitly placed. Similarly, as there are shorts between l_2 with wires l_5, l_6, l_7, the TRA observes response flits "0000 1000", "0000 0100", and "0000 0010" on these wires, respectively. To ensure that wires l_4, l_5, l_6, l_7 are shorted with l_2, these five response flits are *XORed* by the TRA's FDM. The logic operation uses the local "0100 0000" (already derived by the S_y's TPG) and is shown in Equation 7.8. The 1s in the result ensure that l_2, l_4, l_5, l_6, l_7 are shorted together and can be treated as the faulty wires in the channel (S_x, S_y). Thus, underlying NoC senses the packet duplication failure mode.

7.3.3.2.2 Stuck-at Fault Analysis Consider that the channel (S_x, S_y) (Figure 7.1) does not have short faults and attempt to test those five stuck-at faults. S_x's TPG transmits test flits "1111 1111" (*A1*) and "0000 0000" (*A0*) in turn on the channel. If there is an SA0 or SA1 fault on a channel wire, corresponding bit "1/0" in these flits is received as "0/1" at S_y's TRA. The TRA thus receives "0101 1110" and "0000 1010" as the response flits and detects SA0 faults on wires l_1, l_3, l_8 and SA1 faults on wires l_5, l_7. To ensure that these wires are actually affected by SA0/SA1 faults and treat them as faulty wires, the S_y's FDM executes two logic operations: *ANDing* and *ORing* as shown in Equations 7.9 and 7.10, respectively where 0s and 1s after the respective operations indicate the set of wires affected by SA0 and SA1 faults. One can now say that the NoC tastes the packet corruption failure mode.

The failure modes in the run-time are realized by channel errors. If the faulty wires belong to data and handshake categories, the failure modes- packet duplication and corruption, are realized as payload error (PE). If the packet header (i.e., the beginning of packet (bop) signal) on a CW is affected by a fault, the modified header is received at the TRA over the channel.

$$\wedge \begin{cases} 01011110 \\ 11111111 \\ --\,-- \\ 01011110 \end{cases} \quad (7.9) \qquad \vee \begin{cases} 00001010 \\ 00000000 \\ --\,-- \\ 00001010 \end{cases} \quad (7.10)$$

Table 7.2: The 2-bit signals for the coexistent short and stuck-at channel faults.

Bits	Error Type
00	No Err
01	PE
10	ME
11	TE

$$\oplus \begin{cases} 00001000 \\ 00000000 \\ 00001000 \\ --\,-- \\ 00000000 \end{cases} \quad (7.11)$$

A misrouting failure mode is detected. During application, the router forwards the packet to an unknown node resulting in the misrouting error (ME). Similarly, the fault may affect the packet trailer (i.e., end of packet (eop) signal) on a CW. The modified trailer received at the TRA shows that the NoC is in the packet dropping failure mode. During application, the router drops the packet. Destination node never gets the packet resulting in timeout error (TE). Besides, traffic in the network, buffer overflow of a router in a routing path, etc. may enhance the error since the destination may not receive the "eop" signal in the predefined time interval. Now, different failure modes realized as channel errors are reported by a TRA's signal generator (TSG). The TRA's FDM subsequently feeds diagnosis results to the TSG block. On the basis of faulty wire types, the block generates 2-bit signals to report a channel error. The meaning of a 2-bit signal is provided in Table 7.2. The diagnosis information can also be input to a fault-tolerant scheme for maintaining reliable communication.

A channel can also be affected by other logic faults, e.g., packet deadlock. The packet delay occurs depending on various factors, such as packet size, availability of channel width, traffic in the channel, routing distance between source and destination, etc. In the worst case, this delay results in packet deadlock [7]. During testing of the channels, no application packet is allowed on the channels under test. It is further known that each test packet contains four flits to detect a short fault and three flits to detect a stuck-at fault. Therefore, such small-sized test packets have access to the full channel width and can be delivered at the normal speed of the network without any significant delay on the channels. In this work, every flit of a test packet is transmitted using a single clock. The routing distance between test source (TPG) and test destination (TRA) is only single hop i.e., one channel length. Thus, no delay in receipt of the test packet is observed. As a result, no packet deadlock is caused.

7.3.4 Non-diagnosable Faults

In practice, the diagnosis of channel-faults has the significant impact in improving the reliability of NoC-based systems and facilitating maintenance of the systems. The proposed BIST test algorithm detects and locates a potential fault in channels. Many previous works have addressed a particular channel-fault due to the assumption that different kinds of faults, such as shorts and SAFs do not exist on the same channel-wire. The corresponding fault model is named as a single fault model (SFM). When the SFM targets only shorts or SAFs in a channel, the FDM detects all these faults encountered and successfully identifies faulty channel-wires. Whence the assumption of single channel-fault is relaxed, the FDM may fail to detect a fraction of faults, i.e., the diagnosis results fail to locate all detectable faults. The consideration of different channel-faults is defined as multiple fault model (MFM). In this work, the MFM that considers both shorts and SAFs simultaneously in a channel treats some channel-wires as non-faulty although the wires are affected by a short/stuck-at fault. The non-diagnosability of these faults makes a deficiency in covering the channel faults and prevents the underlying test model from achieving 100% fault coverage metric. Consequently, performance overhead may seriously be influenced.

For the reliable communication in many applications, it does not suffice to detect what faults might have occurred in a channel. At the same time, one may show interest to know what faults have *definitely* occurred but remained undetected. The results returned by an FDM are used to decide whether a fault is diagnosable/non-diagnosable. Therefore, the fault like short, SA0, SA1 can be treated as either diagnosable (a fault that is detected) or non-diagnosable (a fault that is undetected). Equations 7.8, 7.9, and 7.10 as the diagnosis operations in previous subsection, respectively show the diagnosable shorts, SA0s, and SA1s under the SFM on the channel (S_x, S_y). Let us take the MFM version of the channel. Assume the CSSAFs on wires l_3, l_5, l_8 in the channel (S_x, S_y) and no fault on other wires of the channel. When the bits "0", "1" are transmitted over the l_3, l_5, the FDM in S_y is unable to recognize these wires as faulty because the SA0 and SA1 faults on these wires become non-diagnosable. On the other hand, if the test flit "0000 1000" is assigned for l_5 to test it for short fault with l_8, the FDM has "0000 1000", and "0000 0000" as responses on l_5, l_8, respectively. The diagnosis as seen in Equation 7.11 shows that l_5 is not shorted with l_8. Thus, the FDM is unable to detect short, SA0, SA1 faults on these wires and the faults are considered as non-diagnosable faults.

7.4 Test-Scheduling

The test time is one of the leading aspects of any test solution because a major part of the cost budget in manufacturing a product is expensed in the test cycle. In addition, the test time in the on-line mode has the direct influence on the network performance, especially on

the packet latency. An efficient test algorithm cannot itself reduce the cost in terms of the test time. However, the goal can be reached by proper selection of the way of executing the algorithm. In the proposed scheme, the test time incurred by the test algorithm is reduced at the node level and the system level. In the first case, unicast and multicast-based test packet transmission address the faults on the channels of a node. In the second case, the overall test time can be reduced by concurrently executing the algorithm at multiple nodes of a network. A system-wide implementation of the proposed solution should, therefore, conform to lower test time requirements and its trade-off with test resource utilization.

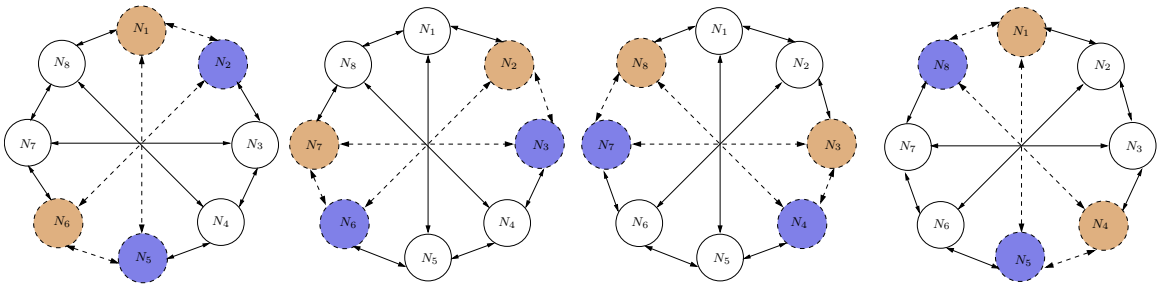
7.4.1 Scheduling Nodes

An NoC provides an efficient realization by its highly parallel and distributed nature. In fact, NoCs with the high degree of parallelism can conveniently execute the test algorithm simultaneously at multiple nodes thereby lowering the overall test time. However, the node selection must be done in such a way that network resource utilization during testing at each test iteration is (almost) same. It is possible whence (nearly) the equal number of nodes in each iteration execute the underlying test algorithm. The main advantage is that every network can be tested by constant test time. The proposed test scheduling scheme is generalized with octagon-like, and grid-like NoCs.

7.4.1.1 Octagon-like NoCs

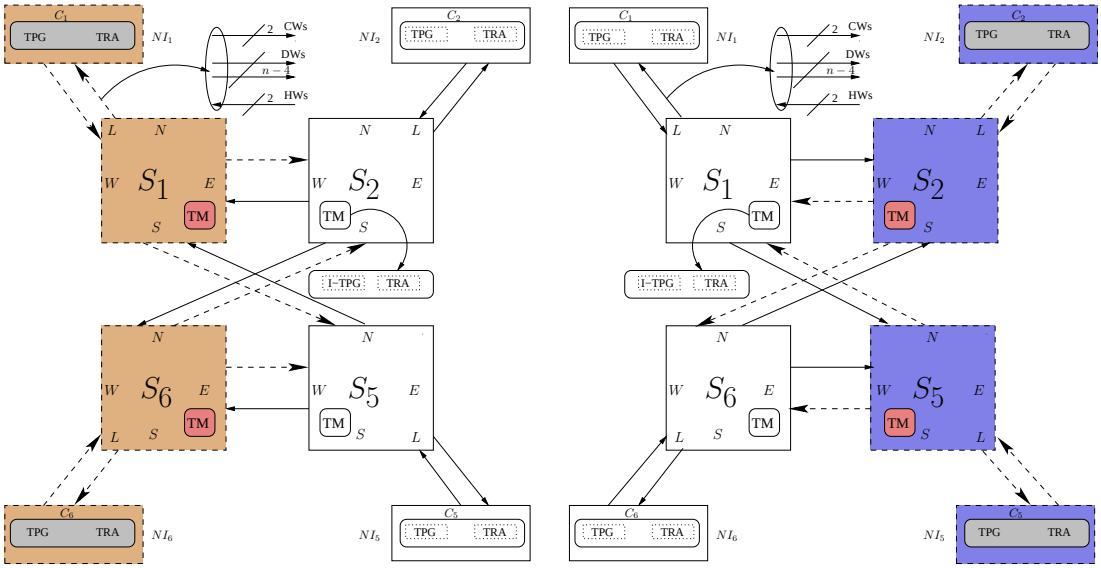
The set of octagon-like NoCs include octagon, κ-octagon (that consists of κ number of basic octagons), SONoC, and spidergon NoCs. The test scheduling on such NoCs is demonstrated with the octagon network. The interconnection of four nodes is selected and the subnet is named here *Damaru* which looks like a closed X. For the simplicity of illustration, Figure 7.5 illustrates this scheduling scheme in the octagon network. The test application on the subnet is referred to as test round (TR). The dotted interconnection represents the *Damaru* and ensures the subnet under test in a test round. Every test round is completed in two consecutive test iterations (Tits). In the first test iteration (Tit=I), half of the nodes (labeled as odd nodes) in the subnet are selected as test sources while rest of the nodes (labeled as even nodes) in the subnet are considered as the test sources in the next iteration (Tit=II). The well known 2-color graph problem can be employed to label the nodes as odd/even. Figure 7.5a shows the first test round (TR=I) or test application on the first subnet (Subnet-1) of the octagon network. Each iteration of the round is further illustrated in Figure 7.6. The test execution by odd and even nodes are denoted by brown and blue colored nodes, respectively. Subsequently, the test execution by odd nodes ($< S_1, C_1 >$ and $< S_6, C_6 >$) at the first round, first iteration ($TR = I, Tit = I$) is shown in Figure 7.6a while the same by even nodes ($< S_2, C_2 >$ and $< S_5, C_5 >$) as the $TR = I, Tit = II$ is shown in Figure 7.6b. Note that, dotted arrows show the test packet transmission in an iteration from a node. On completion of the test round,

other test rounds (TR=II, TR=III, or TR=IV) as shown in Figures 7.5b to 7.5d can likely be exhibited to cover the channel-CSSAFs. Since the same test configuration is repeated in the test rounds, the resource utilization is similar and results in the fixed number of test rounds/iterations in the NoC.



(a) TR=1. (b) TR=2. (c) TR=3. (d) TR=4.

Figure 7.5: Application of the proposed test mechanism at various test rounds in an octagon network.



(a) Test application in 1st iteration at first round. (b) Test application in 2nd iteration at first round.

Figure 7.6: Test execution at first test round (TR=I) i.e., the first subnet (Figure 7.5a) that consists of the interconnection of nodes N_1, N_2, N_5, N_6 on the octagon NoC (Figure 7.2a).

7.4.1.2 Grid-like NoCs

Most NoCs in the literature use 2D-meshes due to simpler floorplanning, and layout regularity that provide the channels with the same length. The generalization of the proposed test solution depends on the test algorithm and scheduling scheme. The test algorithm is described with respect to a node that has the similar configuration in NoCs and thus treated as a general test mechanism. Solution generalization via the scheduling scheme needs its application on both traditional and untraditional networks. The second case is already demonstrated with

octagon-like networks (Section 7.4.1.1). Now, the scheduling scheme is considered to most used common network topologies, such as mesh NoCs. Mesh as well as torus networks represent an $A \times B$ grid-like architecture. One main objective of testing channels is to earn constant test time irrespective of the network size and types. Another objective is to select similar subnets from an NoC at the test rounds so that equal amount of test resources get utilized. Both the objectives have already been described with the test scheduling on the octagon-like networks (Section 7.4.1.1). The same can be done on the grid-like architectures in order to enjoy low test cost, low-performance overhead, and other advantages. The scheduling scheme, without loss of generality, divides an $A \times B$ grid-like structure into four test regions/subnets. The division is done at the middle boundary nodes in the horizontal and vertical directions. Therefore, size of a subnet is $A/2 \times B/2$. For simplicity, one can map each subnet to a quadrant in XY plane. For example, the upper-right subnet may be treated as Subnet-1 (1st Quadrant), upper-left subnet as Subnet-2 (2nd Quadrant), lower-left subnet as Subnet-3 (3rd Quadrant), and lower-right subnet as Subnet-4 (4th Quadrant). Note that, small mesh NoCs mainly 2×2 and 2×3 (or 3×2) can be tested in just 1 and 2 rounds, respectively. As mentioned, one test region is selected at a test round where the odd and even test sources, in turn, address the channel faults. The test configuration, as before, can be repeated to other subnets to complete the testing. Further, each subnet on a mesh is (almost) of equal size, so resource utilization during testing a subnet as well as performance overhead at each test round on the mesh is nearly the same. This scheduling scheme is demonstrated with a 5×5 mesh NoC. Figures 7.7 and 7.8 represent the subnet selection and test application on the first subnet on this network, respectively. It can be viewed that each subnet on the network looks like 3×3 mesh. The odd test sources (Brown colored nodes) and even test sources (Blue colored nodes) subsequently complete the first and second test iterations.

7.4.2 Test Time Evaluation

The test time T_{ch} (Equation 7.12) required for a channel is the sum of the time T_{tpg} for deriving the test set including header and trailer, T_{tpo} for organizing the test input sets, T_{tpt} for transmitting the test packets, and T_{tra} for analyzing the test responses. Since multiple channels of a node are tested concurrently, the T_{ch} is same for them. As mentioned earlier, the proposed model divides the NoCs into fixed four test regions independent of the size and type of the networks and every test round on a test region addresses the channel-CSSAFs in two test iterations. Now, multiple nodes in an iteration execute the test algorithm in parallel i.e., multiple channels undergo the testing concurrently. Therefore, the time T_{it} (Equation 7.13) required to finish testing of the channels from the nodes in a test iteration equals to T_{ch}. Utilizing the advantages offered by an NoC such as regularity, controllability, and concurrent transmission of test packets from test sources, all modeled faults in the NoC architecture

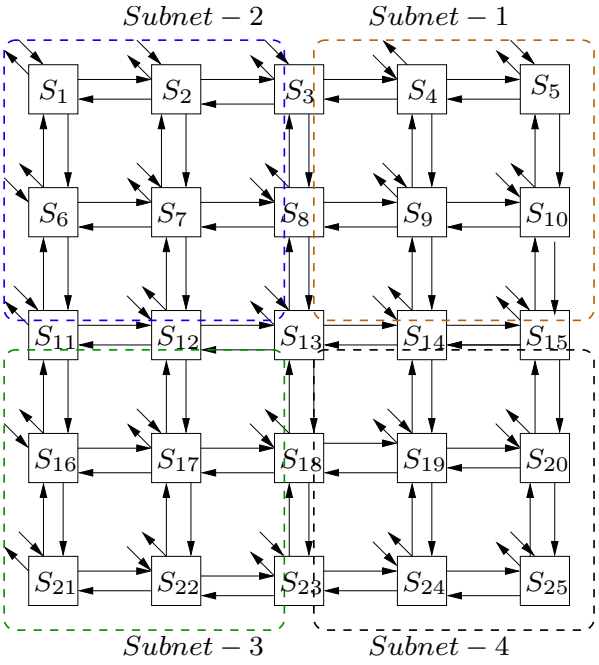
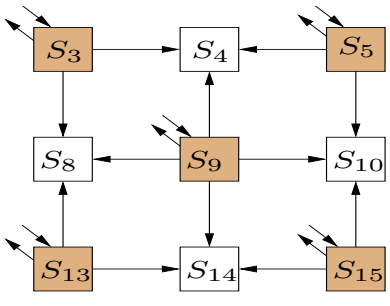
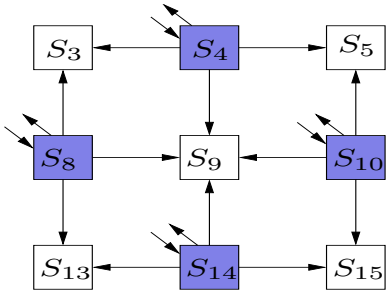


Figure 7.7: Subnet classifications on a 5 × 5 mesh network.



(a) TR=I, Tit=I



(b) TR=I, Tit=II

Figure 7.8: Test application at first round (Subnet-1).

$$T_{ch} = T_{tpg} + T_{tpo} + T_{tpt} + T_{tra} \quad (7.12) \qquad T_{it} = T_{ch} \quad (7.13) \qquad T_{n/w} = 8 \times T_{it} \quad (7.14)$$

can be addressed by just eight test iterations. In other words, the overall test time $T_{n/w}$ (Equation 7.14) for the network is $8 \times T_{it}$ clocks. For example, by shifting the *Damaru* configuration in clockwise/anti-clockwise direction on the octagon network (Figure 7.5), one can complete the task of testing the channel's CSSAFs by 8 test iterations.

7.5 Simulation Results

The efficiency of the proposed test solution on an octagon and other NoC architectures is evaluated in terms of various important quality metrics. Here, four types of quality metrics, namely silicon-hardware area overhead, test time, link and fault coverage, and performance are considered. In this section, the evaluation of the proposed *Damaru*-based test scheme is carried out on the octagon network with a channel of 16-bit width (referred to as 16-bit octagon network). The evaluation is extended in Sections 7.6 and 7.7 where the scalability and adaptability properties of the proposed solution are demonstrated with other networks, such as Spidergon and mesh. The basic architectural characteristics of the 16-bit networks considered in the work are described in Table 7.3.

For the purpose of evaluating the performance, faults in channels of the networks are injected and simulated to observe their impacts. The possible number of instances for shorts

Table 7.3: Characteristics of 16-bit networks.

Network	#R	#C	#Ch	#R-R	#R-C	#W
Octagon	8	8	40	24	16	640
Spidergon	16	16	80	48	32	1280
5×5 Mesh	25	25	130	80	50	2080

and SAFs in the channels as discussed in Section 7.3 are provided in Table 7.4. For example, first #S1=274720 (at $x_k = 1$), #S2=65024, i.e., #S=339744 shorts are injected in channels of a 16-bit octagon network. Next, #SA0=3840 (at $1 \leq y_i \leq 6$), #SA1=3840 (at $1 \leq z_j \leq 6$) i.e., 7680 SAFs are added in turn at different locations- before/after a short is injected on channel-wires. Thus, the proposed *Damaru*-based test configuration handles 347424 CSSAFs on the 16-bit octagon NoC.

Table 7.4: Size of short and stuck-at faults injected in 16-bit networks.

Network	#S1	#S2	#S	#SA0	#SA1	#SAFs	#CSSAFs
Octagon	274720	65024	339744	3840	3840	7680	347424
Spidergon	549440	130048	679488	7680	7680	15360	694848
5×5 Mesh	892840	203200	1096040	12480	12480	24960	1121000

A. Hardware Area Analysis: The proposed test algorithm is executed by a test module (TM) embedded in a router and its core of a node. The module is implemented and synthesized using Xilinx FPGA version 10.1 ISE Design Suite to estimate its area in terms of gate counts (GCs) needed for realizing these components. The Spartan3E FPGA family with the XC3S250E device, CP132 library package, and XST [VHDL/Verilog] tool of the ISE Design Suite are considered to synthesize the TPGs and TRAs, and implement the test setup. The setup is implemented with the set of channels of a subnet, TPGs (test source) at one side, say left and TRAs (test destination) at another side, say right of the channels. The RASoC [65], Xpipe [66] five I/O port routers are considered while calculating hardware area taken by a TPG and TRA i.e., a TM. Table 7.5 provides the synthesis results of a TM. The *core-TM* handles faults on the dedicated local channels only while the *router-TM* does so on both local and interswitch channels. Thus, the area needed by a core-TM is smaller than that of a router-TM. It is observed that a core-TM takes 16.54–18.3% area in RASoC and Xpipe routers while router-TM takes 23.46–25.97% area in these routers. A core is 4-5 times larger than a router. Therefore, this TM takes 4.14–4.58% area in the core. If one wants to analyze only shorts, the TM takes 2.99–3.31% of core-area and 16.35–18.10% of router-area. Likewise, the module takes 1.14–1.26% and 7.11–7.87% core and router area, respectively to analyze only SAFs. Note that the TM on a router is implemented on the assumption that the router

has five active I/O ports. However, each router on the basic octagon NoC has four active I/O ports to connect three neighbor routers and the dedicated core. Therefore, the area of the TM on four-port routers is seen to be 2–5% less for testing these channel faults.

Table 7.5: Synthesis results for a TM in a node.

n	TM Unit	#GCs	RASoC(%)	Xpipe(%)	Core(%)
	Core-TPG	163	9.66	10.69	2.42-2.67
	Core-TRA	116	6.88	7.61	1.72–1.9
	Core-TM	279	16.54	18.30	4.14–4.58
16	Router-TPG	222	13.15	14.56	–
	Router-TRA	174	10.31	11.41	–
	Router-TM	396	23.46	25.97	–

B. Test Time Analysis: While detecting a short fault on a subset of channels in a test iteration, a TPG needs five clocks to derive W1 sequences including header and trailer sets for a 16-bit channel, and one clock to organize these test flits into packets. The module transmits each packet with 1 flit/clock. Therefore, a receiver TRA over the channel-under-test receives the test packet by 4 clocks. On receiving the packet, the module analyzes the packet. Since test transmission of next packet and analysis of the previous packet are held simultaneously, a TRA effectively needs 2 clocks. Therefore, the test algorithm takes 40 clocks/iteration. In other words, the *Damaru*-based test configuration takes 80 clocks for detecting and diagnosing channel-shorts on a subnet. Hence, shorts in 16-bit channels of an octagon NoC can be tested in 320 clocks. If one wants to test SAFs in these channels, a test iteration is completed in 11 ($T_{tpg} = 2, T_{tpo} = 1, T_{tpt} = 6, T_{tra} = 2$) clocks and thus, one would need 88 clocks for the NoC. Hence, the octagon NoC will need 408 clocks to address CSSAFs for its 16-bit channels. This is applicable to other NoCs, such as spidergon, mesh networks and is seen in subsequent sections. However, one can save 5 clocks on every test iteration when the transmission of W1 packets is followed by A1, A0 packets. As a result, $T_{n/w}$ is reduced to $408 - 5 \times 8 = 368$ clocks.

C. Coverage Metrics Analysis: The fault injection and simulation experiment in a test round are performed in the sequence of testing DWs, CWs, and HWs of the channels of a subnet. Shorts followed by SA0s and SA1s are injected and analyzed on the channels during fault simulation. The fault simulation is accordingly conducted using the ModelSim simulator integrated with the Xilinx ISE Design Suite. Let us consider the first test round (TR=1) shown in Figure 7.5a where 16 channels of the first subnet in the octagon network are put in the test mode for shorts. During the first phase of fault simulation, possible shorts are assumed to be present in DWs. Then, #S1=25168, #S2=10224 i.e., #S=35392 shorts are detected by transmitting sufficiently long W1 test sequences and analyzing the respective responses.

In the second phase of fault simulation, shorts are assumed to be on CWs that extend the fault region to DWs. Additional W1 sequences are accommodated in control information ("bop, eop" signals). The fault simulation campaign detects #S1=55328, #S2=13944 i.e., #S=69272 shorts. Finally, the shorts on HWs extend the fault region to both DWs and CWs, i.e., all channel-wires in the subnet. Therefore, extra W1 set is placed in the handshake communication information ("val, ack" signals). The fault simulation detects #S1=109888, #S2=18240 i.e., #S=128128 shorts. In the current test round, the experiment detects shorts in 16 channels achieving 40% link (channel) coverage metric (LCM). All modeled faults in channels in the round are detected resulting 100% fault coverage metric (FCM). However, the FCM is 37.71% at the network level. These shorts are realized to 70.76%, 16.52%, and 12.72% as payload, misrouting, and dropping errors, respectively as shown in Table 7.6.

Table 7.6: Observation on the effect of channel-faults.

Network Type	Fault Type	Channel Errors		
		PE(%)	ME(%)	TE(%)
Octagon	Shorts	70.76	16.52	12.72
	SAFs	87.5	6.25	6.25
	CSSAFs	72.76%	14.52%	12.72%
Spidergon	Shorts	72.45	12.98	14.57
	SAFs	87.5	6.25	6.25
	CSSAFs	71.79	13.86	14.35
5×5 Mesh	Shorts	68.41	14.18	17.41
	SAFs	87.5	6.25	6.25
	CSSAFs	67.91	15.07	17.02

Now, suppose the channels in the subnet are under test for SAFs. When conducting the fault simulation in turn for SA0 and SA1 faults on the channels in above sequences, 1152, 192, 192 SA0 (SA1 in next turn) faults are detected in DWs, CWs, and HWs, respectively. The fault simulation achieves 75%, 12.5%, and 12.5% i.e., 100% as FCM resulting 87.5%, 6.25%, and 6.25% as payload, misrouting, and timeout error, respectively. It is therefore seen that the fault simulation achieves 100% FCM whence the shorts and SAFs are analyzed separately. Instead of simulating the shorts and SAFs separately, the SAFs are now injected at various positions before/after a short fault. When the faults are simulated in the above sequence of wire sets, the results indicate that 91.87% (Table 7.7) FCM is achieved while 8.13% faults remain undetected. The former one is categorized to diagnosable faults (DFs) while non-diagnosable faults (NDFs) are referred by the second category. Consequently, the fault simulation results in 72.76%, 14.52%, and 12.72% as payload, misrouting, and timeout error, respectively. Similar fault simulations in other test rounds are subsequently repeated

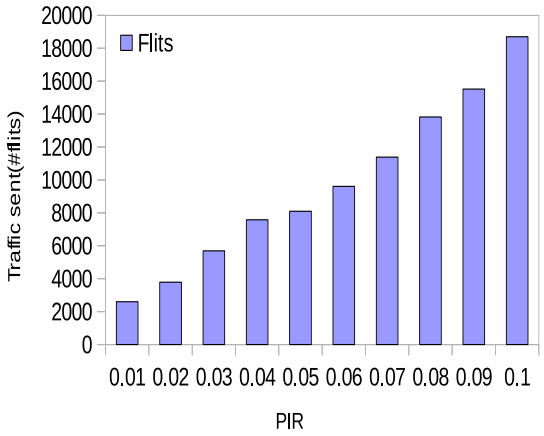
on the subnets as shown in Figures 7.5b–7.5d to detect and diagnose the shorts and SAFS separately as well as inclusively on rest of the channels in the octagon network.

Table 7.7: Observation on the deficiency in FCM.

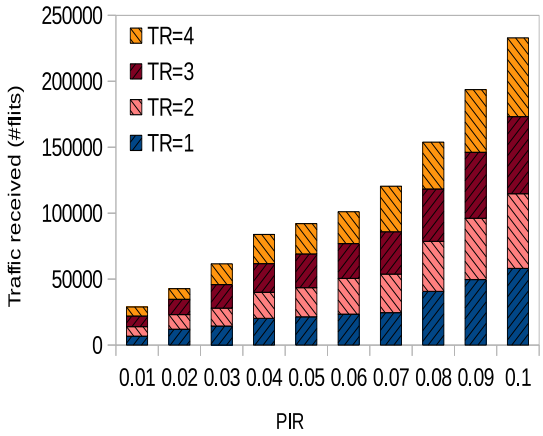
N/w Type	DF(%)	NDF(%)
Octagon	91.87	8.13
Spidergon	90.57	9.43
5x5 mesh	92.89	7.11

D. Network Performance Analysis: The CSSAFs have a strong impact on network performance metrics. If the faulty channel-wires are not handled by a fault-tolerant routing, a degraded performance is observed. Rigorous simulations are performed on the octagon NoC to evaluate the proposed solution and see the effect of these faults. The simulation environment is built-up with the cycle-accurate simulator Noxim that supports a set of topologies and allows users to create an NoC instance and compute performance evaluation [22]. Two basics steps: topology configuration and traffic generation, are done prior to evaluating the performance. Topology configuration is done by specifying the number of nodes and their interconnections. Further, the channel width is determined by assigning a flit size. In this work, 16-bit channels are considered which says each flit has 16 bits. For the simulation, the octagon NoC is modeled into a 2×4 mesh NoC with an interconnection between opposite diagonal router pair. In the traffic generation, synthetic traffic as the application data is generated using the inbuilt traffic model (e.g., uniform random spatial distribution) and the traffic size is determined by packet injection rate (PIR) value. The traffic is organized into small packets using the wormhole switching technique that enables to select the packet size in bytes. For example, four flits per packet and each flit of 16 bits are assumed, which means packet size=8 bytes. Here, application packets in terms of flits are injected at the PIR 0.01-0.1 and are made up of *W1,A1,A0* sequences. For each PIR value, the simulation is continued till 10000 cycles including 10% cycles to collect simulation statistics. The injection rate emulates the realistic nature of accepted traffic variations and is observed to affect the performance. Different performance metrics like latency, throughput, energy etc. are evaluated on transferring the generated traffic from a node to another. The XY-routing is preferred due to its deadlock-free property. The source and destinations are determined by the routing table. Further, the effect of faults on the metrics is induced by updating the routing at some intermediate node in a routing path. For example, the misrouting error is introduced by changing the header information of an incoming packet in the table.

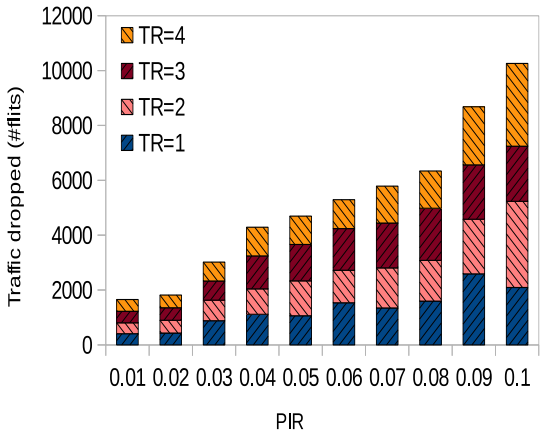
Figure 7.9 demonstrates the on-line evaluation of the proposed test solution. The network receives 2-4X of the injected traffic (flits). This happens because of packet duplication by shorts and more misrouting due to the CSSAFs. Note that many corrupt packets are



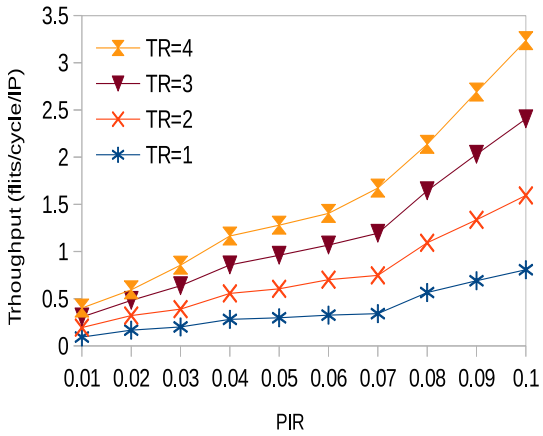
(a) Size of packet flits injected.



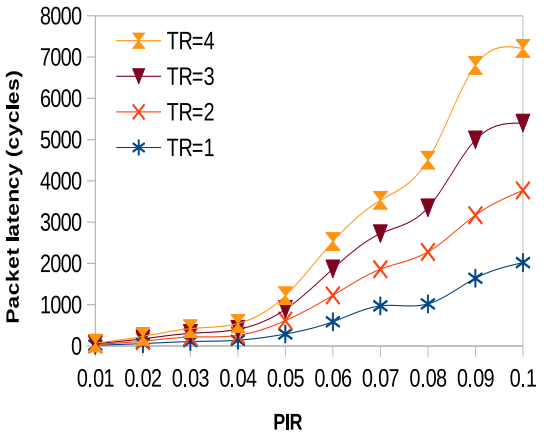
(b) Size of packet flits received.



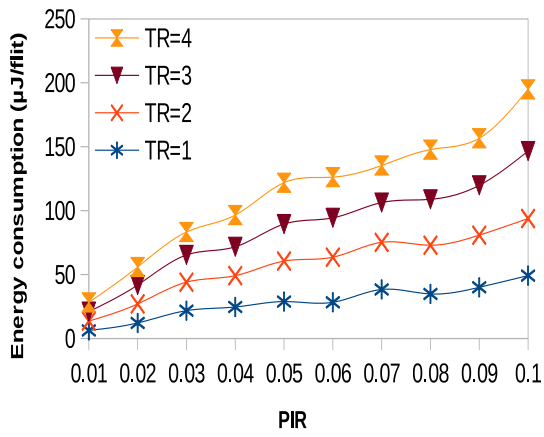
(c) Size of packet flits lost.



(d) Variation of throughput.



(e) Variation of packet latency.



(f) Variation of energy consumption.

Figure 7.9: On-line evaluation of *Damaru*-based test application to see the impact of CSSAFs in 16-bit channels of octagon network.

included in the received-traffic. A fraction of injected traffic is lost due to packet timeout fault. However, this fault is enhanced when CWs that carry packet trailers are affected by CSSAFs. Generally, 4–7% of injected-traffic is lost due to timeout. But, with faulty channels, this loss is increased to 12–16%. The scenario is different when injected-traffic contains only *W1* or *A1/A0* sequences. In the former case, the network receives up to 3X of injected traffic while timeout is observed up to 13%. In the latter case, the network receives 90% of injected traffic including ≈7–9% misrouted packets whereas packet timeout is observed up to 10%.

7.6 Solution Scalability

The implementation of an on-chip octagon architecture is simpler than any shared buses and traditional crossbars. Unlike a crossbar, the implementation complexity of the on-chip octagon network increases with its size in terms of interconnection of channels and nodes that the network must construct. As a result, the network strongly shows the ability to scale linearly with its size [87]. One strategy that implements a larger octagon network called κ-octagon is to connect κ basic octagon architectures with a common node shared by two adjacent architectures. Details on the layout of a larger octagon can be found in [8, 87]. Figure 7.10 shows the scaling strategy that constructs a 2-octagon network by $\kappa = 2$ basic octagon NoCs. The network shares a common node as labeled $\boldsymbol{B}$. The larger network constructed from $\aleph$ basic octagon architectures linearly reduces wiring cost $\Phi(\aleph)$ in terms of the number of nodes and local channels. For example, Figure 7.10 has one node and two local channels ($2 \times \ell c$) less as compared to two basic octagon networks. The cost reduction function grows as Equation 7.15. Additionally, the test cost function $\Psi(T_{n/w}, \aleph)$ is reduced as Equation 7.16. Application of the *Damaru* test configuration on the basic octagon architecture needs $T_{n/w}$ clocks. Therefore, $\Psi(T_{n/w}, \aleph)$ in Figure 7.10 doubles the $T_{n/w}$.

$$\Phi(\aleph) = (\aleph - 1) \times (2 \times \ell c + 1) \tag{7.15}$$

$$\Psi(T_{n/w}, \aleph) = \aleph \times T_{n/w} \tag{7.16}$$

Along with the set of desired properties, an octagon network has been designed for its nodes that can communicate at a maximum distance of two hops. As many $\aleph$ basic octagon architectures get connected to a larger network, any two nodes (each is in the different octagon architectures) in the composite network establish communication by (at most) $2\aleph$ hops including shared nodes. For example, a Brown color node establishes communication with a blue color node (Figure 7.10) via the red colored node $\boldsymbol{B}$. This shared node is known as a *bridge node*. As the name implies, adjacent octagons route packets through the bridge nodes.

The ever increasing performance demands, however, do not scale with the composite network size. The reason is the bridge node through which all packets from one octagon must

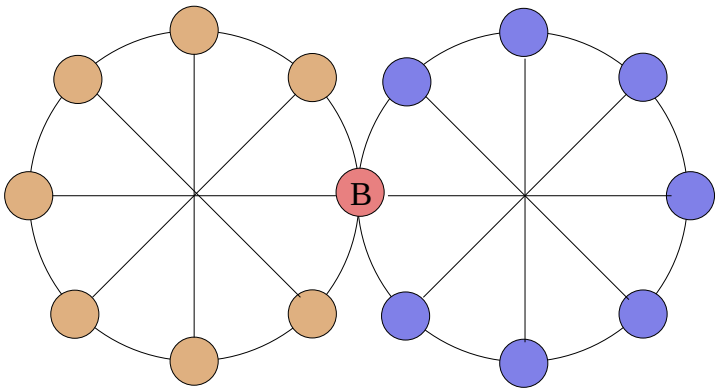
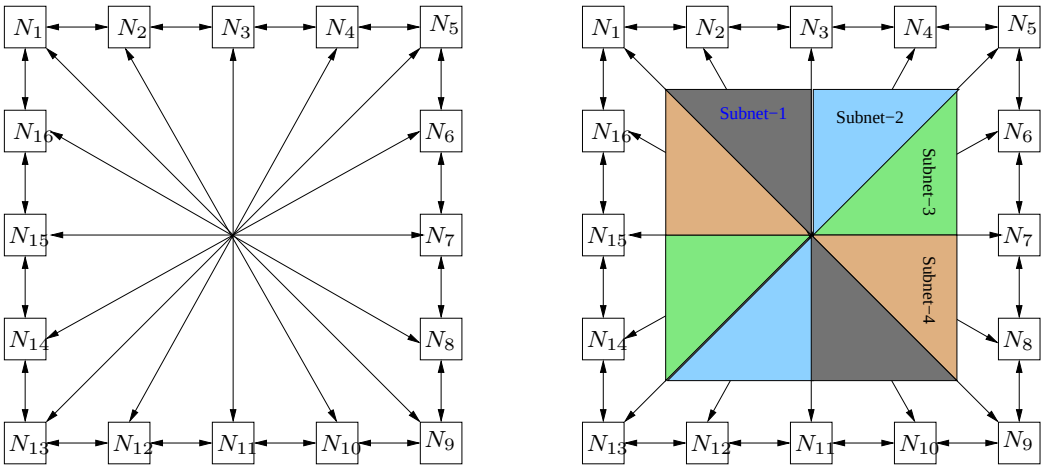


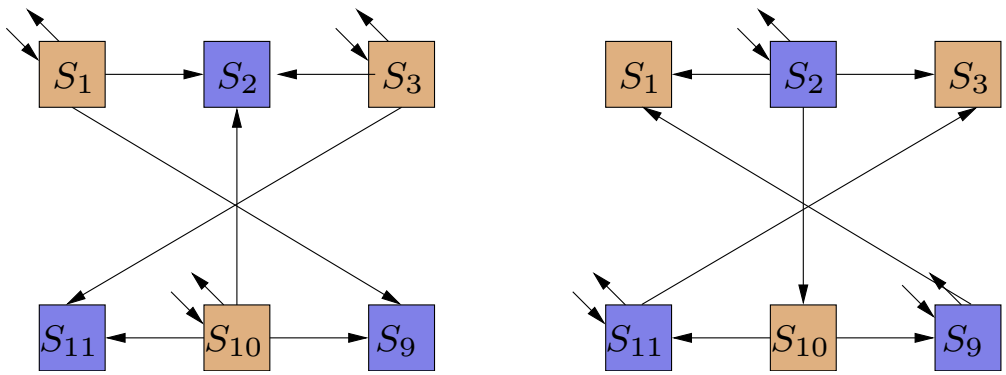
Figure 7.10: Connection of adjacent octagons to construct larger network.



(a) Abstract view of a spidergon network.

(b) Subnet selection in a spidergon network.

Figure 7.11: General representation of a spidergon network with 16 nodes followed by selection of test region in a round (subnet) in this network.



(a) First Iteration.

(b) Second Iteration.

Figure 7.12: Application of the proposed test mechanism on the first subnet (Subnet-1).

be forwarded to the adjacent octagon. For instance, the results are shown in Figure 7.9 that indicates lower performance, i.e., higher latency and energy consumption are observed when one runs the test method on 2-octagon network (Figure 7.10). As many bridge nodes appear in a routing path, the performance degradation is observed. For systems in which the demand for high-performance computation and communication is the dominant consideration, network designers have favored the *Spidergon* network [88, 89, 198] to an equivalent κ-octagon NoC, i.e., NoC shown in Figure 7.11a should replace one shown in Figure 7.10. The implementation layout, dealing with long wires in a larger Spidergon has been detailed in [8, 89].

The proposed test mechanism is now applied to the Spidergon network to see its natural scaling with the channel-CSSAFs provided in Table 7.4. The network is characterized by Table 7.3. The *Damaru* defines the minimum test configuration to detect the CSSAFs in channels of an octagon network. If the faults are required to be detected in the Spidergon's channels, the *Damaru* must be implemented on the network. Now, the proposed test scheduling divides an underlying network into four subnets as test regions. The Spidergon network (Figure 7.11a) can be segmented into four subnets as illustrated in Figure 7.11b. For instance, the Black colored region is treated as the Subnet-1 while the Indigo, Green, and Brown colored regions are the Subnet-2, Subnet-3, and Subnet-4, respectively. It is now noticed that the *Damaru* is expanded from a subnet of four nodes to six nodes interconnection. One can see (Theorem 7.1) that this expanded *Damaru* (*e-Damaru*) reduces test cost (time) overhead by 50% and more with the Spidergon size. Furthermore, it can also be seen (Theorem 7.2) that the *e-Damaru* makes the proposed test mechanism a time-independent solution on the larger Spidergon networks.

The first round (TR=I) test application on the Subnet-1 (Figure 7.12) is obtained by executing the test algorithm at odd and even nodes, in turn, i.e., two consecutive test iterations. Figure 7.12a shows the first iteration at (Brown colored) odd nodes while the next iteration at (Blue colored) even nodes is shown in Figure 7.12b. Another round (TR=II, TR=III, TR=IV) test applications are obtained by appropriately sliding the first test round (current *e-Damaru* configuration) clockwise/anti-clockwise on the respective subnets of the network (Figure 7.11b). The CSSAFs in the 16-bit channels are looked and each router has four port routers. Therefore, the silicon-area overhead of the TM unit remains same as discussed earlier. In a round, the TPGs as test sources at one side applies the test packets on the channels and the TRAs as test destinations on the other side analyze the responses to detect short, stuck-at faults. As before, the $T_{it} = 46$ clocks. Repeating the test rounds one can detect and diagnose the channel-CSSAFs on the Spidergon network at the cost of $T_{n/w} = 368$ clocks.

Theorem 7.1. *The e-Damaru improves by at least 50% test time compared to the basic Damaru on the Spidergon network.*

Proof. The test clocks needed to detect a channel-fault on an Octagon/Spidergon NoC depends on the subnet selection which specifies a test round. Basic *Damaru* configuration consists of the subnet of four nodes. Whence this configuration is applied to the Spidergon network, the number of test rounds becomes 8. On the other hand, the *e-Damaru* results in 4 rounds. Since the test clocks per iteration are same and every round is completed by two iterations, a Spidergon network can also be tested by same test clocks as of the octagon network. Thus, Equation 7.17 shows the improvement on test clocks $T_{n/w}$ needed by the *Damaru* and *e-Damaru* on the Spidergon network. □

$$\begin{aligned} Improvement(\%) &\geq \frac{\Psi(T_{n/w}, Damaru) - \Psi(T_{n/w}, e-Damaru)}{\Psi(T_{n/w}, Damaru)} \\ &= \frac{16 \times T_{it} - 8 \times T_{it}}{16 \times T_{it}}, [basic\ Spidergon] \\ &= 50 \end{aligned} \tag{7.17}$$

Theorem 7.2. *The e-Damaru presents a time-independent solution for Octagon as well as general Spidergon networks.*

Proof. A Spidergon is an extended Octagon and is constructed by interconnection of nodes of $\kappa \geq 2$ basic Octagons. For example, a basic Spidergon (Figure 7.11a) consists of 16 nodes selected from the basic Octagon (Figure 7.2a). The test scheduling divides an Octagon into four subnets, each results in the *Damaru* of four nodes. Without loss of generality, a subnet size as *e-Damaru* on a Spidergon must be larger than the *Damaru*. Equation 7.18 defines $\Upsilon(\kappa)$, the size of an *e-Damaru* which results in equal test rounds on both Octagon and Spidergon networks. Note that, *e-Damaru*=*Damaru* at $\kappa = 1$.

$$\Upsilon(\kappa) = \lceil \frac{8\kappa}{4} \rceil + 2 \tag{7.18}$$

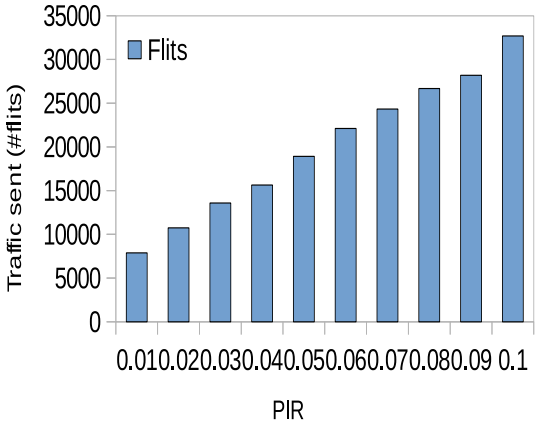
□

During application of the *e-Damaru* on the first subnet of the Spidergon network (Figure 7.11b), 26 channels are put in the test mode. These channels undergo the testing for shorts and SAFs in the sequence of DWs, CWs, and HWs. Let us first take the shorts through the channels. The fault simulation has detected #S=60242 (#S1=40898, #S2=19344) shorts on the DWs by transmitting sufficient W1 test sequences and analyzing responses. The simulations have subsequently detected #S=116284 (#S1=89908, #S2=26376) shorts on the CWs including the DWs, and #S=213064 (#S1=178568, #S2=34496) shorts on the HWs including the DWs and CWs. Thus, all channel-wires in the subnet have been tested resulting 100% (32.5% at the network level) as the LCM. At the same time, all modeled shorts have been detected and the fault simulation achieves 100% as the FCM. On the other hand, the shorts

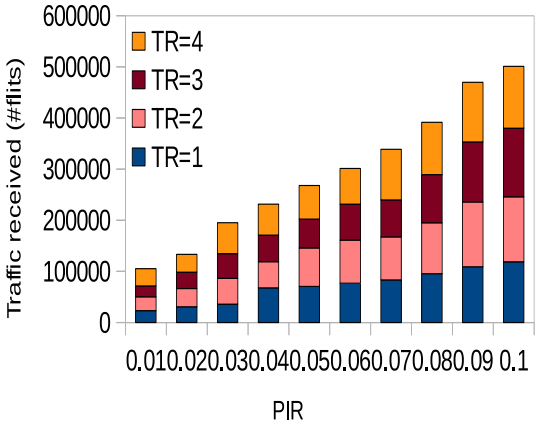
in the subnet have been reported to 72.45%, 12.98%, and 14.57% as payload, misrouting, and dropping errors, respectively (Table 7.6) which cumulatively ensure 100% FCM. Now, let us consider that the channels of the subnet suffer from the SAFs only. The simulation on the channels in the sequence of DWs, CWs, and HWs, have detected 1872, 312, 312 SA0 (SA1 in next turn) faults. As a result, 100% FCM achievement remains and sums up 75%, 12.5%, and 12.5% coverage due to SAFs on the DWs, CWs, and HWs, respectively. The simulation consequently reports 87.5% payload error due to SAFs on DWs and HWs. In the subsequent simulations of SAFs on CWs, 6.25% as the misrouting error and 6.25% as timeout error are observed. Now let us treat the coexistent shorts and stuck-at faults in the channels. Simulating the faults in the above sequence of wire sets, 90.57% of the injected CSSAFs have been recognized as the DFs while 9.43% of the faults remain undetected (Table 7.7). Consequently, the fault simulation results in 71.79%, 13.86%, and 14.35% (Table 7.6) as payload, misrouting, and timeout errors, respectively.

The performance bottleneck of a larger κ-octagon network has been overcome by an equivalent spidergon network, although the later network does not facilitate the wiring cost in terms of nodes and channels compared to the former network. However, the Spidergon network can be preferred as a high-speed communication architecture and placed in the series of high-speed network processors. Now, the well-known performance characteristics are investigated on the Spidergon network (Figure 7.11a) by simulation and treat the results as the severe effects of the channel-CSSAFs on the characteristics. Synthetic uniform random traffic as application packet flits in the range of PIR=0.01–0.1 and the XY routing protocol like before being used in the simulation. Note that each application packet contains the W1, A1, or A0 pattern. The Spidergon network without loss of generality is configured to a 2×8 mesh network with a channel shared by opposite diagonal routers.

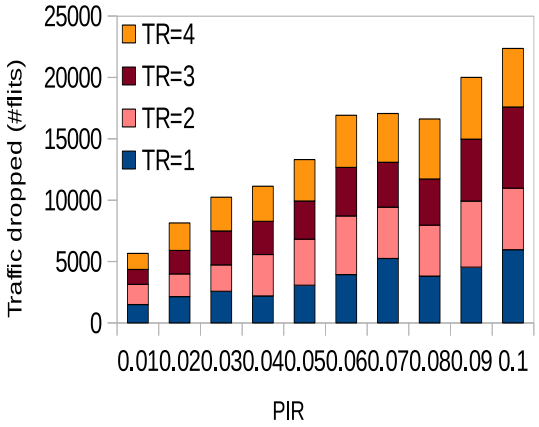
Figure 7.13 reveals the on-line evaluation of the proposed test solution on the 16-bit Spidergon network. The performance is seen in the orders of magnitude better than an equivalent network of multiple octagon networks. Figure 7.13a shows the amount of traffic injected on the Spidergon NoC (Figure 7.11) when a subnet as selected, is in the test mode. Correspondingly, Figures 7.13b, and 7.13c show the amount of received, and dropped traffic, respectively and are treated as traffic behavior in the system. The corresponding performance behavior that includes the performance characteristics (throughput, latency, and energy consumption) is shown in Figures 7.13d, 7.13e, and 7.13f, respectively. In the traffic behavior, the network receives 2.5–4.75X of the injected traffic due to different fault patterns in the channels. The received traffic contains the original, duplicated, and misrouted traffic. At the same time, a significant amount of traffic during transmission is lost due to dropping and timeout errors in the network. As seen, the amount of traffic lost on the average ranges from 13–19%. In the performance behavior figures, a load of received traffic as seen, directly affects the performance metrics. As the load approaches the limit of accepted traffic in the



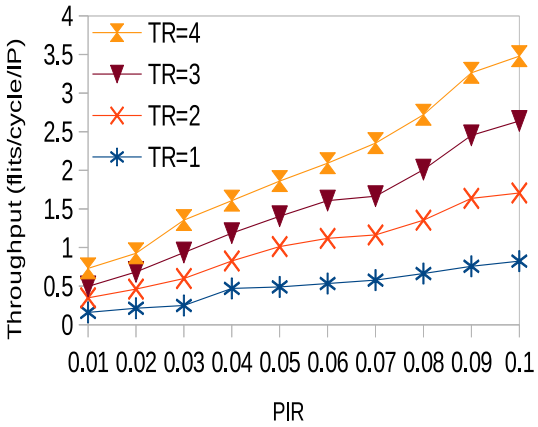
(a) Size of packet flits injected.



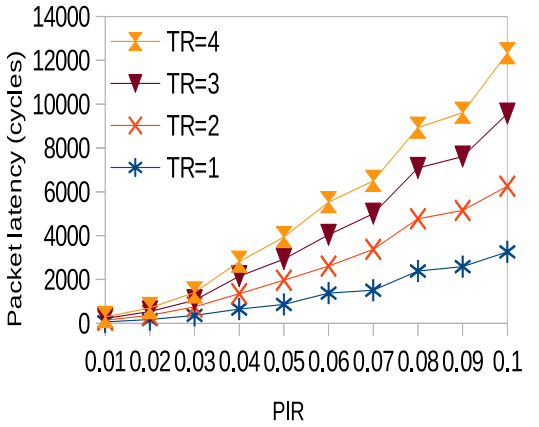
(b) Size of packet flits received.



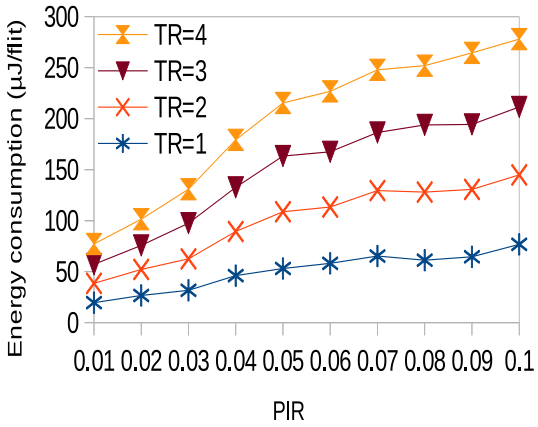
(c) Size of packet flits lost.



(d) Variation of throughput.



(e) Variation of packet latency.



(f) Variation of energy consumption.

Figure 7.13: On-line evaluation of *e-Damaru*-based test application to see the impact of the CSSAFs in 16-bit Spidergon network.

worst case, there will be high latency degradation as well as energy consumption in addition to more traffic contention. With the increasing received traffic, the latency may even increase exponentially. Similar to the nature of received traffic, the energy consumption saturates when the traffic accepted by the network reaches the throughput limit.

7.7 Solution Adaptability

Selecting a network topology is the most important step of NoC design as it deals with the wire length, the node degree, the routing strategies, and so on. The interconnection architectures having the smaller diameter, lower average distance, smaller node degree, and more number of channels are generally preferred [8]. The network topology that defines the organization of nodes and channels, is a key factor for test cost, performance, and fault tolerance [155]. It is known that maximum routing distance λ between two nodes on a basic octagon network is 2 hops. Note that this λ on the κ-octagon network with $\aleph = 8\kappa$ nodes equals to $2 \times \lceil \aleph/8 \rceil = 2 \times \kappa$. Unfortunately, the κ-octagon shows performance bottleneck due to the bridge nodes and subsequently is replaced by an equivalent spidergon network. The λ on a spidergon NoC of $\aleph$ nodes is $\lceil \aleph/4 \rceil$ [8]. Like the κ-octagon, equivalent spidergon may lead to increase in wiring cost for higher $\aleph$ i.e., large-sized spidergon. Additionally, the interswitch channel length between two routers in opposite direction e.g., nodes N_1, N_9 in Figure 7.11a, increases with $\aleph$ on the spidergon. Consequently, a little delay in receiving test packets may be incurred. As the effect, the overall test time $T_{n/w}$ is increased by few clocks and related performance overhead may be noted after certain spidergon size. The occurrence of channel faults followed by their detection at the runtime using a test method, thus, directly affects the test time and related performance overhead due to a topology selection. One can hence prefer an $A \times B$ mesh-based NoC architecture that has $\lambda = (A + B - 2)$ alongside other advantages, such as concurrent data transmission, simple floorplanning, layout regularity, and other topological characteristics and related controlled parameters [8, 191]. It is well known that most NoCs used in industrial NoCs (as Intel, Arteris) and literature are 2D mesh NoCs due to the fact that mesh NoCs possess many of the above-mentioned features. One clear advantage of the proposed test solution is that not only it scales with octagon and spidergon NoCs but can also be adapted to general topologies, such as $A \times B$ mesh.

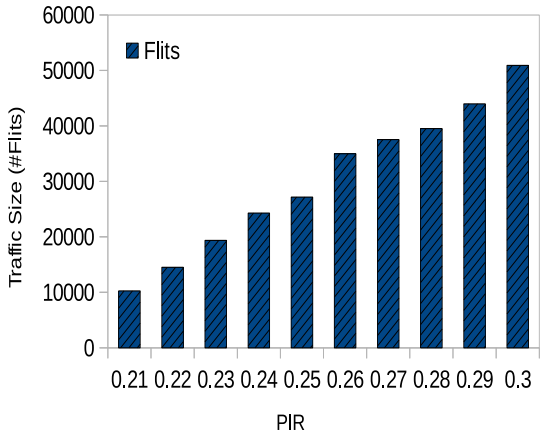
Theorem 7.3. *The n-bit networks, such as the octagon, spidergon, and mesh take same test clocks to detect the channel-CSSAFs.*

Proof. The test scheduling segments a network (Octagon, Spidergon) into four test regions, each as a test round. Every test round, on the other hand, is completed in exactly two iterations. For an *n-bit* channel, the test mechanism exercises $n + 2$ test flits (n W1 types, 1 A0 type and 1 A1 type), i.e., $n/2 + 2$ test packets. Analysis of these packets takes same

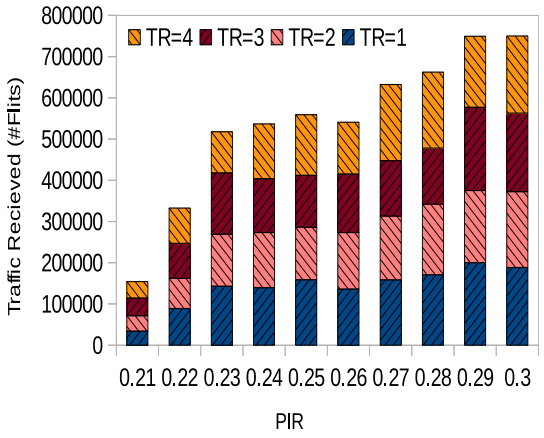
time on a test iteration. For example, consider channel width $n = 16$-bit on the networks. As established, every iteration takes $T_{it} = 46$ clocks for the CSSAFs. Thus, the four test rounds whether applied on an octagon, a spidergon, or a mesh network result to 368 clocks. □

The proposed solution is illustrated here with the 16-bit 5×5 mesh NoC characterized in Table 7.3. The network segmentation into four subnets, each as a test region is shown Figure 7.7. Applying test mechanism on a subnet is treated as a test round which is completed in just two iterations. The test application on the Subnet-1 as the first test round (TR=I) is shown in Figure 7.8. The test mechanism has targeted the CSSAFs in the 16-bit channels; the size of a TM unit as shown before remains unchanged. During test execution, the channels of a subnet are in the test mode. For example, 18 local and 24 interswitch i.e., 42 channels in the Subnet-1 (Figure 7.8) are considered under test. The fault injection campaign adds the shorts, SA0s, and SA1 on the channels. As discussed before, the test mechanism takes 368 clocks for addressing the channel-CSSAFs on the mesh NoC. It is evident that test time needed for all $A \times B(A, B \geq 3)$ mesh NoCs, each with n-bit channels, is same (Theorem 7.3). It makes the proposed *Damaru*-based test application as a time-independent test solution for both conventional and unconventional NoCs. The fault simulations as before are performed on the DWs, CWs, HWs of the channels of a subnet under test using the TPGs and TRAs at the ends of the channels. The area overhead of the TMs remains the same as provided in Table 7.5 due to same channel width and five physical ports to a router. The PTGs apply the test data and TRAs analyze the test responses during the fault simulation. Whence, the shorts are particularly diagnosed, the simulation over the test round detects #S1=288456, and #S2=63472 i.e., #S=351928 shorts and reports the channel-errors to 68.41%, 14.18%, and 17.41% as the payload, misrouting, and timeout errors, respectively. When the fault simulation is performed on the wire-sets with injected SA0 (or SA1) faults only, the simulation detects 3024 faults on the DWs, 504 faults on the CWs, and 504 faults on the HWs i.e., 4032 SA0 (or SA1) faults on the Subnet-1. As before, the payload, misrouting, and timeout errors are consequently reported as 87.5%, 6.25%, and 6.25%, respectively (Table 7.6). However, the fault simulation while diagnosing the CSSAFs on the channels in the subnet has detected 92.89% of the modeled faults. The rest of the faults amounting to 7.11% are treated as the non-diagnosable faults (Table 7.7). Correspondingly, the simulation results in 67.91%,15.07%, and 17.02% as payload, misrouting, and timeout errors, respectively.

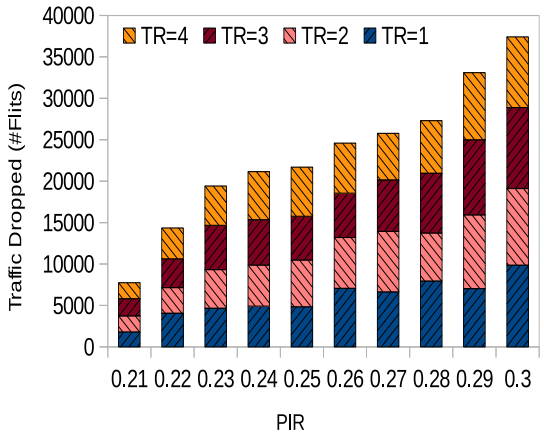
Now, the efficiency of the proposed solution is investigated on the 16-bit 5×5 mesh network. In order to assess the impact of channel's CSSAFs on network performance, the simulation setup is extended on the network. As before, uniform random traffic is considered to generate higher data volume that consists of W1, A1, A0 sequences. Accordingly, PIR=0.21–0.30 is set to inject the traffic on the network. The raw data is packeted and the XY routing is used to transmit the packet flits. Figure 7.14 illustrates the on-line evaluation of the proposed



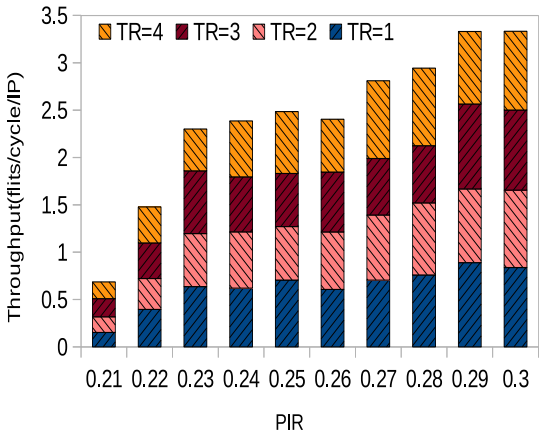
(a) Size of packet flits injected.



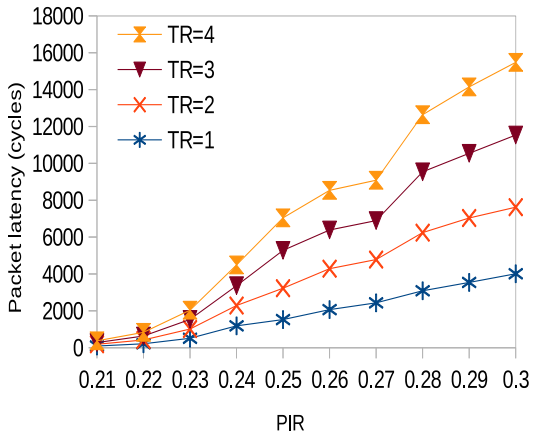
(b) Size of packet flits received.



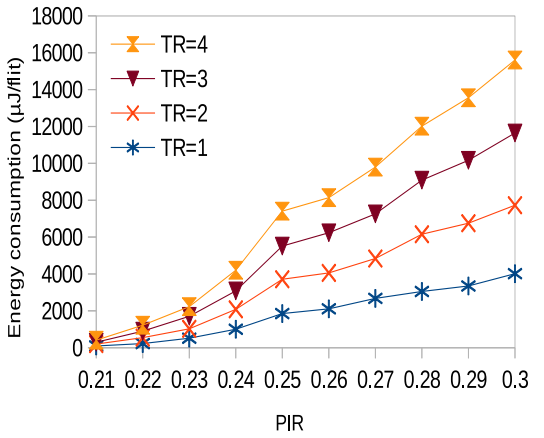
(c) Size of packet flits lost.



(d) Variation of throughput.



(e) Variation of packet latency.



(f) Variation of energy consumption.

Figure 7.14: On-line evaluation of *e-Damaru*-based test application to see the impact of the CSSAFs in 16-bit 5×5 network.

test mechanism on the 16-bit 5×5 mesh network that keeps a subnet under test mode. Figure 7.14a provides the size of synthetic traffic (packet flits) injected in the network. To examine the network behavior under different test rounds, the simulation is performed for every test round with this injected traffic. As the channel-CSSAFs are directly responsible for different failure modes, the injected traffic is consequently affected. Many flits are duplicated, corrupted, misrouted, and lost due to CSSAFs in the channels. Figures 7.14b and 7.14c, respectively show the amount of traffic received and lost in the network. It is observed that 3.25–5X and 13–19% of the injected traffic are received and lost in the network. Corresponding performance metrics namely throughput, packet latency, and the energy consumption by a packet flit, are shown in Figures 7.14d, 7.14e, and 7.14f, respectively. It is found that the traffic volume in addition to channel faults has a crucial role in the network performance. For example, the size of received traffic reaches the saturation point (the maximum traffic acceptance level by the network) rapidly when high volume traffic is allowed on a network with faulty channels. Beyond the limit, all injected traffic will be lost. Accordingly, packet latency, energy consumption are affected.

7.8 Benefits Gained

The proposed solution has the goal to provide a high fault diagnosis resolution while keeping the different overheads like area, test time, and performance overhead at a minimum. In addition, it targets to achieve high coverage metrics. The comparison study for the proposed on-line testing scheme is done with many prior schemes, such as 2×2 neighborhood selection model (2×2-M) [39], 2×1 neighborhood selection model (2×1-M) [41], self-diagnosis model (SD-M) [42], sequential router selection model (Seq-Model) [7], 4×4 subnet selection model (4×4-M)[RPC-12, RPC-14], nodes at two hop distance model (2hop-M) [RPC-13], diagonal node selection model (Diag-M)[Chapter 3], end-to-end model (E2E-M) [28], single channel selection model (SC-M) [44], and router-port based test model (Port-M) [199]. The simulation setup followed in previous sections is extended on these prior schemes to evaluate these overheads. Figure 7.15 shows the comparison of the models on the performance characteristics. The detailed comparison of above quality characteristics is conducted on the 16-bit octagon, spidergon, and mesh networks under similar simulation environment. The results achieved by the proposed solution compare favorably with the previous works.

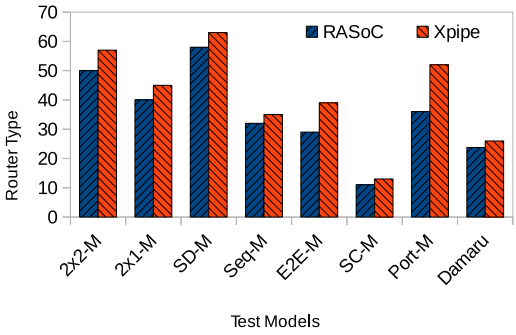
7.8.1 Test Area Benefits

The first comparison with state-of-the-art and current frameworks introduces the test area overhead incurred for a TM unit. In 2×2-M, TMs are placed at cores of a 2×2 neighborhood where test packets are analyzed on traversing four channel-length (hop) to address faults. Naturally, each TM takes larger space which is seen nearly 50–57% of the area in routers

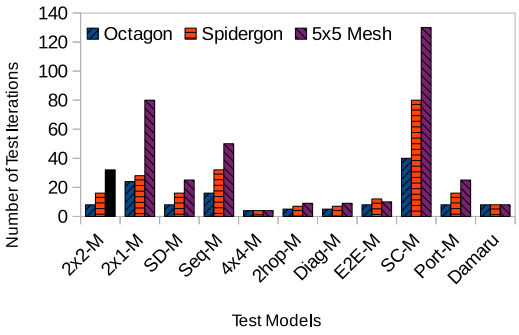
to address CSSAFs. Comparatively low area overhead is found in 2×1-M where each interswitch channel and its adjacent local channels are put in an iteration to address channel-faults. Therefore, test packets are analyzed after traversing three hops and TM takes 40–45% router-area. In SD-M, test data is analyzed on traveling two hops while considering faults in interswitch channels of the neighborhood of a node. Each TM needs nearly 12–16% and 41–44% router area in order to test stuck-at and CSSAFs, respectively. On consideration of the CSSAFs in local channels, the area overhead is increased to $\approx 58 - 63\%$, which is comparatively very high. Later, it is found in Seq-M that TM takes ≈25–27% area in routers while faults are assumed in interswitch channels only. If the faults are needed to be addressed in local channels, the overhead raises to 32–35%. Area overhead incurred by TM units in the E2E-M depends on the distance between two nodes. On a 5×5 mesh NoC, test data is transmitted in the X/Y direction from a leftmost (or top) boundary node to another rightmost (or bottom) boundary node. It means test data is analyzed on traveling four hops. In this case, the TM area is 14% to detect a transient fault on the interswitch channels that form a horizontal (or vertical) routing path. The diagnosed information is used to just observe the existence of a permanent fault. If the detection and diagnosis of the permanent faults are considered, this area overhead raises to ≈29–39% on the networks. In SC-M, TM units as a part of the monitor module placed at routers detect transient faults on the shared interswitch channel and are used to detect permanent faults on the channel. The area overhead is nearly 11–13%. Though it offers lowest area overhead but incurs substantially highest overheads on other parameters i.e., test time and related performance impact. In Port-M, test data is transmitted from a core to another neighbor core by traveling three hops which result in 9–13% area in the cores. In other words, TM takes 36–52% router area since a core is at least four times larger than a router. Figure 7.15a demonstrates a comparison of the test area overhead of the TMs incurred in the *Damaru* and the existing test schemes. As seen in this proposed scheme, a TM takes 17–26% area on the routers and has thus reduced area overhead that ranges from 5.25–37.03% resulting in 18.10–59.05% improvement.

7.8.2 Test Time Benefits

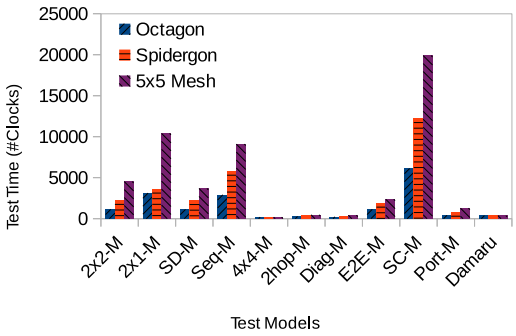
The second comparison introduces the test time required by the prior and current schemes on a network. This quality parameter depends on the number of test iterations and test clocks per iteration. The test iterations by the schemes are shown in Figure 7.15b on the basis of the manner that a test approach follows. For example, the 2×2 network neighborhood is selected in 2×2-M and is considered to be in the test mode at a test round. One can then easily understand the overall test time needed by the schemes. The parameter is evaluated by two techniques. The first way of computation is based on the time that actually the schemes take (Figure 7.15c). The second way of computation is based on executing the proposed test



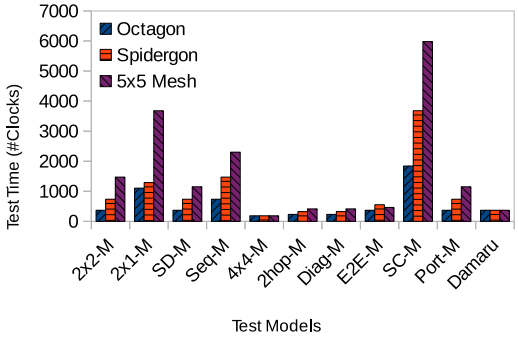
(a) Comparison on area overhead.



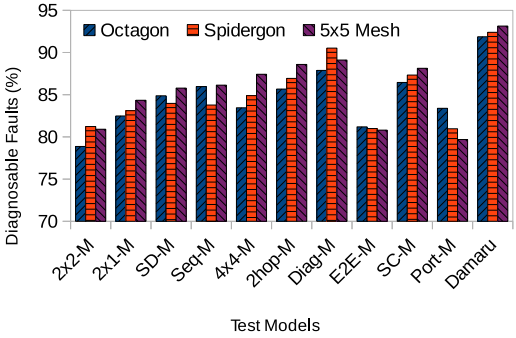
(b) Comparison on test iterations.



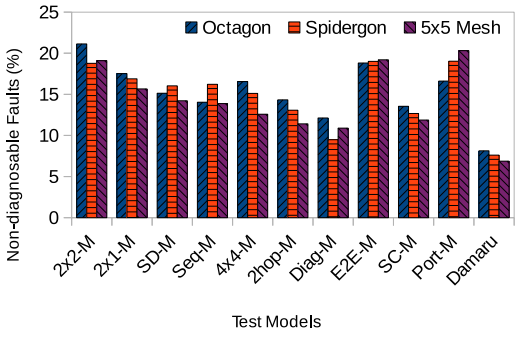
(c) Comparison on own test clocks.



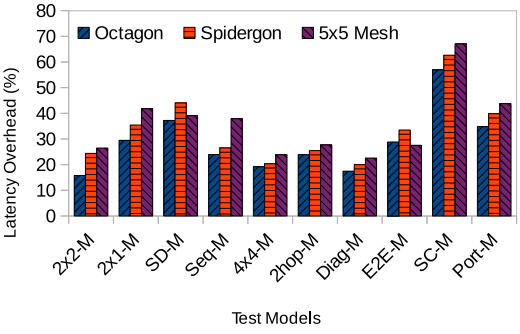
(d) Comparison on proposed test clocks.



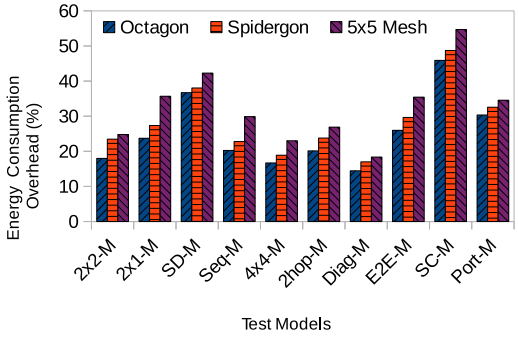
(e) Comparison on diagnosable faults.



(f) Comparison on non-diagnosable faults.



(g) Improvement on packet latency.



(h) Improvement on energy consumption.

Figure 7.15: Comparison study among the test models on various quality metrics.

algorithm on these schemes (Figure 7.15d). For instance, one may assume that the 2×2 neighborhood selected in 2×2-M is equivalent to a subnet of the octagon NoC as decided by *Damaru*. Although the channels can be tested in 4 rounds on the network each iteration needs 282 clocks by the 2×2-M to address the CSSAFs resulting $T_{n/w} = 1128$ clocks. On the contrary, these 4 rounds are completed by 8 iterations using the *Damaru* that takes $T_{n/w} = 368$ clocks only. The result shows that current scheme lowers test time overhead by (1128-3688)*100/1128≈67.38% and becomes $1128/368 \geq 3\times$ faster on the octagon network. The above mentioned two techniques of analysis can be carried out with other existing test schemes on octagon network as well as spidergon and mesh networks. The proposed scheme delivers 9.91–98.15% and 20.00–93.85% reduction on the test time with respect to the first and second technique of computation, respectively. Subsequently, such achievement makes the proposed solution to be $\leq 55\times$ faster than the existing works on these (octagon, spidergon, mesh) networks. The test schemes namely 4×4-M, 2hop-M, and Diag-M on the octagon and spidergon, and the former scheme (4×4-M) on the 5×5 mesh networks show lowest test time as compared to the proposed *Damaru*. The main reason is that network sizes are small. With the increasing network size, the number of test iterations by these test schemes becomes higher resulting in high test time. On the contrary, the current scheme conveys fixed test time irrespective of the network size and type making it more convenient for large-scale networks.

7.8.3 Fault Coverage Benefits

Figures 7.15e, 7.15f compare diagnosable and non-diagnosable faults detected by *Damaru* and prior test schemes. When shorts and SAFs are analyzed independently, most of the prior test schemes, for instance, Seq-M including the proposed model results in the FCM to be 100%. Few schemes, however, fail to meet the level. For example, 2×1-M, SD-M show 94.4% and 96.5% as the FCM, respectively while only shorts and only stuck-at faults are considered in the analysis of the models. It should be intuitive that coexistent nature of the faults in channels makes a fraction of the faults to be non-detectable by a test method. As it can be seen, many short and stuck-at faults in 2×2-M cannot be detected due to multihop transmission of test packets. The scheme results in 78.87% as the diagnosable faults while 21.13% of the modeled faults are nondiagnosable on octagon network. If 2×1-M is applied to the network, it detects 82.48% faults while 17.52% are undetected. Similarly, the fault simulation is conducted for rest of the test models to observe their diagnosis capability, and accuracy to reach the quantity of detected and undetected faults. In the end, the proposed model reaches to 91.87% and 8.13% as diagnosable and non-diagnosable faults. Thus, the proposed model in comparison to rest of the other existing models reaches 3.99–13% above the diagnosis level in terms of FCM achieved. In other words, the fault non-diagnosability is reduced from 13–3.99% on the octagon network. This trend on the spidergon and mesh networks is observable to 11.16–1.87% and 13.44–4.02%, respectively when all the test methods under comparison are executed.

7.8.4 Performance Benefits

The final comparison is on the performance overhead in terms of two metrics, packet latency, and energy consumption, incurred by various test approaches. Figures 7.15g, 7.15h compare the additional performance overhead of the prior methods with reference to the proposed scheme. The on-line evaluation is done on the octagon, spidergon, and mesh networks under similar simulation environment. These metrics are greatly dependent on some key considerations related to a test technique. The number of test iterations, test time per iteration, fault type, traffic size and type, test application mode, are among those considerations. In fact, these criteria have the great impact on the overall performance behavior of an NoC. For instance, the requirement to enter multiple test regions renders an application packet to not only experience higher latency but also consume additional energy to reach its destination. It can be observed that maximum performance degradation is attained by SC-M during the on-line evaluation on the octagon, spidergon, and mesh networks. Note that this degradation as latency overhead shows 56.94%, 62.68%, 67.05% in these networks, respectively. Likewise, this degradation as energy consumption overhead in these networks is 45.88%, 48.71%, and 54.69%. Sequentially continuing the evaluation on the networks with rest of the prior test schemes, it is seen that Diag-M presents lowest performance degradation in the range 17.45–22.57% for latency overhead and 14.44–18.35% for energy consumption overhead. Thus, the prior schemes exhibit performance overhead that corresponds to 15.75–67.05% on average packet latency and 14.44–54.69% on average energy consumption by a packet flit. Considering the evaluation scenarios, it can be seen in the performance degradation list that the prior approaches: SC-M, SD-M, and Port-M occupy the top three positions while the prior approaches: 4x4-M, 2hop-M, Diag-M have top three positions from the bottom. Despite the lower hardware area overhead, acceptable fault coverage, and comparatively low-performance overhead characteristics, the later mentioned three techniques are unfortunately uneconomic for the large-scale NoCs because these systems always incur long test time by these techniques. It is important to note that inclusion of the proposed test solution on the NoCs saves a suggestive amount of performance overhead together with the gains on the test area, test time, and fault coverage metrics.

7.9 Conclusion

This work discusses a cost-effective, time-optimized on-line test solution that can be used to address the co-existent short and stuck-at faults in channels of NoCs in general. The principle of the proposed on-line fault testing for NoC-channels lies in devising a suitable test scheduling scheme that improves test time, resource utilization, and favors scalability. In addition, a key aspect of the proposed scheme is to present a fixed test time for all kind of NoCs. The proposed scheduling approach contributes to the plausibility of exploring an iterative test application

on the NoCs in order to meet this aspect. The efficiency of this scheme is evaluated under the framework of different influential quality characteristics. Comparative experimental study on the framework is accomplished among a set of prior test strategies including the current one. The results of this assessment reveal that the proposed solution offers several benefits over existing solutions. The benefits include in terms of saving the TM area by 37.03% or improving hardware area overhead by 59.05%, saving test time by 98.15% or making the scheme faster by 55×, achieving the fault coverage over 93%, and reducing the performance overhead by 67.05% for packet latency and 54.69% for packet energy consumption. A distinct advantage of this work over many existing approaches is the possibility of implementing fault detection and diagnosis at the runtime of regular application on a general NoC. The current scheme thus provides a favorable alternative for enhancing the reliability and related issues of NoC-based communication architectures.

Chapter 8

Summary and Future Works

8.1 Summary of Contributions

This dissertation has focused on the testing of manufacturing and transient channel faults in NoCs. The channels in terms of single or multiple fault models have been examined during the testing. In the former case, one manufacturing fault, such as only stuck-at, only open, or only short fault has been taken in the channels. In the latter case, coexistent nature of faults, e.g., stuck-at and short faults, has been accounted in the channels. Having addressed the shortcomings of the prior works, the contributions in the dissertation are summarized as follows.

In the first contributed work (Chapter 3), the proposed test mechanism has been dedicated to addressing stuck-at faults in NoC channels. The test architectures designed at both cores and routers handle two test packets, one contains All-One vector for exploring stuck-at-0 faults, another contains All-Zero vector for exploring stuck-at-1 faults in the channels. Multiple nodes that execute an instance of the test algorithm have been selected by either of the two proposed scheduling schemes. The first scheduling scheme has been presented for $M \times N$ networks and scheduled the nodes whether located on the odd/even diagonal level. The second scheduling scheme has been shown to be more general and scheduled the nodes by mapping the node selection problem to the well-known graph coloring problem. Various quality metrics namely hardware area, test time, link and fault coverage metrics, and performance metrics – throughput, packet latency, and energy consumption, have been evaluated to look at the efficiency of the proposed test solution on a set of network architectures. The evaluation has shown that the proposed solution not only scales with network size and channel-width but also scales with all general topologies, such as an octagon. Detailed comparisons between the proposed solution and a set of existing methods have been studied, that shows the proposed scheme has outperformed the prior schemes. It has been shown that the silicon area overhead is reduced by 23.84–78.01% while the proposed scheduling

policy makes the solution $4-16\times$ faster resulting up to 94.57% improvement in test time. Furthermore, the performance overhead has also been significantly reduced. For instance, packet latency has been reduced by 13.21–40.40% while 10.90–46.52% reduction in energy consumption has been perceived. Such gains in terms of quality metrics may be noticed with NoC size, channel width, and topology.

In the second contributed work (Chapter 4), a test mechanism has been proposed for detecting the maximal connectivity in NoCs. The connectivity detection has been carried out in terms of the testing of open faults in channels of the NoCs and focused on preventing the system level failure modes namely partial and full packet loss. This work specifically has completed the bridge of testing manufacturing faults in NoC channels. The test time per iteration has been lowered by considering test architectures at routers and cores, and unicast and multicast routing of a test packet. Note that exercising only single test packet at router/core, the test algorithm is able to explore open faults in the channels. In addition to detecting channel's open faults, a new test scheduling has been proposed in order to lower the overall test time and improve associated performance overhead. A hierarchical test scheduling scheme driven by 4-Corner principle has been discussed. Experimental results have reported that the implementation of the test mechanism incurs small hardware area overhead in a node and shows fewer clocks as the test time for addressing the channel-open faults. The on-line evaluation of the proposed test solution has demonstrated the effect of channel-open faults on NoC performance. Further, application of the proposed test solution on several meshes and an octagon network, has ensured its scalability which is not limited to network size and channel width but also is extended with respect to the network type. In comparison to prior works on a selected set of networks, the present test scheme reduces test area overhead by 35.36–67.73%. At the same time, it reduces the overall test time by 96.43% and becomes $28\times$ faster. Further, the test execution has lowered the performance overhead to an acceptable level. For example, the packet latency is reduced by about 5.83–42.79% while the energy consumption is lowered by about 6.24–46.38%.

In the third contributed work (Chapter 5), a reliability-aware and topology-agnostic test scheme has been presented for on-line testing of short faults in NoC channels. The proposed algorithm detects both intra- and inter-channel short faults and identifies the faulty channel-wires at a node. Efficient test architectures have been designed at cores and routers for detecting and diagnosing interconnect shorts on exercising minimum walking-one test sequences. The test time per iteration has been lowered with single hop unicast and multicast mode of transmission of walking-one test data. A cluster-based distributed test scheduling has been proposed to reduce the overall test time needed by the test algorithm for channel-shorts in the NoC. During a test application, the nodes in a cluster set are appropriately scheduled to concurrently execute the test algorithm. One great advantage of this cluster-based, distributed scheme includes that it does not only reduce the overall test time for the

NoC but also suggests same test time for larger NoCs irrespective of size and topology without increasing any hardware cost constraints. Subsequently, the proposed approach scales to larger NoCs irrespective of size, channel width, and type of the networks. Compared to prior works, it has reduced hardware area overhead up to 27% and test time by more than 21× on several test cases. Additionally, the packet latency and energy consumption have been observed to be reduced by 19.47–40.16% and 17.57–34.20%, respectively.

In the fourth contributed work (Chapter 6), the issue of network resource utilization in terms of test energy dissipated during testing of channel faults has been estimated. The contribution also includes the testing of permanent (short) and transient faults in channels of NoCs. Also, a new test scheduling scheme has been proposed, which accompanies a trade-off between the test rounds (or iterations) and resource utilization. The new scheduling technique partitions an NoC architecture into four subnets providing lowest and fixed test time-based solution for NoCs irrespective of their size and type. The proposed solution has improved various quality metrics on the set of NoCs, for instance, the test area overhead is reduced up to 28% while it has reduced test clocks up to 11.11–93.75% over the existing models on a set of $P \times Q$ networks and thus becomes 16× faster. Also, the improvement in the performance overhead is convincing. The packet latency, for instance, is improved by 14.98–50.67% while packet flit energy consumption is reduced by 6.83–43.89%.

In the fifth contributed work (Chapter 7), a performance-aware, cost-effective and general test mechanism that is capable of diagnosing co-existent short and stuck-at channel-faults in NoCs, has been proposed. The proposed mechanism has also addressed the issue of fault non-diagnosability. More preference has been given to unconventional NoCs, octagon, κ-octagon, and spidergon networks which are similarly able to meet the demand of high-performance requirements of today's multicore, multiprocessor architectures. Nonetheless, the proposed test mechanism has been discussed with traditional mesh architectures. A modified partition-based scheduling technique named *Damaru* has been presented to provide similar network resource utilization on a test round and constant test time on the NoCs irrespective of size, and topology. Simulation results have shown that the proposed test method can achieve nearly 92% fault coverage, and improve area overhead by almost 60%, and test-time by 98% compared to earlier approaches. As a sequel, packet latency and energy consumption are also improved by 67.05% and 54.69%, respectively.

8.2 Future Works

The future works of the proposed solutions can thus be placed in diverse ways that will present numerous opportunities for additional research on the NoCs.

- **Testing of Coexistent Stuck-at, Open, and Short Faults in Channels**
  With the ever-shrinking global geometries on a die and the concomitant rise in the complexity of interconnections in an NoC, the coexistent nature of manufacturing defects in channels used therein often consists of a pair of faults, such as stuck-at and open, short and open, short and stuck-at faults. Also, these faults can be experienced together instead of occurring in pairs in the channels. Consequently, the combined effects of the defects not only cause logical or functional errors but also give rise to various other system level failures such as corruption, duplication, misrouting, or dropping of a packet, etc., thereby impacting the performance of the network significantly. A diagonal node selection based test solution is extended in [RPC-9, RPC-17], a central, border, and corner node selection based method [RPC-16], a matrix-based hierarchical test scheme [RPC-15], and a 4×4 subnet selection method [RPC-12] are discussed to analyze the coexistent stuck-at, open, and short faults in the channels of NoCs. The test solutions described in Chapter 4–7 can be extended for accounting contemporary manufacturing short, stuck-at, and open faults in NoC channels. In particular, the partition-based test solution discussed in Chapter 6 and 7, furnishes a balanced trade-off between the test time and network resource utilization. One can continue research on coexistent channel-faults with this approach.

- **Router Testing**
  The channels in on-chip networks are treated as the communication highway that transports application data in packets using a routing strategy. Similar to a communication channel, a router is a fundamental block that constitutes the backbone of the networks and responsible for routing packets from source to destination in the networks. A router consists of a set of combinational and sequential hardware elements, such as routing logic blocks, first-in-first-out buffers, switch allocator, arbiter, multiplexers, etc. Most of these elements as seen in the literature are vulnerable to stuck-at faults. Accordingly, many test techniques in the literature are discussed for the fault. These techniques may include scan-based testing, functional test, and built-in-self-test. The fault models used in these techniques may differ by abstraction levels (e.g., register level, logic level) or covered blocks (e.g., registers, buffers). In order to reduce NoC redesign costs, one should prefer the functional testing. On the contrary, test application time, fault coverage may be enhanced whence a scan-based testing, the built-in-self-test approach is considered. As such, the techniques may thus raise the basic test issues like test time, fault coverage, hardware area, and so on. In particular, the test time that enhances performance degradation in terms of latency, power consumption is more critical at the runtime of the NoCs. One main reason may be imagined to be the selection of the routers that are put in the test mode. Performance improvement by latency optimization, power

optimization can be made properly through a router selection method. Another possible extension to the contributions presented in this dissertation can be focused on the intelligent scheduling of routers under test in a test iteration or round.

- **Core Testing**
  The IP cores in an NoC architecture is the primary means of message generation and processing center. The unit is another basic building block in the NoCs. The testing of cores is more challenging as compared to that of routers and communication channels. This is because of the large-scale integration into the system design. Another reason is the availability of fault-free interconnection of routers and channels between the source and sink to transport test data for a core under test. As found in the literature that the cores undergo the testing by employing a test access mechanism (TAM) that establishes communication between test source and test destination for a core under test. Design of an efficient TAM always searches for a better trade-off between the capacity of the TAM to transport test data from source to sink and the TAM's application cost. One or multiple cores between the source and sink may be placed in the testing mode. Consequently, the TAM's capacity for transportation of test data is determined by source's as well as sink's capacity. Also, hardware area used in the system for the TAM known as its bitwidth, and routing distance between source and sink are other factors that can limit TAM's capacity. A number of TAM-based test approaches are discussed in the literature. The author of this thesis encourages to consider the test scheduling schemes presented in Chapter 3–7 to those researchers who are willing to get involved in core testing.

- **Correcting Channel Errors**
  The work in the dissertation is mainly dedicated to the testing of manufacturing faults in NoC channels with a progressively improved test solution. Besides, the channels can suffer from the temporary or transient faults caused due to signal integrity, aging, etc., issues. The dissertation also presents a test method for the detection of transient faults. These faults are recoverable or can be corrected. When such faults exist in channels, designers normally prefer to use an error correcting code (ECC) based control scheme, data retransmission using automatic repeat request (ARQ) scheme, or a hybrid of these schemes for dealing with a data packet infected by a transient fault. The problem in using these techniques is the test cost, for instance, hardware area, energy consumption, and network congestion. The designer then needs to find a good trade-off between the test cost and the potential benefit of an underlying test scheme. In Chapter 6, a test mechanism for the detection of transient faults has been provided without any ECC or

ARQ scheme. Another possible extension of the proposed work in this dissertation may be directed to error control coding and data retransmission to overwhelm channel errors due to transient faults in terms of reliability.

- **Testing of Other Channel Faults**

  Due to the effect of deep submicron (DSM) technology, delay and noise faults [159, 200] may occur in the NoC-based interconnections. As a remedy, several test patterns are consecutively applied with a specific timing in order to detect such faults and achieve the desired system's behavior. It should be noted that the order of the delivery of selected test patterns is important for the testing of these faults [201]. The work in this thesis states that every test algorithm routes the test packets to a single hop which means that the proposed test algorithm neither applies test patterns in arbitrary order from the sender nor a receiver receives them arbitrarily. Therefore, the algorithm may be suitable to ensure that the intended order of delivery of test patterns is maintained. Subsequently, the test approaches presented in the thesis may be brought into the detection of delay and noise faults in the NoC interconnections.

- **Research on 3D NoCs**

  With the increasing number of IP cores, a 2D NoC may not be suitable for efficient communication due to long routing distance in the NoC with large size. The 3D NoCs is one of the upcoming NoC architectures where NoC layers grow vertically to reduce communication overhead. Multiple layers of 2D NoCs are stacked above each other and vertically interconnected using through-silicon vias (TSVs) [202]. By stacking active silicon layers, such integration mitigates the problem of interconnect wire delay. Although 3D NoCs offer several advantages over a 2D NoC, one of the major bottlenecks for 3D NoCs is post-manufacturing testing of different components such as cores and channels for different manufacturing and transient faults. While the cores perform computation, defect-free routers and channels in a routing path are needed for reliable data exchange between sender and receiver. Otherwise, faulty components in a 3D NoC will cause different system-level failures that may have a significant impact on its performance. Therefore, one can, for example, employ a test solution discussed in this thesis for detecting a fault in the channels. The channels of a layer or a subset of channels from each layer can be put in the test mode at the runtime. In the former case, the test schedule proposed in Chapter 3 can be suitable. All the nodes which are at the odd diagonal levels and rest of the nodes which are at the even diagonal levels in a layer can execute a test algorithm in alternative test iterations to detect channels faults in the layer. Other scheduling schemes discussed in Chapter 4–7 may be welcome in the latter case where a subset of nodes from all layers on the basis of a scheduling method can be allowed to run the test algorithm for the channels of these nodes.

- **Research on WiNoCs**

  Recent research on silicon integrated antenna has established that such antenna can operate in the millimeter (mm)-wave range from 10–100 gigahertz (GHz) and be accepted as a viable technology for intra-chip as well as inter-chip communication [203, 204]. In recent observations, excellent emission and absorption properties additionally lead to antenna-like behavior in carbon nanotubes (CNTs) which can operate at optical frequencies [205]. The 3D NoCs are the proposed on-chip communication architectures to replace 2D NoCs. Subsequently, researchers at 3D NoCs have evolved globally to address the issues of 2D NoCs. So far the investigations, researchers have accepted the fact that the issues of performance degradation due to high latency, power consumption remain active in 3D NoCs despite their advantages. In 3D NoCs, the high latency is because of multihop communication between sender and receiver whereas the significant power consumption is because of the metallic communication channels. Some efforts are nowadays being put to address this problem by introducing a new NoC architecture with ultra-low-latency and low power express wireless channels. As a result, WiNoCs are coming in next generation NoC-based communications [206, 207]. In a WiNoC, every node is supported with a transceiver that exchanges message signals with a CNT-based antenna. Such implementation reduces the multihop communication between highly separated nodes to a single hop due to the wireless channels. Although distant communications are established with wireless channels, local communications are however done using traditional metallic channels. Even if a local communication is made using a wireless channel in order to reduce power and delay compared to its conventional counterparts, there are still metal control wires. Consequently, reliability issues due to faults in these metallic channels will be arising out of adopting a WiNoC in addition to its other promises and challenges [206]. It would be quite innovative to apply the approaches presented in this thesis for the detection of manufacturing and transient faults in metallic part of the communication channels in the upcoming WiNoCs. Such an investigation provides another possibility to the future direction of the work in the thesis.

- **Research on PhNoCs**

  With recent advances in silicon nanophotonics, PhNoC architectures are being explored in order to replace the traditional 2D NoCs and to support higher bandwidth and lower power dissipation in future CMP communications [176]. A PhNoC which is in progress to be taken in the set of upcoming on-chip networks, consists of two planes: lower plane is the metallic layer built upon the conventional NoC while the upper plane with the same topology, is the optical layer built upon the interconnection of optical switches and optical (wave guided) interconnects. For performing network management and

distributed control functions, short message exchange, etc., the lower electronic plane is used. On the contrary, high-bandwidth multiwavelength dedicated to bulk message transmission is accomplished by the upper optical plane [8, 208]. In addition to the design and implementation issues for the energy-efficient PhNoCs, the reliability will be a major concern due to the faults in the channels, routers, cores of the electronic plane. Thus, another research direction of the works in the thesis can be extended to the testing of these components, especially communication channels for reliable message exchange via the electronic plane of the photonic NoCs.

8.3 Concluding Remarks

In accordance with the current status in terms of several limitations of the prior test approaches, this thesis presents the efficient on-line test solutions. These solutions are dedicated to addressing the logic level faults in the communication channels of NoCs in general. The included logic level faults in the channels are primarily manufacturing faults (stuck-at, open, and short), and secondarily transient and other (e.g., packet deadlock) faults. These faults in the channels are analyzed both at the single as well as multiple fault models. The experimental results reveal that the proposed test schemes successfully overcome the issues of the existing test methods. In particular, the proposed test schemes lower the test-time and test area overhead significantly and consequently become an alternate attempt for testing the channel-faults in general large-scale NoCs. The proposed test schemes with the concluding remarks have the special characteristics that they are not design-specific, can be easily integrated into the NoCs, and should be seeking for faster solutions in order to reduce the test costs, improve the flexibility, and increase the system performance.

Bibliography

[1] Wang, *System-on-Chip Test Architectures: Nanometer Design for Testability*. Morgan Kaufmann Publishers Inc., 2008.

[2] F. Moraes, N. Calazans, A. Mello, L. Möller, and L. Ost, "Hermes: an infrastructure for low area overhead packet-switching networks on chip," *Integration, the VLSI Journal*, vol. 38, no. 1, pp. 69 – 93, 2004.

[3] C. Grecu, A. Ivanov, R. Saleh, and P. Pande, "Testing network-on-chip communication fabrics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, pp. 2201–2214, Dec 2007.

[4] 2015. http://www.tilera.com/products/press-release.

[5] P. Pande, C. Grecu, A. Ivanov, R. Saleh, and G. De Micheli, "Design, synthesis, and test of networks on chips," *IEEE Design Test of Computers*, vol. 22, pp. 404–413, Sept 2005.

[6] M. Kakoee, V. Bertacco, and L. Benini, "A distributed and topology-agnostic approach for on-line noc testing," in *International Symposium on Networks on Chip (NoCS)*, pp. 113–120, May 2011.

[7] M. Kakoee, V. Bertacco, and L. Benini, "At-speed distributed functional testing to detect logic and delay faults in nocs," *IEEE Transactions on Computers*, vol. 63, pp. 703–717, March 2014.

[8] S. Kundu and S. Chattopadhyay, *Network-on-Chip: The Next Generation of System-on-Chip Integration*. CRC Press, Taylor & Francis Group, 2014.

[9] C. Grecu, P. P. Pande, A. Ivanov, and R. Saleh, "Structured interconnect architecture: A solution for the non-scalability of bus-based socs," in *Great Lakes Symposium on VLSI (GLSVLSI)*, pp. 192–195, ACM, 2004.

[10] T. Bjerregaard and S. Mahadevan, "A survey of research and practices of network-on-chip," *ACM Comput. Surv.*, vol. 38, June 2006.

[11] P. Pande, C. Grecu, A. Ivanov, and R. Saleh, "Switch-based interconnect architecture for future systems on chip," *SPIE-Int. Soc. Opt. Eng*, vol. 5117, pp. 228–37, 2003.

[12] A. Amory, E. Briao, E. Cota, M. Lubaszewski, and F. Moraes, "A scalable test strategy for network-on-chip routers," in *International Test Conference (ITC)*, pp. 449–599, Nov 2005.

[13] P. Gratz, C. Kim, R. McDonald, S. W. Keckler, and D. Burger, "Implementation and evaluation of on-chip network architectures," in *International Conference on Computer Design*, pp. 477–484, Oct 2006.

[14] N. E. Jerger and L. shiuan Peh, *On-Chip Networks*. Morgan & Claypool, 2009.

[15] L. Benini and G. De Micheli, "Networks on chips: A new soc paradigm," *Computer*, vol. 35, pp. 70–78, Jan. 2002.

[16] L. Benini and G. D. Micheli, "Networks on chip: a new paradigm for systems on chip design," in *Design, Automation and Test in Europe Conference and Exhibition*, pp. 418–419, 2002.

[17] S. Pasricha and N. Dutt, *On-Chip Communication Architectures: System on Chip Interconnect*. Morgan Kaufmann Publishers Inc., 2008.

[18] J. Flich and D. Bertozzi, *Designing Network On-Chip Architectures in the Nanoscale Era*. Chapman & Hall/CRC, 2010.

[19] U. Y. Ogras, J. Hu, and R. Marculescu, "Key research problems in noc design: A holistic perspective," in *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pp. 69–74, ACM, 2005.

[20] R. Marculescu, U. Y. Ogras, L. S. Peh, N. E. Jerger, and Y. Hoskote, "Outstanding research problems in noc design: System, microarchitecture, and circuit perspectives," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, pp. 3–21, Jan 2009.

[21] W. Zhang, W. Wu, and K. Wang, "Network on chip and key research problems," in *International Conference on E-Product E-Service and E-Entertainment*, pp. 1–5, Nov 2010.

[22] V. Catania, A. Mineo, S. Monteleone, M. Palesi, and D. Patti, "Cycle-accurate network on chip simulation with noxim," *ACM Trans. Model. Comput. Simul.*, vol. 27, pp. 4:1–4:25, Aug. 2016.

[23] A. Alaghi, N. Karimi, M. Sedghi, and Z. Navabi, "Online noc switch fault detection and diagnosis using a high level fault model," in *International Symposium on Defect and Fault-Tolerance in VLSI Systems.*, pp. 21–29, Sept 2007.

[24] É. Cota, A. de Morais Amory, and M. S. Lubaszewski, *Reliability, Availability and Serviceability of Networks-on-chip.* Springer, 2011.

[25] C. Zeferino and A. Susin, "Socin: a parametric and scalable network-on-chip," in *Symposium on Integrated Circuits and Systems Design (SBCCI)*, pp. 169–174, Sept 2003.

[26] C. Feng, Z. Lu, A. Jantsch, M. Zhang, and Z. Xing, "Addressing transient and permanent faults in noc with efficient fault-tolerant deflection router," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 21, pp. 1053–1066, June 2013.

[27] Q. Yu, J. Cano, J. Flich, and P. Ampadu, "Transient and permanent error control for high-end multiprocessor systems-on-chip," in *International Symposium on Networks on Chip (NoCS)*, pp. 169–176, May 2012.

[28] A. Ghofrani, R. Parikh, S. Shamshiri, A. DeOrio, K. T. Cheng, and V. Bertacco, "Comprehensive online defect diagnosis in on-chip networks," in *VLSI Test Symposium (VTS)*, pp. 44–49, April 2012.

[29] A. Frantz, L. Carro, E. Cota, and F. Kastensmidt, "Evaluating seu and crosstalk effects in network-on-chip routers," in *International On-Line Testing Symposium*, p. 2, 2006.

[30] M. Richter and K. Chakrabarty, "Test pin count reduction for noc-based test delivery in multicore socs," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 787–792, March 2012.

[31] M. Agrawal and K. Chakrabarty, "Test-time optimization in noc-based manycore socs using multicast routing," in *VLSI Test Symposium*, pp. 1–6, April 2014.

[32] T. Han, I. Choi, H. Oh, and S. Kang, "A scalable and parallel test access strategy for noc-based multicore system," in *Asian Test Symposium (ATS)*, pp. 81–86, Nov 2014.

[33] L. Huang, J. Wang, M. Ebrahimi, M. Daneshtalab, X. Zhang, G. Li, and A. Jantsch, "Non-blocking testing for network-on-chip," *IEEE Transactions on Computers*, vol. 65, pp. 679–692, March 2016.

[34] A. Dalirsani, M. E. Imhof, and H. J. Wunderlich, "Structural software-based self-test of network-on-chip," in *VLSI Test Symposium (VTS)*, pp. 1–6, April 2014.

[35] A. Dalirsani and H. J. Wunderlich, "Functional diagnosis for graceful degradation of noc switches," in *Asian Test Symposium (ATS)*, pp. 246–251, Nov 2016.

[36] M. S. Abdelfattah, A. Bitar, and V. Betz, "Take the highway: Design for embedded nocs on fpgas," in *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 98–107, ACM, 2015.

[37] E. Cota, F. Kastensmidt, M. Cassel, P. Meirelles, A. Amory, and M. Lubaszewski, "Redefining and testing interconnect faults in mesh nocs," in *International Test Conference*, pp. 1–10, Oct 2007.

[38] E. Cota, F. Kastensmidt, M. Cassel, M. Herve, P. Almeida, P. Meirelles, A. Amory, and M. Lubaszewski, "A high-fault-coverage approach for the test of data, control and handshake interconnects in mesh networks-on-chip," *IEEE Transactions on Computers*, vol. 57, pp. 1202–1215, Sept 2008.

[39] M. Hervé, M. Moraes, P. Almeida, M. Lubaszewski, F. Kastensmidt, and r. Cota, "Functional test of mesh-based nocs with deterministic routing: Integrating the test of interconnects and routers," *Journal of Electronic Testing*, vol. 27, no. 5, pp. 635–646, 2011.

[40] C. Concatto, P. Almeida, F. Kastensmidt, E. Cota, M. Lubaszewski, and M. Herve, "Improving yield of torus nocs through fault-diagnosis-and-repair of interconnect faults," in *International On-Line Testing Symposium*, pp. 61–66, June 2009.

[41] C. Concatto, J. a. Almeida, G. Fachini, M. Hervé, F. Kastensmidt, ı. Cota, and M. Lubaszewski, "Improving the yield of noc-based systems through fault diagnosis and adaptive routing," *J. Parallel Distrib. Comput.*, vol. 71, pp. 664–674, May 2011.

[42] N. Caselli, A. Strano, D. Ludovici, and D. Bertozzi, "Cooperative built-in self-testing and self-diagnosis of noc bisynchronous channels," in *International Symposium on Embedded Multicore Socs (MCSoC)*, pp. 159–166, Sept 2012.

[43] J. Alshraiedeh and A. Kodi, "An adaptive routing algorithm to improve lifetime reliability in nocs architecture," in *International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pp. 127–130, Sept 2016.

[44] J. Liu, J. Harkin, Y. Li, and L. Maguire, "Online traffic-aware fault detection for networks-on-chip," *Journal of Parallel and Distributed Computing*, vol. 74, no. 1, pp. 1984 – 1993, 2014.

[45] A. Dalirsani, S. Holst, M. Elm, and H.-J. Wunderlich, "Structural test and diagnosis for graceful degradation of noc switches," *J. Electron. Test.*, vol. 28, pp. 831–841, Dec. 2012.

[46] C. Chen, Y. Lu, and S. Cotofana, "A novel flit serialization strategy to utilize partially faulty links in networks-on-chip," in *International Symposium on Networks on Chip (NoCS)*, pp. 124–131, May 2012.

[47] M. Radetzki, C. Feng, X. Zhao, and A. Jantsch, "Methods for fault tolerance in networks-on-chip," *ACM Comput. Surv.*, vol. 46, pp. 8:1–8:38, July 2013.

[48] P. Ren, Q. Meng, X. Ren, and N. Zheng, "Fault-tolerant routing for on-chip network without using virtual channels," in *Design Automation Conference*, pp. 102:1–102:6, ACM, 2014.

[49] Intel Core2-Extreme-Processors, [Online]: http://ark.intel.com/products/series/27970/Intel-Core2-Extreme-Processor-QX6000-Series.

[50] Adapteva Epiphany multicore architecture, [Online]: http://www.adapteva.com/products/epiphany-architecture-ip/.

[51] L. Carloni, P. Pande, and Y. Xie, "Networks-on-chip in emerging interconnect paradigms: Advantages and challenges," in *International Symposium on Networks-on-Chip*, pp. 93–102, May 2009.

[52] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures," *IEEE Trans. Comput.*, vol. 54, pp. 1025–1040, Aug. 2005.

[53] A. Ganguly, P. Pande, B. Belzer, and C. Grecu, "Design of low power and reliable networks on chip through joint crosstalk avoidance and multiple error correction coding," *Journal of Electronic Testing*, vol. 24, no. 1-3, pp. 67–81, 2008.

[54] International Techbology Roadmap for Semiconductors, [Online]: http://www.itrs.net/.

[55] R. Saleh, S. Wilton, S. Mirabbasi, A. Hu, M. Greenstreet, G. Lemieux, P. P. Pande, C. Grecu, and A. Ivanov, "System-on-chip: Reuse and integration," *Proceedings of the IEEE*, vol. 94, pp. 1050–1069, June 2006.

[56] http://www.physi.uni-heidelberg.de/ angelov/VHDL/VHDL_SS09_Teil08.pdf.

[57] Semiconductor Intellectual Property Core, [Online] : https://en.wikipedia.org/wiki/Semiconductor_intellectual_property_core.

[58] NoC IP Core, [Online]: https://www.design-reuse.com/sip/network-on-chip-c-19/.

[59] Microsemi FPGA SoC, [Online]: https://www.microsemi.com/products/fpga-soc/design-resources/ip-cores.

[60] Atlas Processor Core, [Online]: https://opencores.org/project,atlas_core.

[61] Intel FPGA IP Core, [Online]: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/ug_intro_to_megafunctions.pdf.

[62] System on a chip, [Online]: https://en.wikipedia.org/wiki/System_on_a_chip.

[63] IP Core in VLSI, [Online]: https://www.quora.com/What-is-an-IP-intellectual-property-core-in-VLSI.

[64] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., 2003.

[65] C. Zeferino, M. Kreutz, and A. Susin, "Rasoc: a router soft-core for networks-on-chip," in *Design, Automation and Test in Europe Conference and Exhibition*, vol. 3, pp. 198–203 Vol.3, Feb 2004.

[66] D. Bertozzi and L. Benini, "Xpipes: a network-on-chip architecture for gigascale systems-on-chip," *IEEE Circuits and Systems Magazine*, vol. 4, no. 2, pp. 18–31, 2004.

[67] R. R. Tamhankar, S. Murali, and G. D. Micheli, "Performance driven reliable link design for networks on chips," in *Asia and South Pacific Design Automation Conference*, pp. 749–754 Vol. 2, Jan 2005.

[68] K. Goossens, J. Dielissen, J. van Meerbergen, P. Poplavko, A. Rădulescu, E. Rijpkema, E. Waterlander, and P. Wielage, *Guaranteeing the Quality of Services in Networks on Chip*, pp. 61–82. Springer US, 2003.

[69] M. Millberg, E. Nilsson, R. Thid, and A. Jantsch

[70] K. Goossens, J. Dielissen, and A. Radulescu, "Aethereal network on chip: Concepts, architectures, and implementations," *IEEE Des. Test*, vol. 22, pp. 414–421, Sept. 2005.

[71] P. T. Wolkotte, G. J. Smit, G. K. Rauwerda, and L. T. Smit, "An energy-efficient reconfigurable circuit switched network-on-chip," in *International Parallel and Distributed Processing Symposium (IPDPS) and Reconfigurable Architecture Workshop (RAW)*, IEEE Computer Society, 2005.

[72] A. Banerjee, R. Mullins, and S. Moore, "A power and energy exploration of network-on-chip architectures," in *International Symposium on Networks-on-Chip*, pp. 163–172, IEEE Computer Society, 2007.

[73] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani, "A network on chip architecture and design methodology," in *IEEE Computer Society Annual Symposium on VLSI*, pp. 105–112, 2002.

[74] D. Sigüenza-Tortosa, T. Ahonen, and J. Nurmi, "Issues in the development of a practical noc: the proteo concept," *Integration, the VLSI Journal*, vol. 38, no. 1, pp. 95 – 105, 2004.

[75] A. Adriahantenaina, H. Charlery, A. Greiner, L. Mortiez, and C. A. Zeferino, "Spin: a scalable, packet switched, on-chip micro-network," in *Design, Automation and Test in Europe Conference and Exhibition*, pp. 70–73 suppl., 2003.

[76] T. Bjerregaard and J. Sparso, "A router architecture for connection-oriented service guarantees in the mango clockless network-on-chip," in *Design, Automation and Test in Europe*, pp. 1226–1231, IEEE Computer Society, 2005.

[77] M. Dehyadgari, M. Nickray, A. Afzali-kusha, and Z. Navabi, "Evaluation of pseudo adaptive xy routing using an object oriented model for noc," in *International Conference on Microelectronics*, pp. 5 pp.–, Dec 2005.

[78] C. J. Glass and L. M. Ni, "The turn model for adaptive routing," *SIGARCH Comput. Archit. News*, vol. 20, pp. 278–287, Apr. 1992.

[79] H. Kariniemi and J. Nurmi, "Arbitration and routing schemes for on-chip packet networks," in *Interconnect-centric design for advanced SoC and NoC*, pp. 253–282, Springer, 2005.

[80] M. Majer, C. Bobda, A. Ahmadinia, and J. Teich, "Packet routing in dynamically changing networks on chip," in *International Parallel and Distributed Processing Symposium*, pp. 154b–154b, April 2005.

[81] J. Kim, D. Park, T. Theocharides, N. Vijaykrishnan, and C. R. Das, "A low latency router supporting adaptivity for on-chip interconnects," in *Design Automation Conference*, pp. 559–564, June 2005.

[82] B. A. Forouzan, *Data Communications and Networking (McGraw-Hill Forouzan Networking)*. McGraw-Hill Higher Education, 2007.

[83] T. Chelcea and S. M. Nowick, "Robust interfaces for mixed-timing systems with application to latency-insensitive protocols," in *Design Automation Conference*, pp. 21–26, ACM, 2001.

[84] D. Siguenza-Tortosa and J. Nurmi, "Proteo: a new approach to network-on-chip," *IASTED-Communication Systems and Networks (CSN)*, 2002.

[85] W. J. Dally and B. Towles, "Route packets, not wires: On-chip inteconnection networks," in *Design Automation Conference*, pp. 684–689, ACM, 2001.

[86] F. Karim, A. Nguyen, S. Dey, and R. Rao, "On-chip communication architecture for oc-768 network processors," in *Design Automation Conference*, pp. 678–683, June 2001.

[87] F. Karim, A. Nguyen, and S. Dey, "An interconnect architecture for networking systems on chips," *IEEE Micro*, vol. 22, pp. 36–45, Sept. 2002.

[88] M. Coppola, S. Curaba, M. D. Grammatikakis, G. Maruccia, and F. Papariello, "Occn: a network-on-chip modeling and simulation framework," in *Design, Automation and Test in Europe Conference and Exhibition*, vol. 3, pp. 174–179 Vol.3, Feb 2004.

[89] M. Coppola, R. Locatelli, G. Maruccia, L. Pieralisi, and A. Scandurra, "Spidergon: a novel on-chip communication network," in *International Symposium on System-on-Chip*, pp. 15–, Nov 2004.

[90] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C.-C. Miao, J. F. Brown III, and A. Agarwal, "On-chip interconnection architecture of the tile processor," *IEEE Micro*, vol. 27, pp. 15–31, Sept. 2007.

[91] J. Balfour and W. J. Dally, "Design tradeoffs for tiled cmp on-chip networks," in *International Conference on Supercomputing*, pp. 187–198, ACM, 2006.

[92] D. Rahmati, A. E. Kiasari, S. Hessabi, and H. Sarbazi-Azad, "A performance and power analysis of wk-recursive and mesh networks for network-on-chips," in *International Conference on Computer Design*, pp. 142–147, IEEE, 2007.

[93] P. Guerrier and A. Greiner, "A generic architecture for on-chip packet-switched interconnections," in *Conference on Design, Automation and Test in Europe*, pp. 250–256, ACM, 2000.

[94] H. Hossain, M. Akbar, and M. Islam, "Extended-butterfly fat tree interconnection (efti) architecture for network on chip," in *Pacific Rim Conference on Communications, Computers and signal Processing*, pp. 613–616, Aug 2005.

[95] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," in *International Symposium on Microarchitecture (MICRO)*, pp. 172–182, Dec 2007.

[96] S. Kundu and S. Chattopadhyay, "Network-on-chip architecture design based on mesh-of-tree deterministic routing topology," *International Journal of High Performance Systems Architecture*, vol. 1, no. 3, pp. 163–182, 2008.

[97] L. Benini, "Application specific noc design," in *Design, Automation and Test in Europe*, pp. 491–495, 2006.

[98] J. Duato, S. Yalamanchili, and N. Lionel, *Interconnection Networks: An Engineering Approach*. Morgan Kaufmann Publishers Inc., 2002.

[99] M. Bushnell and V. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Springer Publishing Company, Incorporated, 2013.

[100] E. Larsson, *Introduction to advanced system-on-chip test design and optimization*, vol. 29. Springer Science & Business Media, 2006.

[101] S. Mourad and Y. Zorian, *Principles of testing electronic systems*. John Wiley, 2000. A Wiley-Interscience publication.

[102] J. Liu, J. Harkin, Y. Li, and L. Maguire, "Low cost fault-tolerant routing algorithm for networks-on-chip," *Microprocessors and Microsystems*, vol. 39, no. 6, pp. 358 – 372, 2015.

[103] W.-K. Chen, *"VLSI Technology"*. CRc Press, 2003.

[104] M. Abramovici, M. Breuer, and A. Friedman, *"Digital Systems Testing and Testable Design"*. IEEE Press, 1994.

[105] L.-T. Wang, C.-W. Wu, and X. Wen, *VLSI Test Principles and Architectures: Design for Testability*. Morgan Kaufmann Publishers Inc., 2006.

[106] W. Shi and W. K. Fuchs, "Optimal interconnect diagnosis of wiring networks," *IEEE Trans. on VLSI Systems*, vol. 3, pp. 430–436, 1995.

[107] J. H. Patel, "Stuck-at fault: a fault model for the next millennium," in *International Test Conference*, pp. 1166–, Oct 1998.

[108] W. Feng, F. Meyer, and F. Lombardi, "Two-step algorithms for maximal diagnosis of wiring interconnects," in *International Symposium on Fault-Tolerant Computing*, pp. 130–137, June 1999.

[109] R. L. Wadsack, "Fault modeling and logic simulation of cmos and mos integrated circuits," *The Bell System Technical Journal*, vol. 57, pp. 1449–1474, May 1978.

[110] M. Renovell and G. N. Cambon, "Electrical analysis and modeling of floating-gate fault," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, pp. 1450–1458, Nov 1992.

[111] W. Zou, W. t. Cheng, and S. M. Reddy, "Interconnect open defect diagnosis with physical information," in *Asian Test Symposium*, pp. 203–209, Nov 2006.

[112] C. Liu, W. Zou, S. M. Reddy, W.-T. Cheng, M. Sharma, and H. Tang, "Interconnect open defect diagnosis with minimal physical information," in *International Test Conference*, pp. 1–10, Oct 2007.

[113] S. Spinner, I. Polian, P. Engelke, B. Becker, M. Keim, and W.-T. Cheng, "Automatic test pattern generation for interconnect open defects," in *VLSI Test Symposium*, pp. 181–186, April 2008.

[114] S. Hillebrecht, I. Polian, P. Engelke, B. Becker, M. Keim, and W. T. Cheng, "Extraction, simulation and test generation for interconnect open defects based on enhanced aggressor-victim model," in *International Test Conference*, pp. 1–10, Oct 2008.

[115] Y. H. Chen, C. L. Chang, and C. H. P. Wen, "Diagnostic test-pattern generation targeting open-segment defects and its diagnosis flow," *IET Computers Digital Techniques*, vol. 6, pp. 186–193, May 2012.

[116] J.-C. Lien and M. A. Breuer, "Maximal diagnosis for wiring networks," pp. 96–105, 1991.

[117] W. H. Kautz, "Testing for faults in wiring networks," *Computers, IEEE Transactions on*, vol. C-23, pp. 358–363, April 1974.

[118] N. Jarwala and C. W.Yau, "A new framework for analyzing test generation and diagnosis algorithms for wiring interconnects," in *International Test Conference*, pp. 63–70, Aug 1989.

[119] Y. Kim, H. don Kim, and S. Kang, "A new maximal diagnosis algorithm for interconnect test," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, pp. 532–537, May 2004.

[120] M. Cuviello, S. Dey, X. Bai, and Y. Zhao, "Fault modeling and simulation for crosstalk in system-on-chip interconnects," in *International Conference on Computer-Aided Design, 1999. Digest of Technical Papers.*, pp. 297–303, Nov 1999.

[121] B. Aghaei, A. Khademzadeh, M. Reshadi, and K. Badie, "Link testing: a survey of current trends in network on chip," *Journal of Electronic Testing*, vol. 33, no. 2, pp. 209–225, 2017.

[122] G. Nazarian, "On-line testing of routers in networks-on-chip," dec 2008.

[123] R. D. Eldred, "Test routines based on symbolic logical statements," *J. ACM*, vol. 6, pp. 33–37, Jan. 1959.

[124] B. G. West, "At-speed structural test," in *International Test Conference)*, pp. 795–800, 1999.

[125] A. Dalirsani, "Self-diagnosis in network-on-chips," 2015.

[126] M. Sebastian, P. Axer, R. Ernst, N. Feiertag, and M. Jersak, "Efficient reliability and safety analysis for mixed-criticality embedded systems," tech. rep., SAE Technical Paper, 2011.

[127] R. Bishnoi, P. Kumar, V. Laxmi, M. S. Gaur, and A. Sikka, "Distributed adaptive routing for spidergon noc," in *18th International Symposium on VLSI Design and Test*, pp. 1–6, July 2014.

[128] E. J. Marinissen, S. K. Goel, and M. Lousberg, "Wrapper design for embedded core test," in *International Test Conference*, pp. 911–920, 2000.

[129] IEEE P1500 - A Standard for Embedded Core Test, [online]: http://grouper.ieee.org/groups/1500/,.

[130] E. Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg, and C. Wouters, "A structured and scalable mechanism for test access to embedded reusable cores," in *International Test Conference*, pp. 284–293, Oct 1998.

[131] M. Nahvi and A. Ivanov, "A packet switching communication-based test access mechanism for system chips," in *IEEE European Test Workshop*, pp. 81–86, 2001.

[132] E. Cota, L. Carro, F. Wagner, and M. Lubaszewski, "Power-aware noc reuse on the testing of core-based systems," in *International Test Conference*, vol. 1, pp. 612–621, Sept 2003.

[133] E. Cota, L. Carro, and M. Lubaszewski, "Reusing an on-chip network for the test of core-based systems," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 9, pp. 471–499, Oct. 2004.

[134] I. Ghosh, S. Dey, and N. K. Jha, "A fast and low-cost testing technique for core-based system-chips," *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol. 19, pp. 863–877, Nov. 2006.

[135] J. Raik, V. Govind, and R. Ubar, "An external test approach for network-on-a-chip switches," in *Asian Test Symposium*, pp. 437–442, Nov 2006.

[136] K. Stewart and S. Tragoudas, "Interconnect testing for networks on chips," in *VLSI Test Symposium*, pp. 6 pp.–, April 2006.

[137] C. Grecu, P. Pande, B. Wang, A. Ivanov, and R. Saleh, "Methodologies and algorithms for testing switch-based noc interconnects," in *International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, pp. 238–246, Oct 2005.

[138] K. Petersen and J. Oberg, "Toward a scalable test methodology for 2d-mesh network-on-chips," in *Design, Automation Test in Europe Conference Exhibition*, pp. 1–6, April 2007.

[139] C. Grecu, P. Pande, A. Ivanov, and R. Saleh, "Bist for network-on-chip interconnect infrastructures," in *VLSI Test Symposium*, pp. 6 pp.–35, April 2006.

[140] M. Hosseinabady, A. Banaiyan, M. Bojnordi, and Z. Navabi, "A concurrent testing method for noc switches," in *Design, Automation and Test in Europe*, vol. 1, pp. 6 pp.–, March 2006.

[141] M. Sedghi, A. Alaghi, E. Koopahi, and Z. Navabi, "An hdl-based platform for high level noc switch testing," in *Asian Test Symposium*, pp. 453–458, Oct 2007.

[142] M. Hosseinabady, A. Dalirsani, and Z. Navabi, "Using the inter- and intra-switch regularity in noc switch testing," in *Design, Automation Test in Europe Conference Exhibition*, pp. 1–6, April 2007.

[143] S. Babaei, M. Mansouri, B. Aghaei, and A. Khadem-zadeh, "Online-structural testing of routers in network on chip."

[144] T. Lehtonen, D. Wolpert, P. Liljeberg, J. Plosila, and P. Ampadu, "Self-adaptive system for addressing permanent errors in on-chip interconnects," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, pp. 527–540, April 2010.

[145] S. Shamshiri, A. Ghofrani, and K. T. Cheng, "End-to-end error correction and online diagnosis for on-chip networks," in *IEEE International Test Conference*, pp. 1–10, Sept 2011.

[146] H. Zimmer and A. Jantsch, "A fault model notation and error-control scheme for switch-to-switch buses in a network-on-chip," in *First IEEE/ACM/IFIP International Conference on Hardware/ Software Codesign and Systems Synthesis)*, pp. 188–193, 2003.

[147] S. Murali, T. Theocharides, N. Vijaykrishnan, M. J. Irwin, L. Benini, and G. D. Micheli, "Analysis of error recovery schemes for networks on chips," *IEEE Design Test of Computers*, vol. 22, pp. 434–442, Sept 2005.

[148] D. Bertozzi, L. Benini, and G. De Micheli, "Error control schemes for on-chip communication links: The energy-reliability tradeoff," *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol. 24, pp. 818–831, Nov. 2006.

[149] Q. Yu and P. Ampadu, "Transient and permanent error co-management method for reliable networks-on-chip," in *International Symposium on Networks-on-Chip*, pp. 145–154, May 2010.

[150] A. Ejlali, B. M. Al-Hashimi, P. Rosinger, and S. G. Miremadi, "Joint consideration of fault-tolerance, energy-efficiency and performance in on-chip networks," in *Design, Automation Test in Europe Conference Exhibition*, pp. 1–6, April 2007.

[151] A. Vitkovskiy, V. Soteriou, and C. Nicopoulos, "A dynamically adjusting gracefully degrading link-level fault-tolerant mechanism for nocs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, pp. 1235–1248, Aug 2012.

[152] A. Runge, "Fault-tolerant network-on-chip based on fault-aware flits and deflection routing," in *International Symposium on Networks-on-Chip*, pp. 9:1–9:8, ACM, 2015.

[153] S. Abed, M. AlShayeji, Z. Abdullah, and Z. Al-Saeed, "Fault tolerance design for nocs: Partial virtual-channel sharing," in *International Conference on Software and Computer Applications*, pp. 233–238, ACM, 2017.

[154] N. Chatterjee, S. Paul, and S. Chattopadhyay, "Fault-tolerant dynamic task mapping and scheduling for network-on-chip-based multicore platform," *ACM Trans. Embed. Comput. Syst.*, vol. 16, pp. 108:1–108:24, May 2017.

[155] S. Werner, J. Navaridas, and M. Luján, "A survey on design approaches to circumvent permanent faults in networks-on-chip," *ACM Comput. Surv.*, vol. 48, pp. 59:1–59:36, Mar. 2016.

[156] T. Bengtsson, S. Kumar, R. Ubar, and A. Jutman, "Off-line testing of crosstalk induced glitch faults in noc interconnects," in *NORCHIP*, pp. 221–225, Nov 2006.

[157] M. Botelho, F. Kastensmidt, M. Lubaszewski, E. Cota, and L. Carro, "A broad strategy to detect crosstalk faults in network-on-chip interconnects," in *VLSI System on Chip Conference*, pp. 298–303, Sept 2010.

[158] R. Nourmandi-Pour, A. Khadem-Zadeh, and A. M. Rahmani, "An ieee 1149.1-based bist method for at-speed testing of inter-switch links in network on chip," *Microelectronics Journal*, vol. 41, no. 7, pp. 417 – 429, 2010.

[159] T. Bengtsson, A. Jutman, S. Kumar, R. Ubar, and Z. Peng, "Off-line testing of delay faults in noc interconnects," in *EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools*, pp. 677–680, 2006.

[160] S. Borkar, "Microarchitecture and design challenges for gigascale integration," in *Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 3–3, IEEE Computer Society, 2004.

[161] N. Miskov-Zivanov and D. Marculescu, "Multiple transient faults in combinational and sequential circuits: A systematic approach," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, pp. 1614–1627, Oct 2010.

[162] T. Boraten and A. K. Kodi, "Runtime techniques to mitigate soft errors in network-on-chip (noc) architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, pp. 682–695, March 2018.

[163] M. Herve, E. Cota, F. Kastensmidt, and M. Lubaszewski, "Diagnosis of interconnect shorts in mesh nocs," in *International Symposium on Networks-on-Chip*, pp. 256–265, May 2009.

[164] E. A. Rambo, A. Tschiene, J. Diemer, L. Ahrendts, and R. Ernst, "Fmea-based analysis of a network-on-chip for mixed-critical systems," in *International Symposium on Networks-on-Chip (NoCS)*, pp. 33–40, Sept 2014.

[165] A. Agarwal, C. Iskander, and R. Shankar, "Survey of network on chip (noc) architectures & contributions," *Journal of engineering, Computing and Architecture*, vol. 3, no. 1, pp. 21–27, 2009.

[166] B. S. Feero and P. P. Pande, "Networks-on-chip in a three-dimensional environment: A performance evaluation," *IEEE Transactions on Computers*, vol. 58, pp. 32–45, Jan 2009.

[167] J. W. McPherson, "Reliability challenges for 45nm and beyond," in *Design Automation Conference*, pp. 176–181, ACM, 2006.

[168] M. Herve, P. Almeida, F. Kastensmidt, E. Cota, and M. Lubaszewski, "Concurrent test of network-on-chip interconnects and routers," in *Latin American Test Workshop*, pp. 1–6, March 2010.

[169] L.-T. Wang, C.-W. Wu, and X. Wen, *VLSI Test Principles and Architectures: Design for Testability (Systems on Silicon)*. Morgan Kaufmann Publishers Inc., 2006.

[170] B. Bhowmik, *Design and Analysis of Algorithms*. Katson, 2nd ed., 2012.

[171] F. Gebali, H. Elmiligi, and M. W. El-Kharashi, *Networks-on-Chips: Theory and Practice*. CRC Press, Inc., 1st ed., 2009.

[172] G. V. Varatkar and R. Marculescu, "On-chip traffic modeling and synthesis for mpeg-2 video applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, pp. 108–119, Jan 2004.

[173] D. R. Avresky, V. Shurbanov, R. W. Horst, and P. Mehra, "Performance evaluation of the servernet r san under self-similar traffic," in *International Symposium on Parallel Processing and Symposium on Parallel and Distributed Processing*, pp. 143–147, IEEE Computer Society, 1999.

[174] M. F. A. Qasem and H. Gu, "Square-octagon interconnection architecture for network-on-chips," in *International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pp. 715–719, Aug 2014.

[175] B. Ghoshal, K. Manna, S. Chattopadhyay, and I. Sengupta, "In-field test for permanent faults in fifo buffers of noc routers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, pp. 393–397, Jan 2016.

[176] S. V. R. Chittamuru, S. Desai, and S. Pasricha, "Swiftnoc: A reconfigurable silicon-photonic network with multicast-enabled channel sharing for multicore architectures," *J. Emerg. Technol. Comput. Syst.*, vol. 13, pp. 58:1–58:27, June 2017.

[177] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar, "A 5-ghz mesh interconnect for a teraflops processor," *IEEE Micro*, vol. 27, pp. 51–61, Sept 2007.

[178] D. Xiang, K. Chakrabarty, and H. Fujiwara, "A unified test and fault-tolerant multicast solution for network-on-chip designs," in *International Test Conference (ITC)*, pp. 1–9, IEEE, 2016.

[179] A. Ghiribaldi, D. Ludovici, F. Triviño, A. Strano, J. Flich, J. L. Sánchez, F. Alfaro, M. Favalli, and D. Bertozzi, "A complete self-testing and self-configuring noc infrastructure for cost-effective mpsocs," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 12, no. 4, p. 106, 2013.

[180] H. J. Mahanta, A. Biswas, and M. A. Hussain, "Networks on chip: The new trend of on-chip interconnection," in *International Conference on Communication Systems and Network Technologies*, pp. 1050–1053, April 2014.

[181] H. Xue, C. Di, and J. A. G. Jess, "Probability analysis for cmos floating gate faults," in *European Design and Test Conference*, pp. 443–448, Feb 1994.

[182] C. F. Hawkins, J. M. Soden, A. W. Righter, and F. J. Ferguson, "Defect classes-an overdue paradigm for cmos ic testing," in *International Test Conference*, pp. 413–425, Oct 1994.

[183] L. Benini, , and G. D. Micheli, "Networks on chips," Morgan Kaufmann, first ed., 2006.

[184] D. Fick, A. DeOrio, J. Hu, V. Bertacco, D. Blaauw, and D. Sylvester, "Vicis: A reliable network for unreliable silicon," in *Design Automation Conference*, pp. 812–817, ACM, 2009.

[185] A. Strano, C. Gómez, D. Ludovici, M. Favalli, M. E. Gómez, and D. Bertozzi, "Exploiting network-on-chip structural redundancy for a cooperative and scalable built-in self-test architecture," in *Design, Automation Test in Europe*, pp. 1–6, March 2011.

[186] R. Ho, K. W. Mai, and M. A. Horowitz, "The future of wires," *Proceedings of the IEEE*, vol. 89, pp. 490–504, Apr 2001.

[187] Z. Zhang, A. Greiner, and S. Taktak, "A reconfigurable routing algorithm for a fault-tolerant 2d-mesh network-on-chip," in *Design Automation Conference*, pp. 441–446, June 2008.

[188] http://opencores.org/forum,Cores.

[189] R. Prolonge and F. Clermidy, "Network-on-chip traffic modeling for data flow applications," in *Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools*, pp. 2:1–2:6, ACM, 2013.

[190] D. Pradhan and J. Stiffler, "Error-correcting codes and self-checking circuits," *Computer*, vol. 13, no. 3, pp. 27–37, 1980.

[191] R. Holsmark and S. Kumar, "Design issues and performance evaluation of mesh noc with regions," in *NORCHIP*, pp. 40–43, Nov 2005.

[192] R. Abdel-Khalek and V. Bertacco, "Functional post-silicon diagnosis and debug for networks-on-chip," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 557–563, Nov 2012.

[193] H. Kim, A. Vitkovskiy, P. V. Gratz, and V. Soteriou, "Use it or lose it: Wear-out and lifetime in future chip multiprocessors," in *International Symposium on Microarchitecture (MICRO)*, pp. 136–147, Dec 2013.

[194] R. Abdel-Khalek and V. Bertacco, "Post-silicon platform for the functional diagnosis and debug of networks-on-chip," *ACM Trans. Embed. Comput. Syst.*, vol. 13, pp. 112:1–112:25, Mar. 2014.

[195] M. F. A. Qasem and H. Gu, "Square-octagon interconnection architecture for network-on-chips," in *International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pp. 715–719, Aug 2014.

[196] W. Song, G. Zhang, and J. Garside, "On-line detection of the deadlocks caused by permanently faulty links in quasi-delay insensitive networks on chip," in *Great Lakes Symposium on VLSI (GLSVLSI)*, pp. 211–216, ACM, 2014.

[197] H.-J. Wunderlich and M. Radetzki, "Multi-layer test and diagnosis for dependable nocs," in *International Symposium on Networks-on-Chip*, pp. 5:1–5:8, ACM, 2015.

[198] M. Moadeli, A. Shahrabi, W. Vanderbauwhede, and P. Maji, "An analytical performance model for the spidergon noc with virtual channels," *Journal of Systems Architecture*, vol. 56, no. 1, pp. 16 – 26, 2010.

[199] B. Aghaei, A. Khademzadeh, M. Reshadi, and K. Badie, "A new bist-based test approach with the fault location capability for communication channels in network-on-chip," *Journal of Electronic Testing*, Jun 2017.

[200] H. Cheng, S. Y. Jiang, Y. Liu, S. S. Jiang, and L. Chen, "A test method for delay faults in noc interconnects," in *International Conference on Applied Superconductivity and Electromagnetic Devices (ASEMD)*, pp. 371–374, Oct 2013.

[201] S. Kajihara, S. Ohtake, and T. Yoneda, "Delay testing: Improving test quality and avoiding over-testing," *Information and Media Technologies*, vol. 6, no. 4, pp. 1053–1066, 2011.

[202] V. F. Pavlidis and E. G. Friedman, "3-d topologies for networks-on-chip," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 10, pp. 1081–1090, 2007.

[203] K. Kim, B. Floyd, J. Mehta, H. Yoon, C.-m. Hung, D. Bravo, T. Dickson, X. Guo, R. Li, N. Trichy, *et al.*, "Silicon integrated circuits incorporating antennas," in *Custom Integrated Circuits Conference*, pp. 473–480, IEEE, 2006.

[204] J.-J. Lin, H.-T. Wu, Y. Su, L. Gao, A. Sugavanam, J. E. Brewer, *et al.*, "Communication using antennas fabricated in silicon integrated circuits," *IEEE Journal of solid-state circuits*, vol. 42, no. 8, pp. 1678–1687, 2007.

[205] K. Kempa, J. Rybczynski, Z. Huang, K. Gregorczyk, A. Vidan, B. Kimball, J. Carlson, G. Benham, Y. Wang, A. Herczynski, *et al.*, "Carbon nanotubes as optical antennae," *Advanced Materials*, vol. 19, no. 3, pp. 421–426, 2007.

[206] S. Deb, A. Ganguly, P. P. Pande, B. Belzer, and D. Heo, "Wireless noc as interconnection backbone for multicore chips: Promises and challenges," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 2, pp. 228–239, June 2012.

[207] R. G. Kim, W. Choi, Z. Chen, P. P. Pande, D. Marculescu, and R. Marculescu, "Wireless noc and dynamic vfi codesign: Energy efficiency without performance penalty," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, pp. 2488–2501, July 2016.

[208] A. Shacham, B. G. Lee, A. Biberman, K. Bergman, and L. P. Carloni, "Photonic noc for dma communications in chip multiprocessors," in *Symposium on High-Performance Interconnects (HOTI)*, pp. 29–38, Aug 2007.

List of Publications

A. Related Publications

- ***Journal***

[RPJ-1]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, Bhargab B. Bhattacharya, “Reliability-Aware Test Methodology for Detecting Short-Channel Faults in On-Chip Networks”, IEEE Transactions on VLSI Systems (**TVLSI**), Vol. 26, Issue 6, 2018. Pages: 1–14.

[RPJ-2]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, “On-line Analysis of Stuck-at Faults in on-Chip Network Interconnects”, Journal of Circuits, Systems, and Computers (**JCSC**), World Scientific Publishing, Vol 27, Issue 13, 2018. Pages: 1–13.

[RPJ-3]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "A Time-Optimized Scheme Towards Analysis of Channel-Shorts in on-Chip Networks", Journal of Electronic Testing: Theory and Applications (**JETTA**), Springer, Vol. 33, Issue 2, April, 2017. Pages: 227–254.

- ***Conference***

[RPC-1]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "Charka: A Reliability-Aware Test Scheme for Diagnos of Channel Shorts Beyond Mesh NoCs", 2017 IEEE/ACM 20th Design, Automation, and Test in Europe (**IEEE/ACM DATE 2017**), March 27-31, 2017, Lausanne, Switzerland. **Conference Rank: Tier-2/B**. [Acceptance Rate $\approx$ 24%].

[RPC-2]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "When Clustering Shows Optimality Towards Analyzing Stuck-at Faults in Channels of on-Chip Networks", 2016 IEEE 18th International Conference on High Performance

Computing and Communications (**IEEE HPCC 2016**), December 12-14, 2016, Sydney, Australia. **Conference Rank: Tier-2/B**. [Acceptance Rate $\approx$ 20%].

[RPC-3]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "A Reliability-Aware Topology-Agnostic Test Scheme for Detecting, and Diagnosing Interconnect Shorts in on-Chip Networks", 2016 IEEE 18th International Conference on High Performance Computing and Communications (**IEEE HPCC 2016**), December 12-14, 2016, Sydney, Australia. **Conference Rank: Tier-2/B**. [Acceptance Rate $\approx$ 20%].

[RPC-4]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "A Concurrent Approach to Detect and Diagnose Shorts in Interconnects of on-Chip Networks", 2016 IEEE 28th Region Ten Conference (**IEEE TENCON 2016**), November 22-25, 2016, Singapore. **Conference Rank: Tier-3/C**.

[RPC-5]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "On-Line Testing of Coexistent Stuck-at and Open Faults in NoC Interconnects", 2016 IEEE 28th Region Ten Conference (**IEEE TENCON 2016**), November 22-25, 2016, Singapore. **Conference Rank: Tier-3/C**.

[RPC-6]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, Bhargab B. Bhattacharya, "A Topology-Agnostic Test Model for Link Shorts in on-Chip Networks", 2016 IEEE 29th International Conference on Systems, Man, and Cybernetics (**IEEE SMC 2016**), October 9-12, 2016, Budapest, Hungary. **Conference Rank: Tier-2/B**.

[RPC-7]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, Bhargab B. Bhattacharya, "On-Line Detection and Diagnosis of Stuck-at Faults in Channels of NoC-Based Systems", 2016 IEEE 29th International Conference on Systems, Man, and Cybernetics (**IEEE SMC 2016**), October 9-12, 2016, Budapest, Hungary. **Conference Rank: Tier-2/B**.

[RPC-8]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, Bhargab B. Bhattacharya, "Detecting and Diagnosing Open Faults in NoC Channels on Activation of Diagonal Nodes", 2016 IEEE 29th International Conference on Systems, Man, and Cybernetics (**IEEE SMC 2016**), October 9-12, 2016, Budapest, Hungary. **Conference Rank: Tier-2/B**.

[RPC-9]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, Bhargab B. Bhattacharya, "One Poison is Antidote Against Another Poison", 2016 IEEE 29th International Conference on Systems, Man, and Cybernetics (**IEEE SMC 2016**), October 9-12, 2016, Budapest, Hungary. **Conference Rank: Tier-2/B**.

[RPC-10]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "Towards a Scalable Test Solution for the Analysis of Interconnect Shorts in on-Chip Networks", 2016 IEEE 24th International Conference on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (**IEEE MASCOTS 2016**), September 19-21, 2016, London, UK. **Conference Rank: Tier-1/A**. [Acceptance Rate $\approx$ 19%].

[RPC-11]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "An On-Line Test Solution for Addressing Interconnect Shorts in on-Chip Networks", 2016 IEEE 22nd International On-Line Testing Symposium (**IEEE IOLTS 2016**), July 4-6, 2016, Catalunya, Spain. **Conference Rank: Tier-3/C**.

[RPC-12]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "An Odd-Even Scheme to Prevent a Packet from Being Corrupted and Dropped in Fault Tolerant NoCs", 2016 IEEE 22nd International On-Line Testing Symposium (**IEEE IOLTS 2016**), July 4-6, 2016, Catalunya, Spain. **Conference Rank: Tier-3/C**.

[RPC-13]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "Impact of NoC Interconnect Shorts on Performance Metrics", 2016 IEEE 22nd National Conference on Communications (**IEEE NCC 2016**), March 4-6, 2016, Guwahati, India. **Rank: Tier-3/C**.

[RPC-14]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas "An Odd-Even Model for Diagnosis of Shorts on NoC Interconnects", 2015 IEEE 12th India International Conference (**IEEE INDICON 2015**), New Delhi, India, December 17-20, 2015. [Acceptance Rate $\approx$ 33%].

[RPC-15]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "A Matrix Model for Redefining and Testing NoC Interconnect Shorts", 2015 IEEE 27th Region Ten Conference (**IEEE TENCON 2015**), November 1-4, 2015, Macau. ***[Selected as Best Paper and Received Young Scientist Award]***. **Conference Rank: Tier-3/C**.

[RPC-16]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "Reliability on Top of Best Effort Delivery: Maximal Connectivity Test on NoC Interconnects", 2015 ACM 8th India Annual Conference (**ACM Compute 2015**), October 29-31, 2015, Ghaziabad, India. [Acceptance Rate $\approx$ 19%].

[RPC-17]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "An Optimal Diagnosis of NoC Interconnects on Activation of Diagonal Routers", 2015 IEEE 28th International Conference on Systems, Man, and Cybernetics (**IEEE SMC 2015**), October 9-12, 2015, Hong Kong. **Conference Rank: Tier-2/B**.

[RPC-18]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "A Packet Address Driven Test Strategy for Stuck-at Faults in Networks-on-Chip Interconnects", 2015 IEEE 23rd Mediterranean Conference on Control and Automation (**IEEE MED 2015**)", June 16-19, 2015, Spain. [Acceptance Rate ≈ 31%].

[RPC-19]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "Detection of Faulty Interswitch Links in 2-D Mesh Network-on-Chips", 2015 IEEE 8th International Conference on Advanced Networks and Telecommunication Systems (**IEEE ANTS 2014**), New Delhi, India, December 14-17, 2014. [Acceptance Rate ≈ 33%].

B. Unrelated Publications

- ***Conference***

[URP-1]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas "Directed Symbolic Execution for VLSI Circuits", 2015 IEEE 28th International Conference on Systems, Man, and Cybernetics (**IEEE SMC 2015**), October 9-12, 2015, Hong Kong. **Conference Rank: Tier-2/B**.

[URP-2]. Biswajit Bhowmik, Santosh Biswas, Jatindra Kumar Deka, "Crossing Register Transfer Level for VLSI Circuits", 2015 IEEE International Conference on Industrial Instrumentation and Control (**IEEE ICIC 2015**), Pune, India, May 28-30, 2015. [Acceptance Rate ≈ 32%].

[URP-3]. Biswajit Bhowmik, Jatindra Kumar Deka, Santosh Biswas, "Beyond Test Pattern Generation: Coverage Analysis", 2015 IEEE International Conference on Industrial Instrumentation and Control (**IEEE ICIC 2015**), Pune, India, May 28-30, 2015. [Acceptance Rate ≈ 32%].

C. Author's Other Publications

- ***Book***

[AOPB-1]. Biswajit R Bhowmik, "Design & Analysis of Algorithms", Second Edition, Katson Publisher, ISBN: 978–93–5014–135–9, 2012, Pages: 475.

[AOPB-2]. Biswajit Bhowmik, "Design & Analysis of Algorithms", First Edition, Katson Publisher, ISBN: 978–93–5014–135–9, 2011, Pages: 275.

- ***Journal***

[AOPJ-1]. Biswajit Bhowmik, "A Comparison Study on Selective Traffic Models with Handoff Management Scheme for Wireless Mobile Network Infrastructure", International Journal of Information Technology and Computer Science (IJITCS), MECS Press, Vol. 5, No. 2, January, 2013. ISSN: 2074–9015. Pages: 66–72.

[AOPJ-2]. Biswajit Bhowmik, Pooja, Nupur Thakur, Piyali Sarkar, "Modeling Prioritized Hard Handoff Management Scheme for Wireless Mobile Networks", International Journal of Computer Network and Information Security (IJCNIS), MECS Press, Vol. 4, No. 8, August, 2012. ISSN: 2074–9104. Pages: 21–32.

[AOPJ-3]. Biswajit Bhowmik, Pooja, Nupur Thakur, Piyali Sarkar, "Experimental Analysis of Xie and Kuek's Traffic Model with Handoff Scheme in Wireless Networks", International Journal of Information Engineering and Electronic Business (IJIEEB), MECS Press, Vol. 4, No. 1, February, 2012. ISSN: 2074–9031. Pages: 34–43.

[AOPJ-4]. Biswajit Bhowmik, "Simplified Optimal Parenthesization Scheme for Matrix Chain Multiplication Problem using Bottom-up Practice in 2-Tree Structure", Journal of Applied Computer Science & Mathematics (JACSM), Vol. 5, Issue 11, October 2011. ISSN: 2066–4273. Pages: 9–14.

[AOPJ-5]. Biswajit Bhowmik, Sreyasi Nag Chowdhury, "Prograph Based Analysis of Single Source Shortest Path Problems with Few Distinct Positive Lengths", Engineering, Technology & Applied Science Research (ETASR), Vol. 1, No. 4, August 2011. ISSN: 1792–8036. Pages: 90–97.

[AOPJ-6]. Biswajit Bhowmik, Pooja, Piyali Sarkar, Nupur Thakur, "Received Signal Strength Based Effective Call Scheduling in Wireless Mobile Network", International Journal of Advancements in Technology (IJoAT), Vol. 2, No. 2, April 2011. ISSN: 0976–4860, Pages: 292–305.

[AOPJ-7]. Biswajit Bhowmik, Smita Roy, Parag Kumar Guha Thakurta, Arnab Sarkar, "Priority Based Hard Handoff Management Scheme for Minimizing Congestion Control in Single Traffic Wireless Mobile Networks", International Journal of Advancements in Technology (IJoAT), Vol. 2, No. 1, February 2011. ISSN: 0976–4860. Pages: 90–99.

[AOPJ-8]. Biswajit Bhowmik, Arnab Sarkar, Parag Kumar Guha Thakurta, "Simulation of Handoff Management Scheme for Improved Priority Based Call Scheduling with a Single Traffic System in Mobile Network", International Journal of

Advanced Research in Computer Science (IJARCS), Vol. 1, No. 3, September 2010. ISSN: 0976–5697. Pages: 354–358.

[AOPJ-9]. Biswajit Bhowmik, "Dynamic Programming – Its Principles, Applications, Strengths, and Limitations", International Journal of Engineering Science and Technology (IJEST), Vol. 2, No. 9, September, 2010. ISSN: 0975–5462. Pages: 4822–4826.

[AOPJ-10]. Biswajit Bhowmik Abhishek Kumar, Abhishek Kumar Jha, Rajesh Kumar Agrawal, "A New Approach of Complier Design in Context of Lexical Analyzer and Parser Generation for NextGen Languages", International Journal of Computer Applications (IJCA), Foundation of Computer Science (FCS), Vol. 6, No. 11, September 2010, ISSN: 0975–8887. Pages: 21–25.

[AOPJ-11]. Biswajit Bhowmik, "Studies on Dimultigraph and Prograph Based Applications of Graph Theory in Computer Science", International Journal of Computer & Communication Technology (IJCCT), Interscience, Vol. 1, Issue 4, August 2010. ISSN: 2231–0371. Pages: 57–61.

- ***Conference***

[AOPC-1]. Biswajit Bhowmik, "Studies on Dimultigraph and Prograph Based Applications of Graph Theory in Computer Science", 2010 International Conference on Advances in Computer, Communication Technology and Applications (ACCTA 2010), Bhubaneswar, India, August 3-5, 2010. [Acceptance Rate $\approx$ 35%].

[AOPC-2]. P. K. Guha Thakurta, Souvik Sonar, Biswajit Bhowmik, Swapan Bhattacharya, Subhansu Bandyopadhyay, "A New Approach on Priority Queue based Scheduling with Handoff Management for Mobile Network", 2010 19th ISCA International Conference on Software Engineering and Data Engineering (SEDE 2010), California, USA, June 16-18, 2010. [Acceptance Rate $\approx$ 30%].

Author's Brief-Biography

Mr. Biswajit Bhowmik received the B.Tech(Hons.) degree in Information Technology from the Vidyasagar University, Midnapore, West Bengal, India in 2004. Then he joined as Assistant Professor in the Department of Computer Science and Engineering, Bengal College of Engineering and Technology, Durgapur, India, and served the institute over six and half years. During his service period, he was a very renowned faculty member among students, academic staffs, and colleagues in this institute. He has completed his M.Tech in Computer Science and Engineering from the National Institute of Technology Durgapur, West Bengal, India in 2010. He is currently a PhD student in the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati, Assam, India. He has a consistently good academic record and maintained first division throughout. He is the *Institute Topper* at the Secondary (10th Std.) Examination, 1997 under the West Bengal Board of Secondary Education (WBBSE), Kolkata, India. He has been awarded Silver Medals twice in 1997 and 1998, respectively. He is the *Second Topper* in the department at the under graduate level (B.Tech). He is the *Second Topper* in the department and *Third Topper* in the institute at the post graduation level (M.Tech). He also stands topper, second topper positions at different years on Talent Search Programme conducted by and has been subsequently certified by many prestigious institutes like National Science Society, India; Paschim Banga Bigyan Mancha, India; and Centre for Pedagogical Studies in Mathematics, India. During his studies from school level to current PhD programme, he has received many prestigious scholarships, such as National Scholarship, Govt. of India, 1994–1997 at the school level, Indian Centre for Advancement of Research and Education (ICARE) Scholarship, 2002–2004 at Graduation level, MHRD Post Graduation Scholarship, Govt. of India, 2008–2010, and MHRD PhD Scholarship, Govt. of India, 2011–2016. His research interests specifically include 2D, 3D, Wireless, and Photonic Network-on-Chip (NoC), and broadly spread to System-on-Chip (SoC), VLSI and Embedded Systems (Design, Verification, and Testing), High-Performance Computing, Formal Verification, and Cyber-Physical Systems. He has published more than 35 research papers in many prestigious international conferences and journals. One of his research

paper has been selected as the *Best Paper* and won *Young Scientist Award* in IEEE TENCON 2015. It is glad to share that he is the *Institute First (PhD) Student* whose one research finding has been appeared in the prestigious and premier platform viz. IEEE MASCOTS 2016 and another research outcome has appeared in another premier and prestigious conference namely IEEE/ACM DATE 2017. He has recently received the *Best Paper Award* for his another research finding presented in the Research Conclave 2018 held in IIT Guwahati during March 8–11, 2018. At the same event he has won the First Prize from the Springer-Nature for the Best Paper Oral Presentation. He has received Travel Grants several times in order to attend international conferences. The Grants include IEEE SMC 2016 Student Travel Grant, ACM-IARCS Travel Grant, and MeitY Travel Grant, Govt. of India. Besides the publication of research findings at different venues, he has authored a book titled *Design and Analysis of Algorithms* whose First and Second Edition has been appeared in 2011 and 2012, respectively. He is a student member of ACM, IEEE and its various societies, such as IEEE Computer Society, IEEE Circuit and System Society, IEEE SMC Society, IEEE Young Scientist Society, IEEE Signal Processing Society. Besides, he is a member of following well known professional organizations like IACSIT, PASS, IAOE, IAS, UACEE, IAENG and its different societies, e.g., IAENG Society of Computer Science, IAENG Society of Wireless Networks, IAENG Society of Software Engineering, and IAENG Society of Artificial Intelligence. Side by side, he has reviewed many research articles for a number of peer-reviewed and refereed journals, such as MICPRO (Elsevier), JSA (Elsevier), and conferences, such IEEE SMC 2018, IEEE ANTS 2015, and been performing other professional activities, such as Technical Program Committee Member of the IEEE SMC 2018. At present, he together with his supervisors is engaged in drafting a book titled *VLSI Design, Verification, and Testing.*

"An Idea, A Thought That Can Keep Us Safe .....That Is Save Paper ..... Save Tree ..... Save Our Beautiful Planet !!!"
– Author

"A Few Ślokas"

"Uttisthata Jagrata Prapya Varannibodhata
Kshurasanna Dhara Nishita Dustayadurgama Pathah
Tat Kavayo Vadanti"
∼: Translation :∼
"Arise! Awake! Approach the great and learn.
Like the sharp edge of a razor is that path,
so the wise say–hard to tread and difficult to cross."
–Katha Upanishad

"Otho, Jago, Laksmye Na Pounchhano Paryantya,
Thamio Na"
∼: Translation :∼
"Arise, awake, and stop not till the goal is reached"
– Swami Vivekananda

"Karmany-evādhikāras Te Mā Phaleṣhu Kadāchana
Mā Karma-phala-hetur Bhūr Mā Te Saṅgo'stvakarmaṇi"
∼: Translation :∼
"You have the right to work only but never to its fruits.
Let not the fruits of action be your motive,
nor let your attachment be to inaction."
– Shrimad Bhagavad Gita, Sankhya Yoga, Verse: 47

"Om Asato Maa Sad-Gamaya,
Tamaso Maa Jyotir-Gamaya,
Mrityor-Maa Amritam Gamaya. "
∼: Translation :∼
"O Lord, Lead me from falsehood to truth,
Lead me from darkness to light,
Lead me from death to the immortality. "
–Brihadāranyaka Upanishad

Om Shāntihi, Shāntihi, Shāntihi.