Multi-stage Interconnection Networks with Fault-tolerance and Collision Solving

A DISSERTATION

Submitted in partial fulfillment of the
Requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE ENGINEERING

By

SHRUTI GARHWAL (CSE) – 041205

KUMAR VAIBHAV (CSE) – 041209

NEHA SRIVASTAVA (CSE) – 041210

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING AND INFORMATION TECHNOLOGY

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
WAKNAGHAT, SOLAN - 173215 (HP), INDIA

MAY-2008

CERTIFICATE

This is to certify that the work entitled, "MULTISTAGE INTERCONNECTION NETWORKS with Fault Tolerance & Collision Solving" submitted by Shruti Garhwal, Neha Srivastava, Kumar Vaibhav in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science Engg of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Nitin
22nd May 2008

MR NITIN

(ADVISOR)

ACKNOWLEDGEMENTS

We would like to express our deep sense of gratitude and heartiest thanks to our teacher ***Mr. Nitin,*** Senior Lecturer, Jaypee University of Information Technology for guiding us throughout this project and providing us each and every resource required to make this project a success.

Furthermore, we would like to thank all the friends who helped us directly or indirectly in project development inclusive of typical programming involved in this project.

Lastly, we would like to show our high gratitude to the entire faculty for showing us different methodologies used in our project.

Shruti Garhwal (041205)

Kumar Vaibhav (041209)

Neha Srivastava (041210)

TABLE OF CONTENTS

LIST OF FIGURES

ACRONYMS and ABBREVIATIONS

IN:	Interconnection Network
MIN:	Multi Stage Interconnection Network
GIN:	Gamma Interconnection Network
VLSI:	Very Large Scale Integration
DOT:	Double Order Tree
CSMIN:	Combining Switches Multi Stage Interconnection Networks
FCSMIN:	Fully Chained Combining Switches Multistage Interconnection Network

ABSTRACT

Multistage interconnection networks connect input devices to output devices through a number of switch stages, where each switch is a crossbar network. MINs play a major role in performance of modern parallel computers and are well suited for communications among tightly coupled system components since they offer a good balance between cost and performance. In addition, the performance of interconnection networks has a great impact on system latency, reliability and throughput of parallel systems. In the light of developments made in the field of MINs to achieve the above objectives, our work aims to design and simulate multistage interconnection networks that provide fault tolerance and collision solving. In our attempts towards this direction, we have studied and simulated various classes of multistage interconnection network. We studied and simulated a simple Cube interconnection network which is a regular network and can be implemented as a recirculating network or a multistage network with exactly one path between a given pair of source and destination. We have also focused on irregular networks such as Double tree Multistage Interconnection Networks in which we have achieved collision solving only for some source-destinations pairs for which more than one path exists in the network. DOT network gives larger throughput than any regular type of Interconnection Network because of smaller path lengths for favorite Processor- memory pair, but does not guarantees even one fault tolerance. To achieve fault tolerance we simulated partially one fault- tolerant network with multiple paths between all source destination pairs, Zeta interconnection network which makes use of multiplexers, demultiplexers and chain links to achieve high collision solving rate. Due to the use of chaining links there exists a tradeoff between cost and performance for Zeta network.

Multistage interconnection networks can be designed to achieve fault tolerance and collision solving by providing a set of disjoint paths. The previous work in this direction had proposed a fault-tolerant network called CSMIN (Combining Switches Multistage Interconnection Network) which provides two disjoint paths to guarantee one fault-tolerant and can dynamically reroute packets between these two paths with an average of one rerouting hop to solve the collision situation or a fault in the switch or communication link. To study fault tolerance capabilities of interconnection networks we simulated CSMIN using distance tag routing algorithm. This thesis proposes a more comprehensive and accurate algorithm that always generates routing tags for the two disjoint paths for every source-destination pair in the CSMIN. The 1-fault tolerant CSMIN causes the two disjoint paths to have regular distances at each stage. The proposed algorithm backtracks a packet to the previous stage and takes the other disjoint path in the event of a fault in switch or communication link or a collision in the network. The simulation results show that our algorithm provides a much better arrival and collision ratio for every source–destination pair in CSMIN. Our algorithm backtracks a packet to the previous stage and takes the other disjoint path in the event of a fault or a collision in the network. To eliminate the backtracking penalties of CSMIN, we propose a new design called Fully-chained Combining Switches Multistage Interconnection Network (FCSMIN). FCSMIN has the similar characteristics of 1-fault tolerance and two disjoint paths between any source-destination pair but it can tolerate at least one link or switch fault at each stage without backtracking. Our comparative analysis and results show that FCSMIN has the added advantages of destination tag routing, lower hardware costs than CSMIN, strong reroutability and lower pre-processing overhead.

CHAPTER 1: INTRODUCTION, MOTIVATION AND CONTRIBUTION

1.1 PROBLEM STATEMENT

A Multistage Interconnection Network (MIN) connects many input devices to output devices through a number of switch stages, where each switch is a crossbar network. The capabilities of fault-tolerance and collision solving are crucial for MINs serving the communication needs of large-scale multiprocessor systems. The basic idea for fault tolerance is to provide multiple paths between an input-output pair, so that the alternate paths can be used in case of a fault in a switching element or a communication link in the exiting routing path. The feature of effective collision solving is guided by the efficiency of a routing algorithm or the topology of the MIN to reroute packets on an alternative path when two or more packets are in conflict for the use of a resource such as a switching element or a communication link in the existing routing path. This efficiency of rerouting enables a MIN to reduce packets lost in the network due to collisions and thus increase system throughput. Designing a suitable efficient MIN that meets the above necessities has become an important issue in multiprocessor systems because overall system performance relies on the interconnection network.

Many prior researches in this direction have resulted in the development of many regular and irregular networks with the help of chaining links, addition of extra stages or modifications in routing algorithms. However, these works suffer from the disadvantage of not being able to always provide alternative paths for a packet transferred between any input-output pair in cases of a fault or a conflict in the use of a switching element or a communication link. Any fault or collision in the existing routing path should not make the alternative rerouting path cease to exist. This emphasizes the need for the two paths to be disjoint in nature, that is, they should not have any common communication link or switching element whose busy status or failure will make both the paths cease to exist for routing in the network and thus result in packet loss.

Hence, the lack of suitable designs and algorithms that effectively meet the objectives of fault tolerance and collision solving of a MIN for any input-output pair has pushed us to design and simulate a new cost-effective, reliable, fault-tolerant multistage interconnection network.

1.2 MOTIVATION

Interconnection Networks (IN) are currently being used for many different applications, ranging from internal buses in Very Large–Scale Integration (VLSI) circuits to wide area computer networks. A Multistage Interconnection Network (MIN) connects input devices to output devices through a number of switch stages, where each switch is a crossbar network. The number of stages and the connection patterns between stages determine the routing capability of the networks. The lack of standards and the need for very high performance and reliability pushed the development of MIN for parallel computers with hundred of processors and some commercial machines. Since the assurance of high reliability is a significant task in complex systems, fault-tolerance is crucial for MINs to serve the communication needs. In the absence of faults the most important performance metrics of a MIN are system latency and throughput.

The fault-tolerance capability in a MIN guarantees that a packet will have an alternative routing path if it encounters a faulty or busy switch or a communication link in its existing routing path. A MIN is fully able to meet the reliability demands if it is at least one fault tolerant, that is, there is at least one alternative path to deal with faults or collisions. This alternative path should be disjoint in nature with the existing routing path followed, that is, there is no common communication link or switching element in the two paths so that there is no such implication that if a switch or a link fails in the existing routing path then the alternative path will also fail. The performance of a MIN in terms of its throughput is highly dependent on its collision solving ability. A MIN should be to reroute packets on an alternative path when two or more packets are in conflict for the use of a resource such as a switching element or a communication link in the existing routing path. Lesser the number of packets lost due to collision, better is its efficiency in solving collisions. Better the collision solving ability, better the performance.

With the aim to achieve the above objectives of fault tolerance and collision solving, we attempt to design and simulate a MIN that is at least 1-fault tolerant and has a high rate of collision solving.

Many prior researches and developments have been made in this direction. Many designs and routing algorithms for MINs have been put forth to effectively deal with faults and collisions in the network. The nature of these designs and algorithms has been characterized to either compromise, balance or optimize, all, any or some of the following factors such as cost-effectiveness, reliability, throughput, communication delay, pre-processing overhead and memory capacity. When our work was done, many approaches existed that led to the design of several regular, irregular and hybrid MINs. These classes of MINs exploited the topology of a MIN in the following ways:

- Changing the number of switching elements at each stage
- Adding or removing extra stages
- Changing the nature of communication links from straight to non-straight upward or downward
- Introducing buffer in the switching elements
- Introducing a centralized controller in the form of additional circuitry for the control logic
- Introducing chaining links in some or all stages
- Introducing multiplexers and demultiplexers in stages 0 and n
- Combining the topologies of two or more MINs

Many significant changes have also been made in the routing schemes adopted for MINs with aim of minimizing latency, easy rerouting, a decrease in pre-processing overhead.

Previous approaches or solutions were mostly blocking in nature. They always resulted in high rates of packet losses due to collisions or faults. Some regular networks like Cube Interconnection Network provided only one path for routing packets between any source and destination node. If this path failed, no other path existed to route the packets and hence the packet was lost resulting in performance degradation. Some irregular networks like Double Order Tree Interconnection Network (DoT) provided more than one path of different path lengths for some source-destination pair. However, some cases of source-destination pair existed, for which only one path existed. This one path had only one switching element in its middlemost stage and whose failure could result in a choking condition. There were also other approaches like Zeta Interconnection Networks which used multiplexers, demultiplexers and chaining links in an attempt to provide fault tolerance. However, these approaches were only fault tolerant for some cases. Then, there were some MINs like Gamma Interconnection Networks (GIN) which were one fault tolerant. Although GIN provided two sets of path to deal with a faulty or busy switch or link, these paths were not disjoint in nature because when the distance between the source and the target is even, the straight link between stage 0 and stage 1 and the switch at stage 1 connected by the straight link is the common element contained in these paths. Furthermore, Gamma networks have only one single path when the indices of the source and the target are the same.

To address the problems of both performance and fault-tolerant capability, our approach was to design and simulate a one fault tolerant network with the following issues in mind:

- Guarantee of at least two disjoint paths
- Easy rerouting between disjoint paths
- Keep low rerouting hops
- Solve the occurrences of packet collisions.

We have considered the design of a fault tolerant network called Combining Switches Multistage Interconnection Network (CSMIN) and applied the above issues. Though, this design proposed by Ching-Wen Chen and Chung-Ping Chung made an impressive attempt in meeting the above requirements but it had an inaccurate routing algorithm that did not always generate correct disjoint paths for every source-destination pair. We have smartly modified the algorithm and provided a more accurate and comprehensive approach that always generates correct routing tags for the two disjoint paths for every source–destination pair in the CSMIN. The 1–fault tolerant CSMIN causes the two disjoint paths to have regular distances at each stage. Moreover, our algorithm backtracks, a packet to the previous stage and takes the other disjoint path in the event of a fault or a collision in the network. Thus, our approach guarantees 1-fault tolerance for all cases by providing two disjoint paths. Moreover, an easy rerouting algorithm is provided that simply puts the packets on other disjoint path to deal with faults or collision. This rerouting algorithm makes use of backtracking mechanism with a very low average rerouting hop of one. Hence, one can analyze our simulation results in terms of arrival and collision ratio and decide whether our approach is better than others. This work is important since it provides our modified distance tag routing algorithm that can be applied to other MINs as well.

However, neither any system is 100% efficient nor any approach or solution is wrong. With the aim of refining our methodology applied to CSMIN, we looked for improvements in our approach. The algorithm and design analysis of CSMIN presented us with one such opportunity. We discovered that our proposed algorithm for CSMIN backtracked a packet to the previous stage and then took the other disjoint path in order to tolerate a switch or a link fault in the network. This backtracking mechanism not only increases the system latency but also increases the hardware cost as bi-directional switches are used in CSMIN for stages 1 to n to achieve reroutability through backtracking. In our search for solutions to the above problems that we encountered, we came out with an altogether new design called Fully-chained Combining Switches Multistage Interconnection Network (FCSMIN). FCSMIN has the similar characteristics of 1-fault tolerance and two disjoint paths between any source-destination pair but it can tolerate at least one link or switch fault at each stage without backtracking by making use of destination tag routing algorithm for stages 1 to n. For stages 1 to n chain links are added between nodes belonging to a neighboring group at the same stage. When a link fault occurs at a stage in FCSMIN, the chain link is taken.

We have a competitive world in which we are always looking for new questions in the hidden or grey places and answers to the unanswered questions. We also have a selfish mind which leads us to make attempts to override the approach suggested by another. This has been our guiding motto which led us to suggest an accurate algorithm for CSMIN and a new design by introducing chaining links in CSMIN to reduce hardware costs and better the performance. We strongly believe in the guiding motto cited above and we expect others to follow the same. Hence, looking back on our words, we also hold the belief that our work is open to further improvements or refinements in future.

1.3 CONTRIBUTION

Our approach of providing a pair of disjoint paths for every-source and destination pair in CSMIN has been successfully recognized in the form of a research paper being accepted for publication in the 2008 World Congress in Computer Science, Computer Engineering and Applied Computing (WORLDCOMP'08: July 14-17, 2008). This paper is currently in press in the category of a regular research paper in the arena of Parallel and Distributed Processing Techniques and Applications (PDPTA) for the conference to be held in Las Vegas, Nevada, USA.

Multistage interconnection networks can be designed to achieve fault tolerance and collision solving by providing a set of disjoint paths. Ching–Wen Chen and Chung–Ping Chung had proposed a fault–tolerant network called Combining Switches Multistage Interconnection

Network (CSMIN) and an inaccurate algorithm that provided two correct disjoint paths only for some source–destination pairs. Our paper provided a more comprehensive and accurate algorithm that always generates correct routing tags for the two disjoint paths for every source–destination pair in the CSMIN. The 1–fault tolerant CSMIN causes the two disjoint paths to have regular distances at each stage. Moreover, our algorithm backtracks, a packet to the previous stage and takes the other disjoint path in the event of a fault or a collision in the network. The simulation results of our algorithm show the arrival and collision ratio for every source–destination pair in CSMIN.

Our work on CSMIN makes use of distance tag routing algorithm to compute the routing tags of the two disjoint paths. But this algorithm backtracks a packet to the previous stage and then takes the other disjoint path in order to tolerate a switch or a link fault in the network. To eliminate the backtracking penalties of CSMIN, we have also proposed a new design called Fully-chained Combining Switches Multistage Interconnection Network (FCSMIN). FCSMIN has the similar characteristics of 1-fault tolerance and two disjoint paths between any source-destination pair but it can tolerate at least one link or switch fault at each stage without backtracking. FCSMIN also has the added advantages of destination tag routing, lower hardware costs than CSMIN, strong reroutability and lower pre-processing overhead. We have registered this study of FCSMIN in the form of a research paper for the IEEE TENCON 2008 conference to be held in Hyderabad, India from November 18 to 21.

1.4 ORGANIZATION OF THESIS

Our B.Tech. Major Project thesis is divided into three Chapters namely Introduction, Literature Survey & "CSMIN Revisited: Accurate algorithm and strategic design issues". Chapter 1 deals with the overview of the dissertation which is in the form of the problem statement, motivation and the major contribution of our work. In Chapter 2, we provide our literature survey of multistage interconnection networks, which includes detailed analysis on various types of MINs, their functionality and capabilities. Chapter 3 covers our research on CSMIN, its accurate algorithm and strategic design issues. An accurate and more comprehensive algorithm to generate routing tags for two disjoint paths for any source-destination pair in CSMIN and an alternative design of CSMIN with chaining has been proposed. There is also a conclusion section. We have also provided the codes for simulation of Cube, DoT, Zeta and CSMIN in a CD at the end of the thesis.

CHAPTER 2: LITERATURE SURVEY

2.1 MULTISTAGE INTERCONNECTION NETWORKS

The interconnection network plays a central role in determining the overall performance of a multi-computer system. If the network cannot provide adequate performance, for a particular application, nodes will frequently be forced to wait for data to arrive. With interconnection networks it is possible to design a network in such a way that there are several independent paths between two modules being connected which increases the available bandwidth. Interconnection networks may be used not only to connect memory modules but also to interconnect computers. Such an interconnected set of computers is a parallel computer, often called distributed memory parallel computer.

2.1.1 *Reliability*

This is a very important parameter in judging the performance of a multi stage interconnection network. This factor indicates the ability of the network to be able to transfer the packet even in the case in which one or more faults occur in the network. This is implemented by having multiple paths for transfer of the packets between a source and destination pair. When these paths are non intersecting and independent of each other then they provide reliability to the network. More the number of distinct, non intersecting paths more would be reliability of the network. The actual use of these paths is in case of faults the packet is rerouted on the other path hence avoiding a failed transfer state.

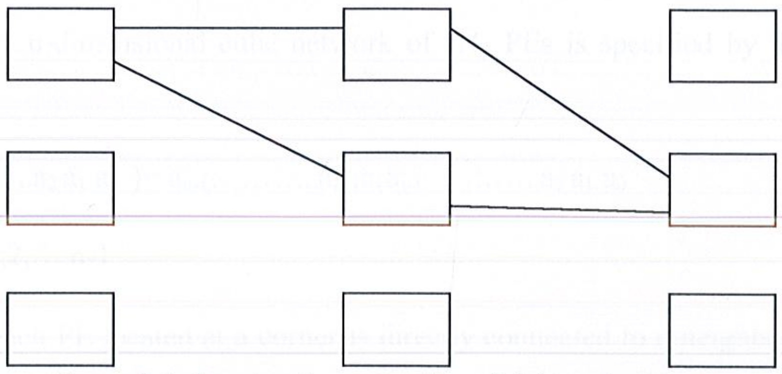


Figure 2.1 showing the mutually reliable paths in a network

2.2 CUBE INTERCONNECTION NETWORK

The Cube Network can be implemented as either a recirculating network or as a multistage network.

A 3- dimensional cube is illustrated in fig below

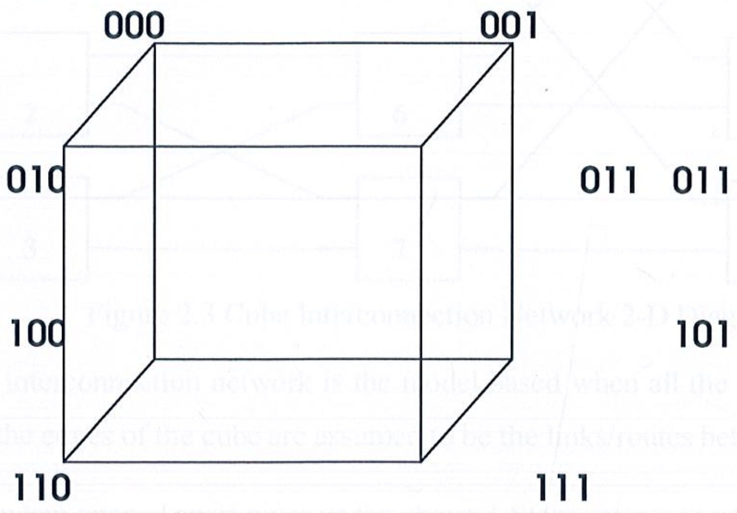


Figure 2.2 Cube Network Architecture

Vertical lines connect vertical PEs whose addresses differ in the most significant bit position. Vertices at both the ends of the diagonal lines differ in the middle bit position. Horizontal lines differ in Least significant bit position .This unit cube concept can be extended to an n-dimensional unit space, called an n-cube with n bits per vertex, we shall use binary sequence

$A=(a_{n-1}\ldots\ldots\ldots\ldots a_2\, a_1\, a_0)_2$,to represent the vertex (PE) address for $0<=i<n-1$

Formally, an n-dimensional cube network of N PEs is specified by the following n routing function,

$C(a_{n-1}\ldots\ldots\ldots\ldots a_2\, a_1\, a_0\ )= a_{n-1}\ldots\ldots\ldots\ldots a_{i+1} a_i\, a_{i-1}\ \ldots\ldots\ldots\ldots a_2\, a_1\, a_0$

for $i=0,1,2,\ldots.n-1$

In n-cube ,each PE located at a corner is directly connected to n neighbors. The neighbors PEs differ in exactly 1 bit position

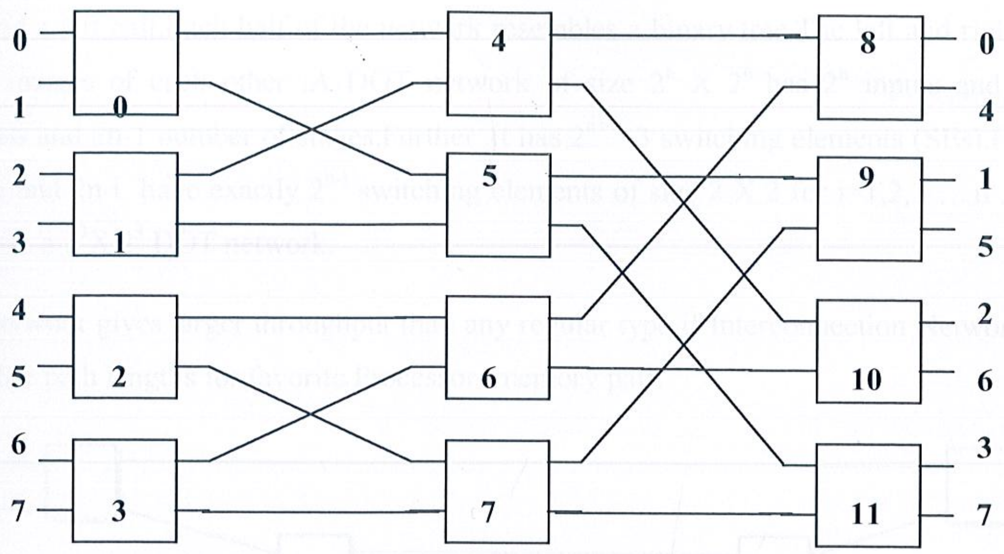


Figure 2.3 Cube Interconnection Network 2-D Diagram

A single cube interconnection network is the model based when all the PEs are taken to be part of a cube and the edges of the cube are assumed to be the links/routes between them.

This structure when opened apart gives us the above 3-Stage interconnection network.

2.2.1 *Salient features of Cube Interconnection Network*

- Exactly one path exists from a particular source to destination, there are no alternative paths if a route is busy or broken.
- Although the number of PEs is 8 but a maximum of only 4 data packets can be sent successfully over the network in parallel without collision.
- The way to reach a destination follows a pattern irrespective of the source from which the packet is coming.
- Two function switching elements have been used, straight and exchange are the two functions which are allowed in the SEs.

2.3 DOUBLE TREE NETWORK

A double tree network is an irregular type of Multistage interconnection network.It consist of a right and a left half.Each half of the network resembles a binary tree.The left and right trees are mirror images of each other .A DOT network of size 2^n X 2^n has 2^n inputs and 2^n output terminals and 2n-1 number of stages.Further ,it has 2^{n+1} -3 switching elements (SEs).Each of the stages i and 2n-i have exactly 2^{n-i} switching elements of size 2 X 2 for i=1,2,3…..n .Fig below illustrates a 2^3X 2^3 DOT network.

DOT network gives larger throughput than any regular type if Interconnection Network because of smaller path lengths for favorite Processor- memory pair.

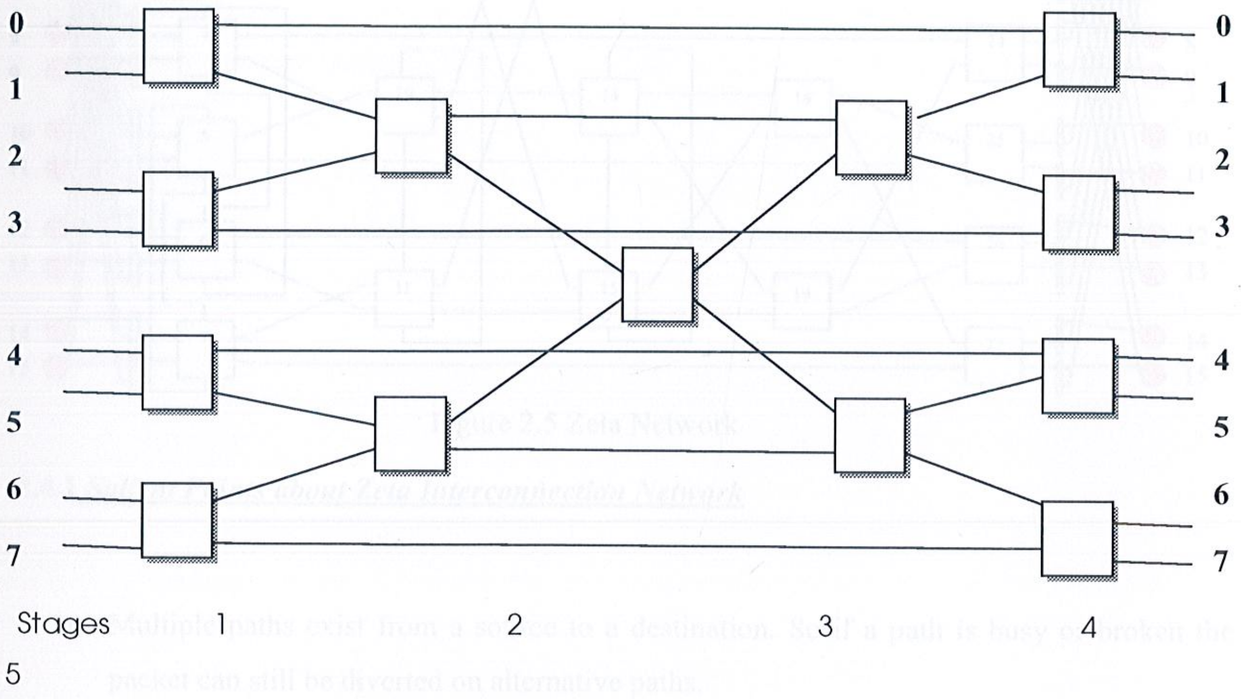


Figure 2.4 DOT Network Diagram

2.3.1 *Disadvantages of Double Tree Network*

- Though a redundant network but not even a single fault tolerant
- Longer path lengths (i.e more communication delays)
- Less bandwidth
- High probability of contention in a path leading to unfavorite memory modules.

2.4 ZETA NETWORK

An irregular multistage interconnection network with chaining links

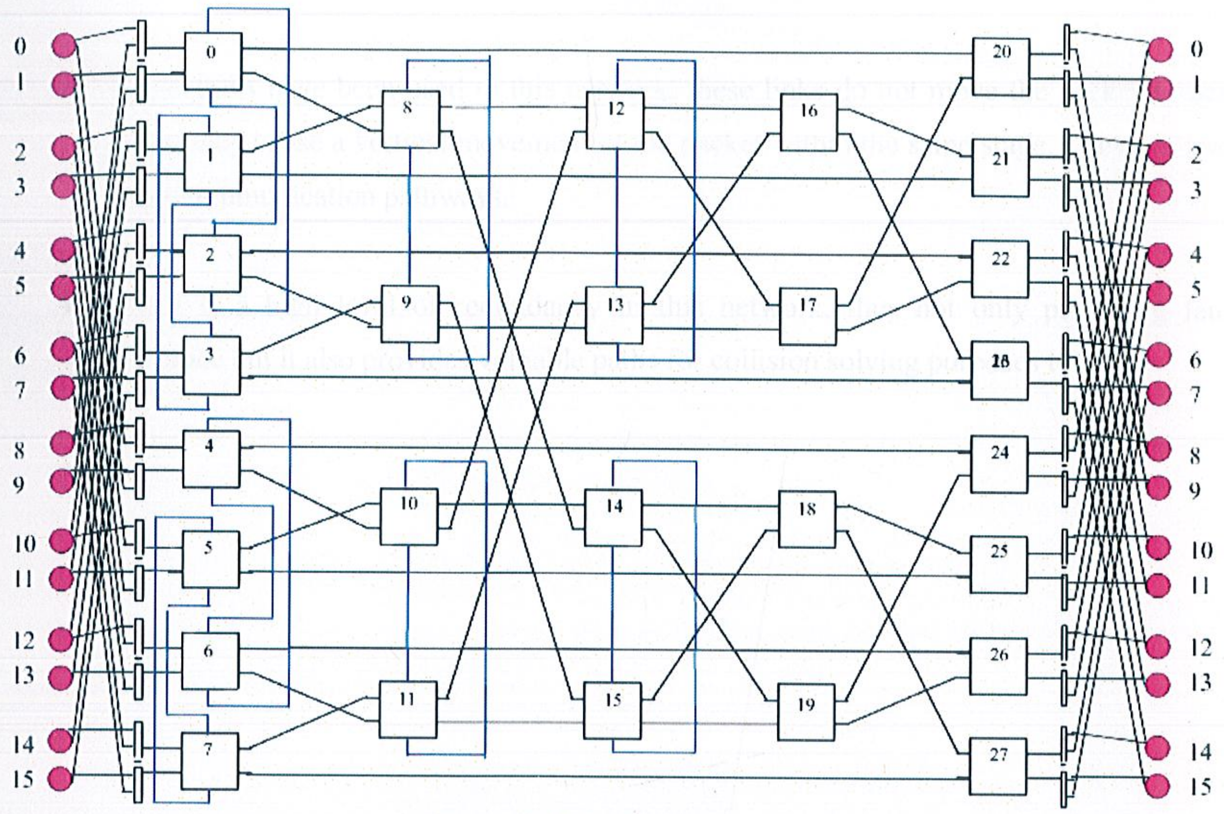


Figure 2.5 Zeta Network

2.4.1 *Salient Points about Zeta Interconnection Network*

- Multiple paths exist from a source to a destination. So if a path is busy or broken the packet can still be diverted on alternative paths.
- Although the number of Sources is 16 but a maximum of 8 packets can be sent without collision at the same time.
- Zeta network is completely one fault tolerant till stage 2 after that fault tolerance may or may not be present depending on the source and destination values of the packet.

- Multiplexers and demultiplexers have been used in this network so that fault tolerance is present at entry and exit stage to the network.

- Chain links have been used in this network, these links do not move the packet to next stage, they cause a vertical movement of the packet within the same stage. They are intra stage communication pathways.

- There is a high level of redundancy in this network, thus not only providing fault tolerance but it also provides valuable paths for collision solving purposes too.

2.5GAMMA INTERCONNECTION NETWORK

The Gamma interconnection network (GIN) uses 3 × 3 switches to enhance fault tolerance capabilities. The GIN provides multiple paths between any source-target pair except when the indices of the source and target are the same. To improve the fault-tolerant capability of GIN, some researchers modified GIN to provide multiple paths between any source-target pair. However, the method of adding extra stage results in more hardware cost and more collisions.

2.5.1 Topology

A Gamma network of $N = 2^n$ consists $n + 1$ stages labeled from $0\ to\ n$. Each stage has N switches labeled from $0\ to\ 2^n - 1$. The switch architecture at the first and the last stage has 1×3 and 3×1 crossbars, respectively, and switches located at the intermediate stages have 3×3 crossbars. A switch labeled j at stage i has three output links connecting to switches at stage $i + 1$ based on the plus-minus 2^i function; that is, the jth switch at stage i has three output links connected to switches $[(j - 2^i) mod\ N], j$ and $[(j + 2^i) mod\ N]$at the next stage. Figure (1) below illustrates a GIN network of size 8

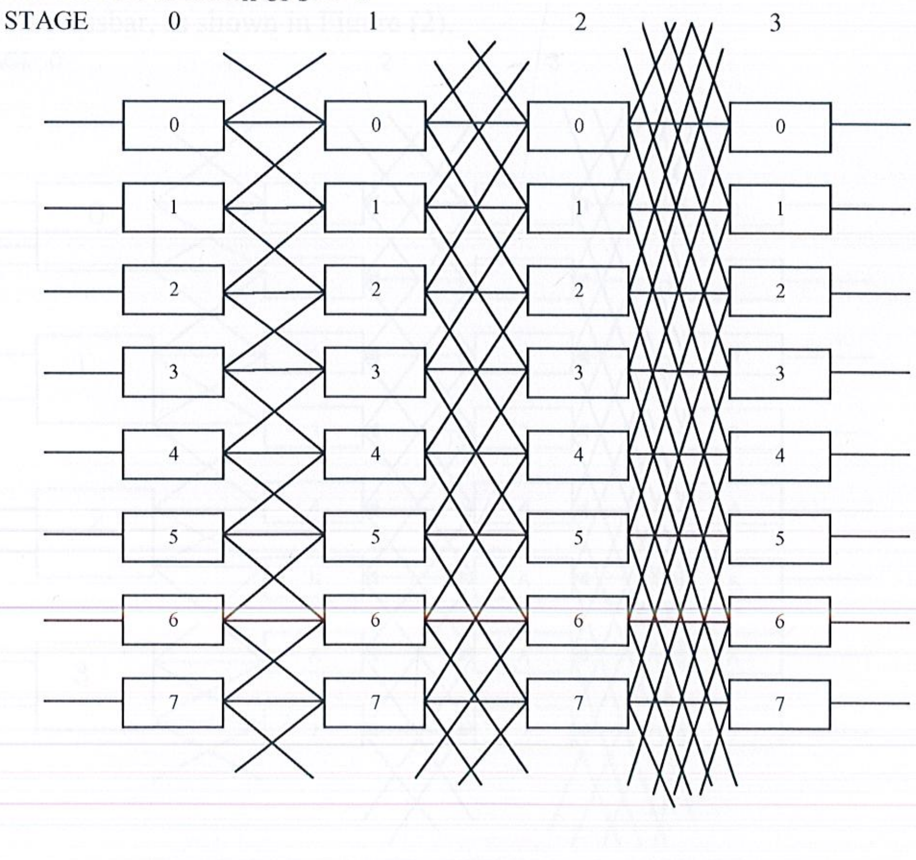


Figure 2.6 Gamma Interconnection Network of size 8

2.6COMBINING SWITCHES MULTISTAGE INTERCONNECTION NETWORKS

Combining Switches Multistage Interconnection Network (CSMIN) guarantees one fault-tolerance by providing two disjoint paths for any source and target pair. Besides, the CSMIN can also reroute a packet between these two disjoint paths at each stage when a packets encounters a faulty or busy element, with an average of one rerouting hop.

2.6.1 *Topology of CSMIN*

The topology of CSMIN is described as follows:

1. At stage 0, switch $2i$ and switch $2i + 1$ are coupled into a 2×4 switch, for $i = 0\ to\ (\frac{N}{2} - 1)$

2. All straight links between stage 1 and stage n are bi-directional.

3. All the rest remain the same as those in GIN.

In CSMIN, the sizes of switches of the first and last stages are 2×4 and 3×2, respectively. Switches at stage 1 have 3×3 crossbars. Moreover, each switch located at the intermediate stages has a 4×4 crossbar, as shown in Figure (2).

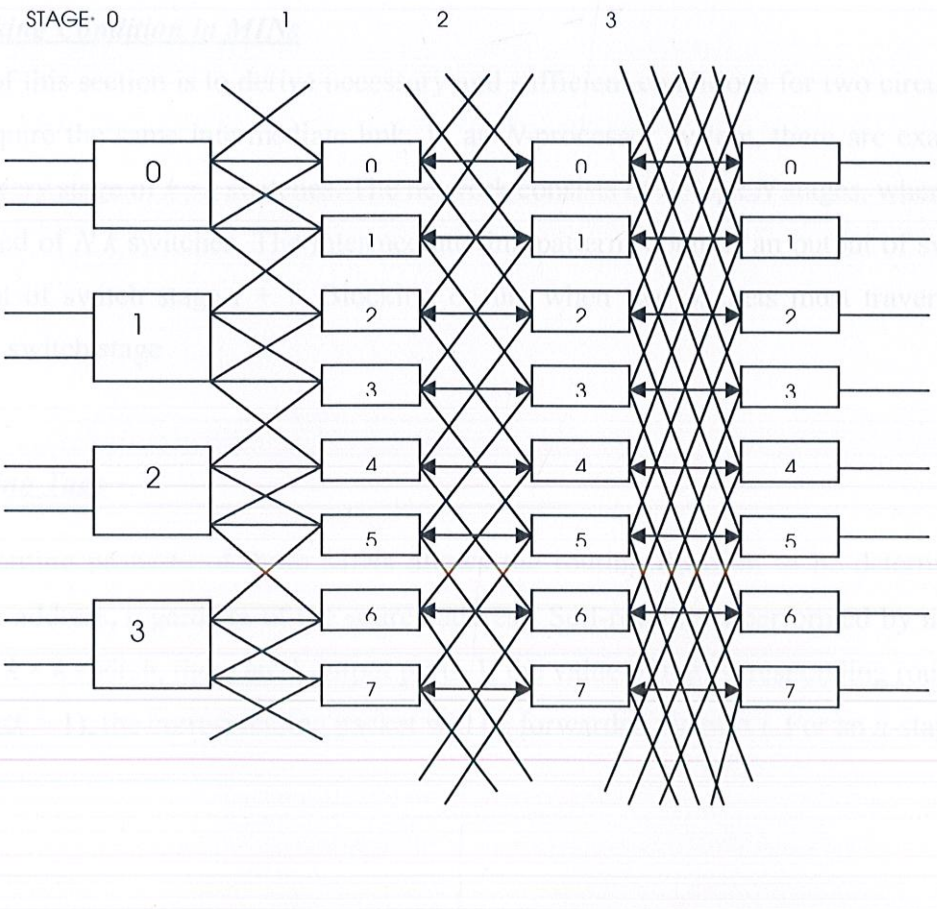


Figure 2.7 Combining Switches Multistage Interconnection Network of size 8

2.7 ROUTING TAG ALGORITHMS

In this section, we describe several issues concerning routing in MINs. For array processors, a central controller establishes the path from input to output. In cases where the number of inputs equals the number of outputs, each input synchronously transmits a message to one output, and each output receives a message from exactly one input. Computing the switch settings to realize such a permutation is a complex task. Furthermore, some permutations may not be realizable in one pass through the network. In this case, multiple passes of the data through the network may be required, and the goal is to minimize the number of passes. The complexity of the off-line computation of the switch settings is proportional to the number of switches.

In asynchronous multiprocessors, centralized control and permutation routing are infeasible. So, a routing algorithm is required to establish the path across the stages of a MIN. The simplest solution consists of using source routing. In this case, the source node specifies the complete path As this solution produces a considerable overhead, we will focus on distributed routing.

2.7.1*Blocking Condition in MINs*

The goal of this section is to derive necessary and sufficient conditions for two circuits to block, that is, require the same intermediate link. In an *N*-processor system, there are exactly *N* links between every stage of $k \times k$ switches. The network consists of $n = \log k\ N$ stages, where each stage is comprised of $N\ k$ switches. The intermediate link patterns connect an output of switch stage i to an input of switch stage $i + 1$. Blocking occurs when two packets must traverse the same output at a switch stage

2.7.2*Routing Tags*

The self-routing property of these MINs allows the routing decision to be determined by the destination address, regardless of the source address. Self-routing is performed by using routing tags. For a $k \times k$ switch, there are k output ports. If the value of the corresponding routing tag is i ($0 \leq i \leq k - 1$), the corresponding packet will be forwarded via port i. For an n-stage

MIN, the routing tag is $T = tn-1 \ldots t1t0$, where ti controls the switch at stage Gi . each switch is only able to change the least significant digit of the current address. Therefore, routing tags will take into account which digit is the least significant one at each stage, replacing it by the corresponding digit of the destination address.

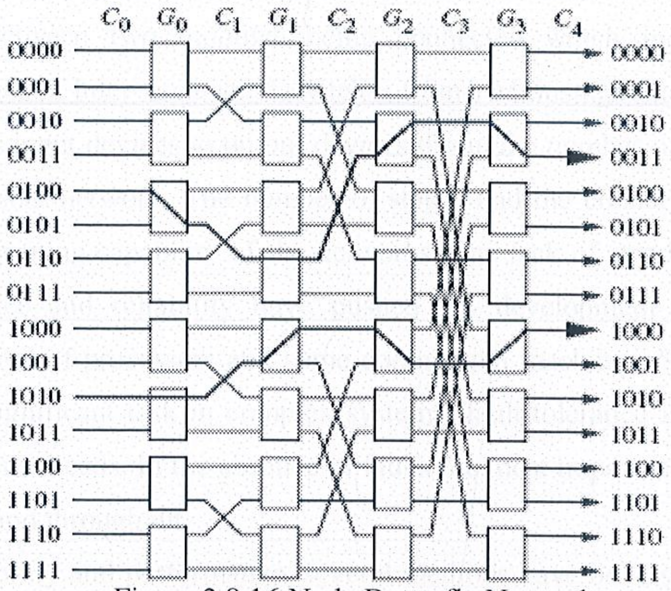


Figure 2.8 16 Node Butterfly Network

Above figure shows a 16-node butterfly MIN using 2×2 switches, and the paths followed by packets from node 0100 to node 0011, and from node 1010 to node 1000.As indicated above, the routing tag for a given destination $dn-1dn-2 \ldots d0$ is formed by having $ti = di+1$ for $0 \leq i \leq n-2$ and $tn-1 = d0$. Thus, the routing tag for destination 0011 is 1001. This tag indicates that the packet must take the upper switch output (port 0) at stages $G2$ and $G1$ and the lower switch output (port 1) at stages $G3$ and $G0$. The routing tag for destination 1000 is 0100. This tag indicates that the packet must take the upper switch output at stages $G3$, $G1$, and $G0$ and the lower output at stage $G2$.

CHAPTER 3: CSMIN REVISITED: ACCURATE ALGORITHMS AND STRATEGIC DEIGN ISSUES

3.1 INTRODUCTION

Interconnection Networks (IN) are used to design a network in which there are several independent paths between two modules being connected which increases the available bandwidth. Many stages of Inter–connected switches form a Multistage Interconnection Network (MIN). MIN connects input devices to output devices through a number of switch stages, where each switch is a crossbar network. The number of stages and the connection patterns between stages determine the routing capability of the networks. The lack of standards and the need for very high performance and reliability have pushed the development of MIN for parallel computers with hundred of processors and some commercial machines. Since the assurance of high reliability is a significant task in complex systems, fault-tolerance is crucial for MINs to serve the communication needs. In the absence of faults the most important performance metrics of a MIN are latency and throughput.

For high reliability and performance, several methods have been suggested that provide fault tolerance and collision solving to MINs. The basic idea for fault tolerance is to provide multiple paths for a source– destination pair, so that alternate paths can be used in case of a fault in the path. However to guarantee 1–fault tolerance, a network should have a pair of alternate paths for every source destination pair which are disjoint in nature. The alternative path should be disjoint in nature with the existing routing path followed so that there is no such implication that if a switch or a link fails in the existing routing path then the alternative path will also fail. Most design of multistage interconnection networks do not generate at least two disjoint paths and hence are not always fault tolerant resulting in packet losses and eventual performance degradation. Hence, this approach of two disjoint paths will always guarantee a way out of the problem of faults or collisions in a network.

The previous work in this direction by Ching–Wen Chen and Chung–Ping Chung in had proposed a fault–tolerant network called Combing Switches Multistage Interconnection Network (CSMIN) and an incorrect algorithm that did not provide two correct disjoint paths for some source–destination pairs. Their work did not generate correct routing tags for some source destination pairs. The routing tags generated for these source destination pairs were not correct in the sense that the resulting two disjoint paths in CSMIN for the desired destination did not reach the desired destination. This paper provides a more comprehensive and accurate algorithm that always generates correct routing tags for every source–destination pair in the CSMIN so that the resulting two disjoint paths reach the desired destination. Our algorithm can also dynamically reroute packets between these two paths to solve the faults or collision situation for every source destination pair in CSMIN.

One fault tolerant CSMIN provides two set of disjoint paths for every source destination pair which are at regular distance from each other in each stage. Our work on CSMIN makes use of distance tag routing algorithm to compute the routing tags of the two disjoint paths. But this algorithm backtracks a packet to the previous stage and then takes the other disjoint path in order to tolerate a switch or a link fault in the network. This backtracking mechanism not only increases the system latency but also increases the hardware cost as bi-directional switches are used in CSMIN for stages 1 to n to achieve reroutability through backtracking.

In this chapter, we have also proposed a new design called Fully- chained Combining Switches Multistage Interconnection Network (FCSMIN) that make use of destination tag routing for stages 1 to n to overcome the backtracking problem in CSMIN. FCSMIN has the similar characteristics of 1-fault tolerance and two disjoint paths between any source-destination pair but it can tolerate at least one link or switch fault at each stage without backtracking. For stages 1 to n chain links are added between nodes belonging to a neighbouring group at the same stage. When a link fault occurs at a stage in FCSMIN, the chain link is taken. We also introduce two new destination tag routing functions, *UpRoute* and *DownRoute* which can be used to find two disjoint paths in FCSMIN.

The rest of this chapter is divided into 13 sections. Section 2 provides an insight into the topology and the salient features of the 1–fault tolerant CSMIN. Section 3 covers our proposed algorithm that provides two disjoint paths for every source destination pair. In Section 4, we illustrate routing in a CSMIN. Section 5 covers dynamic rerouting between the two disjoint paths to solve collisions or faults for every packet. In section 6, we illustrate dynamic rerouting using the algorithm in Section 5. We show the simulation results of CSMIN in Section 7 and provide the screen shots of its simulation in Section 8. Section 9 deals with algorithm and design issues in CSMIN while Section 10 covers our proposed design of FCSMIN and the routing scheme is provided in Section 11. In Section 12, we present a comparative analysis of FCSMIN over CSMIN followed by the conclusion. We present our results and discussions in Section 13.

3.2 TOPOLOGY OF COMBINING SWITCHES MULTISTAGE INTERCONNECTION NETWORKS

A CSMIN of size $N = 2^n$ consists of $n+1$ stages labeled from 0 to n At stage 0, switch $2i$ and switch $2i+1$ are coupled into a 2 × 4 switch, for $i = 0$ to $(N/2 - 1)$. Stage 1 to Stage n have N switches labeled from 0 to $2^n - 1$. All straight links between stage 1 and stage *n* are bi-directional .The switch architecture at the first and the last stages has 2 × 4 and 3 × 2 crossbars respectively .Switches located at stage 1 have 3 × 3 crossbars. Moreover, each switch located at the intermediate stages has a 4 × 4 crossbar switch. Figure (3.1) illustrates a CSMIN of size 8.

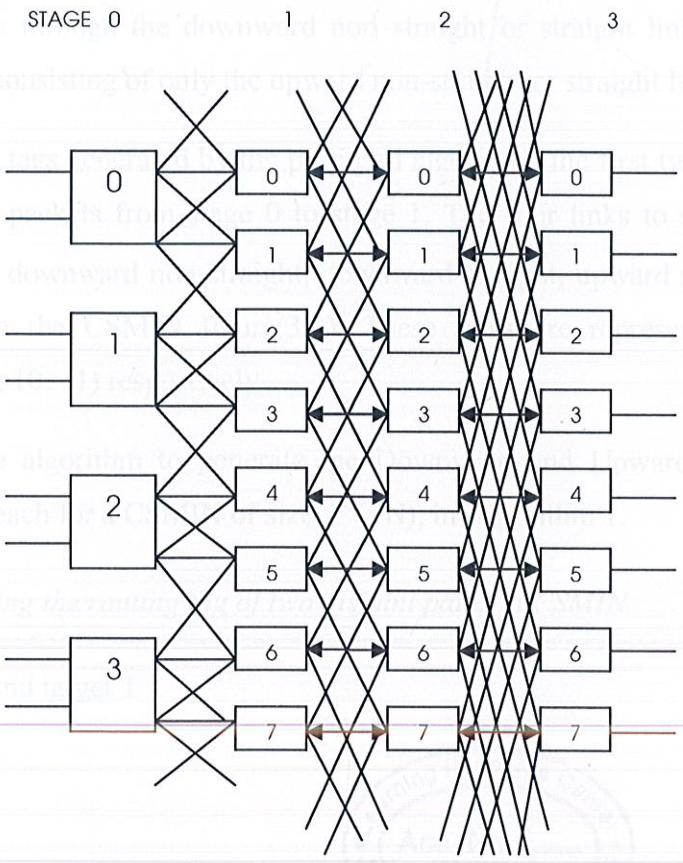


Figure 3.1 Combining switches Multistage Interconnection Network

3.3 ALGORITHM TO GENERATE ROUTING TAGS OF TWO DISJOINT PATHS IN CSMIN

The previous works on CSMIN have used routing tags to identify the two disjoint paths for a source destination pair. However the earlier algorithms generated incorrect routing tags of the two disjoint paths for some source destination pairs. Our proposed algorithm provides comprehensive and accurate routing tags for all source destination pairs, so that CSMIN is completely 1-fault tolerant.

Our algorithm exploits the topology of CSMIN so that there exists two paths between any source-destination pair, which always have a regular vertical distance of 2^i at each stage i, where $1 \le i \le n-1$, so that the two paths are always disjoint in nature. We define two distance tag routing functions as downward and upward. The path generated by the downward function always goes through the downward non–straight or straight links only. The upward function is the route consisting of only the upward non-straight or straight link.

In the routing tags generated by the proposed algorithm, the first two routing tag bits are used for transferring packets from stage 0 to stage 1. The four links to stage 1 from a 2×4 switch in stage 0 are downward non-straight, downward straight, upward straight, upward non-straight as shown in the CSMIN figure(3.1). These links are represented by the two bit combinations (00, 01, 10, 11) respectively.

We describe the algorithm to generate the Downward and Upward routing tags which consist of $n+1$ bits each for a CSMIN of size 2^n(=N), in Algorithm 1.

Algorithm1: Generating the routing tag of two disjoint paths in CSMIN

```
1.Input the source S and target T
2.Let S1=S and T1=T
3.If T-S is even
    If S is even
```

```
        S=S+1
        Else
        S=S-1
    End If
4.Let distance DownwardD=(T-S)ModN and UpwardD=(N-(T-S))Mod N.
5.Convert DownwardD to its binary representation $c_0c_1c_2\ldots c_{n-1}$
6.Convert UpwardD to its binary representation $b_0b_1b_2\ldots b_{n-1}$ and replace any 1 with 2
7. If $c_0=0$ and S is odd
      Replace $c_0$ $c_1$ with 10 and $b_0$ $b_1$ with 00
    Else
     If $c_0=0$ and S is even
       Replace $c_0$ $c_1$ with 11 and $b_0$ $b_1$ with 01
    End If
8. If $c_0=1$ and S is odd
      Replace $c_0$ $c_1$ with 11 and $b_0$ $b_1$ with 01
    Else
      If $c_0=1$ and S is even
        Replace $c_0$ $c_1$ with 10 and $b_0$ $b_1$ with 00
    End If
11. For Downward tag $c_0c_1c_2\ldots c_{n-1}$ &
      For Upward tag $b_0b_1b_2\ldots b_{n-1}$
```

```
If s1=0/1
 If t1=1/2/3/4
  If $c_{n-1}$=1
   $c_{n-1}$=0
  Else
   If t1=0/5/6/7
    If $b_{n-1}$=2
     $b_{n-1}$=0
End If


If s1=2/3
 If t1=0/1/2/7
  If $b_{n-1}$=2
   $b_{n-1}$=0
Else
  If t1=3/4/5/6
   If $c_{n-1}$=1
   $c_{n-1}$=0
End If


If s1=4/5
If  t1=1/2/3/4
 If $b_{n-1}$=2
  $b_{n-1}$=0
 Else
  If t1=0/5/6/7
   If $c_{n-1}$=1
    $c_{n-1}$=0
End If

If s1=6/7
 If  t1=0/1/2/7
  If $c_{n-1}$=1
   $c_{n-1}$=0
  Else
   If t1=3/4/5/6
    If $b_{n-1}$=2
     $b_{n-1}$=0
End If
```

12. Display Downward and Upward tag.

In the Downward and Upward routing tags for CSMIN of size 8, the first two routing tag bits represent transfer from stage 0 to stage 1.The third routing tag bit represents stage 1 to stage 2 transfer and the fourth routing tag bit represent stage 2 to stage 3 transfer .The Downward function is characterized by motion either in the Straight or Downward direction, and the Upward function goes upward or straight. In the case of transfer from stage 1 to stage 2 and transfer from stage 2 to stage 3, the routing bits for downward non–straight, straight, upward non–straight are denoted by 1, 0, 2 respectively.

3. 4 ROUTING IN CSMIN

In this section, we describe the routing behavior in CSMIN. If a packet does not encounter a faulty or busy element, distance tag routing is applied in CSMIN; that is, we compute the routing tags, the Downward and Upward tags, and then one of the two routing tags is used to send packets to the destination. Example 1 illustrates the routing path for the source S=5 and the destination T=5 with size N=8.

Example 1. If source S=5, destination T=5 and network size N=8, the routing situation in CSMIN is as shown in Figure (3.2). Following algorithm 1 for S=5 and T=5 we have

1. S1=5 and T1=5
2. S=4 since T–S is even and S is odd
3. DownwardD=(5–4)mod8=1=0001 and UpwardD=(8–(5–4))mod8=7=0111=0222
4. DownwardD=1101 and UpwardD=0122 since c0=0 and S is even
5. DownwardD=1001 and UpwardD=0022 since c0=1 and S is even
6. DownwardD=1000 since S1=5 and T1=5

The Downward and Upward routing tags, thus, generated from algorithm 1 are 1000 and 0022 respectively. Take one of these as the main routing tag.

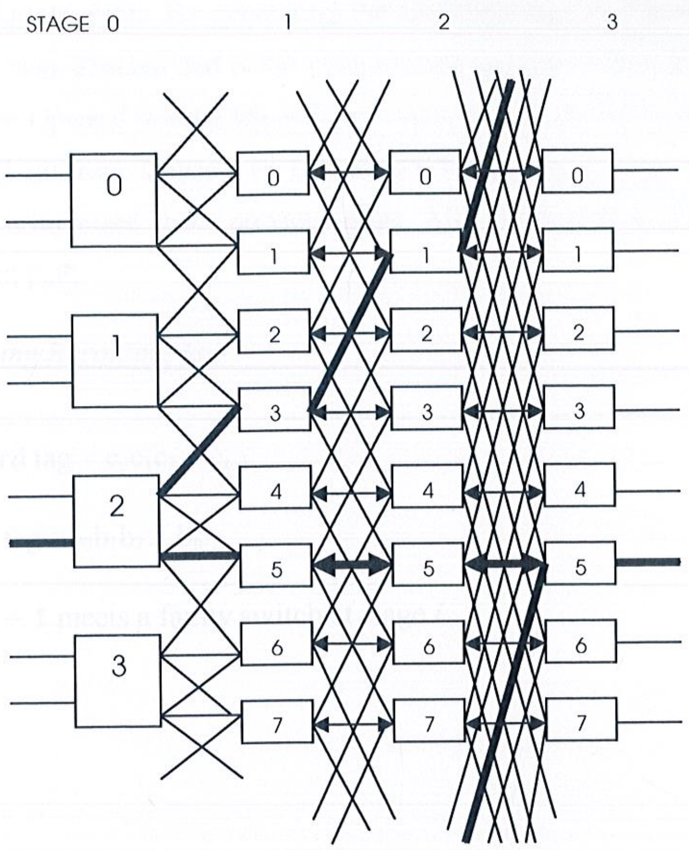


Figure 3.2 The two disjoint paths (2, 5, 5, 5) and (2, 3, 1, 5) indicated by the bold line in CSMIN with source=5 and target=5 where the 4-tuple items mean the switch indices at stage 0, 1, 2 and 3 respectively.

3.5 <u>ALGORITHM FOR DYNAMIC REROUTING IN CASE OF FAULTS OR COLLISIONS</u>

In this section, we introduce the rerouting methods when a faulty switch or link or a busy switch or link is encountered. We have used the backtracking scheme in which the switch sends the packet back along the traversed path to the pre–stage switch. The pre–stage switch takes the other disjoint path to tolerate the faulty or busy element. To dynamically switch packets between the two disjoint paths, the reversed straight link of the bidirectional straight link is used to reroute packets to the previously traversed switch in the previous stage of CSMIN.

We describe the algorithm for generating the rerouting tags in Algorithm 2. If the packet encounters a faulty or busy element and if the main routing tag followed, was Downward, then the packet now follows the Upward routing tag or vice–versa. Hence, the routing tag is changed from Downward to Upward or from Upward to Downward for further routing. The rerouting tag is computed by the switch traversed at the previous stage. After the rerouting behaviour, the packet is sent to the other disjoint path.

Algorithm 2. Generating Rerouting tag

```
The original downward tag = $c_0c_1c_2...c_{n-1}$
The original Upward tag= $b_0b_1b_2...b_{n-1}$
The packet at stage $i-1$ meets a faulty switch at stage $i$
Begin
  If ($i = 1$)
    If ($c_0c_1/ b_0b_1 = 10/00$)
      $c_0c_1/ b_0b_1 = 00/10$;
      remaining routing tag $b_2...b_{n-1} /c_2...c_{n-1}$
    Else If ($c_0c_1/ b_0b_1 = 11/01$)
      $c_0c_1/ b_0b_1 = 01/11$;
      remaining routing tag $b_2...b_{n-1} /c_2...c_{n-1}$
    End If
  Else If ($c_i/b_i = 1$)
    $c_i/b_i = 2$ ;
    remaining routing tag = $b_{i+1}.....b_{n-1}/c_{i+1}.....c_{n-1}$;
  Else If ( $c_i/b_i = 2$)
```

```
    $c_i/b_i = 1$ ;
        remaining routing tag = $b_{i+1}.....b_{n-1}/c_{i+1}.....c_{n-1}$;
Else If ( $c_i/b_i = 0$)
    $c_i/b_i = b_i/c_i$ ;
        remaining routing tag = $b_ib_{i+1}.....b_{n-1}b/c_ic_{i+1}.....c_{n-1}$;
Output the rerouting tag
End
```

3.6 RERUTING IN CSMIN

In this section, we present two rerouting situations one while traversing a non–straight link illustrated by Example (2) and the other while traversing a straight link illustrated by Example (3).

Example 2. The source is 2, the destination is 3, and the switch 7 at stage 2 is faulty (or the upward link to this switch is faulty).

Following algorithm 1 for S=2 and T=3 we have

1. S1=2 and T1=3
2. S=2 since T–S is odd
3. DownwardD=(3–2)mod8=1=0001 and UpwardD=(8–(3–2))mod8=7=0111=0222
4. DownwardD=1101 and UpwardD=0122 since c_0=0 and S is even
5. DownwardD=1001 and UpwardD=0022 since c_0=1 and S is even
6. DownwardD=1000 since S1=2 and T1=3

The Downward and Upward routing tags, thus, generated from algorithm 1 are 1000 and 0022 respectively. Take one of these as the main routing tag. Figure (3.3) shows the two disjoint paths (1,1,7,3) and (1,3,3,3) where the 4–tuple item mean the switch indices at stage 0,1,2 and 3 respectively.

We assume that the packet uses the Upward tag. Since the switch 7 at stage 2 is faulty (or the upward link to this switch is faulty) , the switch 1 at stage 1 computes the rerouting tag by using algorithm 2 and reroutes the packet to the downward non–straight link. The packet then uses the Downward tag to its destination. The number of rerouting hops to find the other disjoint path is 0. In Figure (3.4), switch 1 at stage 1 reroutes a packet to switch 3 at stage 2 that is in the other disjoint path. As a result, the packet can be delivered along the other disjoint path to tolerate the faulty switch that is indicated by gray color and the dash line means the original routing path.

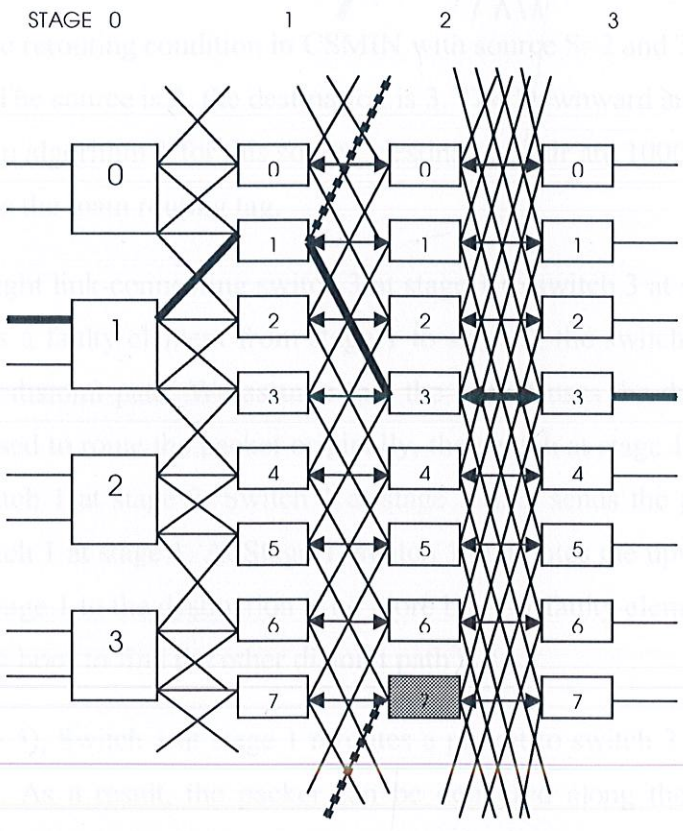


Figure 3.3 The routing condition in CSMIN with source S=2 and T=3.

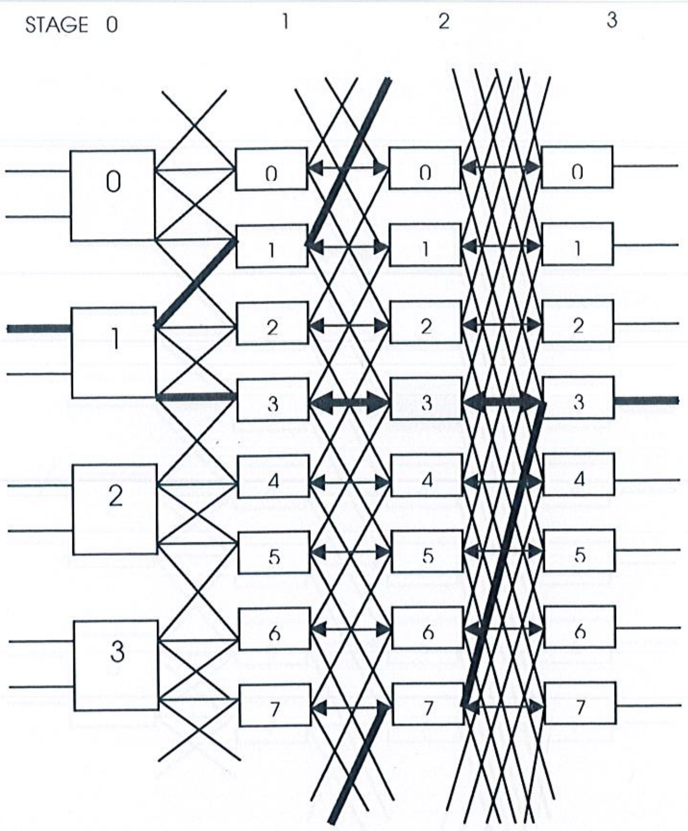


Figure 3.4 The rerouting condition in CSMIN with source S=2 and T=3 for example 2

Example3. The source is 2, the destination is 3. The Downward and Upward routing tags, thus, generated from algorithm 1 for this source–destination pair are 1000 and 0022 respectively. Take one of these as the main routing tag.

Let the straight link-connecting switch 3 at stage 1 to switch 3 at stage 2 be faulty. When a packet encounters a faulty element from stage 1 to stage 2, the switch at stage 1 reroutes the packet to the other disjoint path. We assume that the packet uses the downward tag. Since the Downward tag is used to route the packet originally, the switch at stage 1 takes the upward non–straight link to switch 1 at stage 2. Switch 1 at stage 2 then sends the packet to the backward straight link to switch 1 at stage 1. At Stage 1, switch 1 computes the upward routing tag, which is then used from stage 1 to the destination if no more busy or faulty element is encountered. The number of rerouting hops to find the other disjoint path is 1.

In Figure (3.5), Switch 1 at stage 1 reroutes a packet to switch 3 at stage 2 that is in the other disjoint path. As a result, the packet can be delivered along the other disjoint path to tolerate the faulty switch that is indicated by gray colour and the dash line means the original routing path.

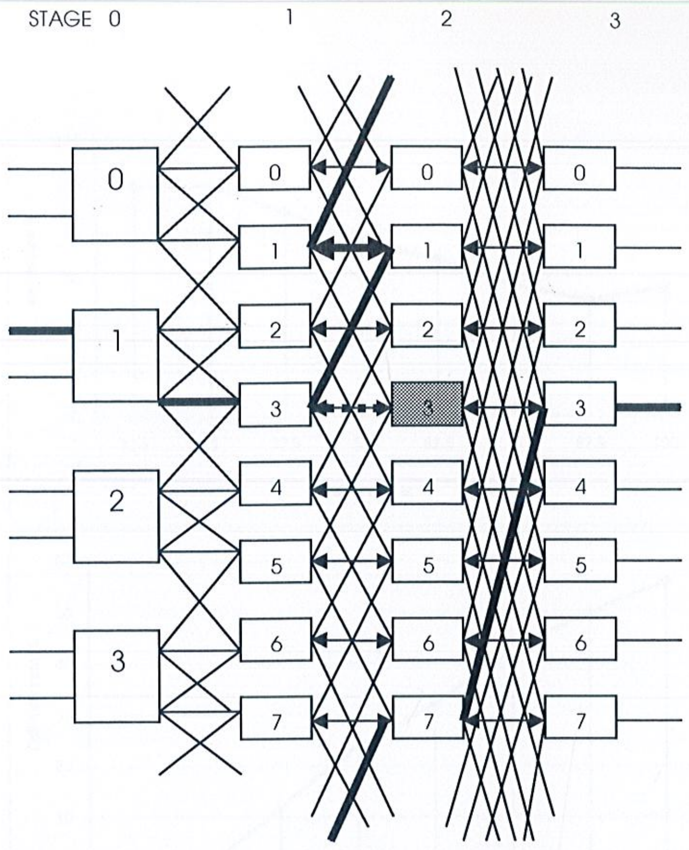


Figure 3.5 The rerouting condition in CSMIN with source S=2 and T=3 for example 3.

3.7 SIMULATION RESULTS

The previous work on routing algorithm on CSMIN did not generate correct routing tags for some source destination pairs. The routing tags generated for these source destination pairs were not correct in the sense that the resulting two disjoint paths in CSMIN for the desired destination did not reach the desired destination. Since their routing algorithm was not valid, its simulation resulted in packets arriving at invalid destinations. Therefore, we have not considered their algorithm for comparison with ours.

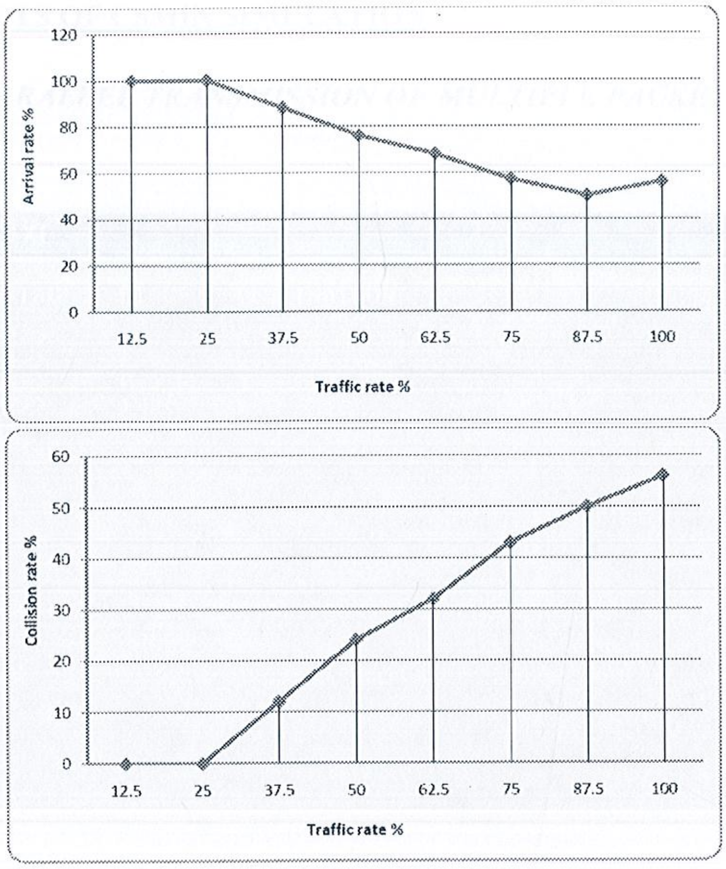
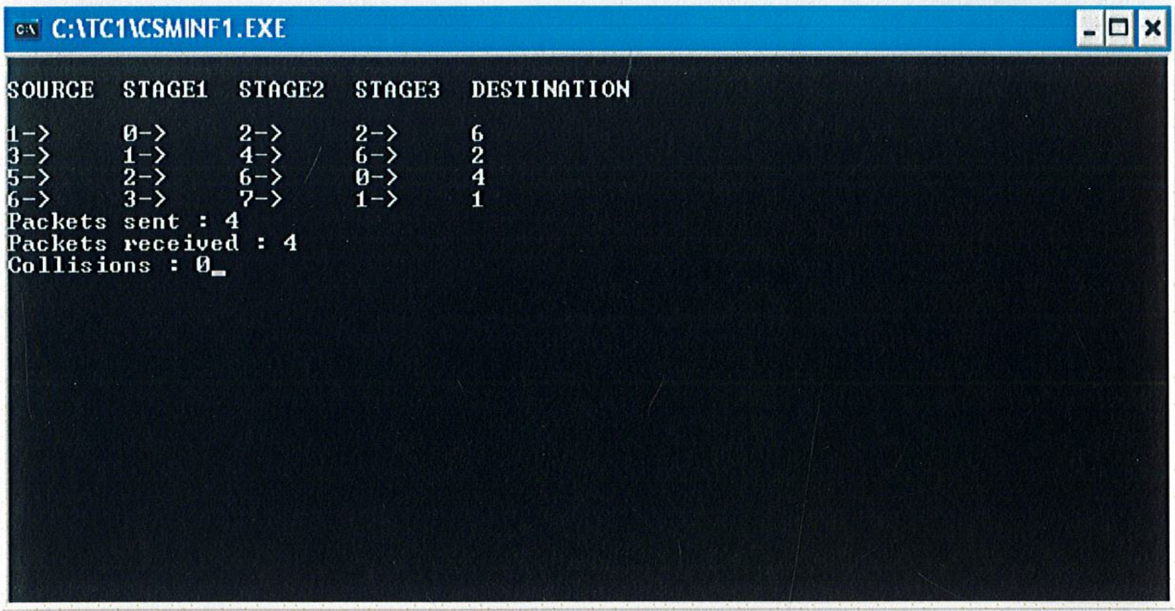


Figure 3.6 Simulation results for CSMIN of size 8

Our experimental studies for CSMIN of size 8 were done under traffic load of 6.25 to 100% in order to obtain results for arrival rate and collision rate. We assume that faulty switch is fully faulty, that is, the faulty switch cannot receive or send any packets from any input links or output links. Figure (3.6) shows the collision rates and the arrival rates of the CSMIN of size 8. Results obtained for arrival rate and collision rate clearly show that CSMIN with backward straight link has better throughput than other previously designed and simulated networks.

3.8 SCREEN SHOTS OF CSMIN SIMULATION

***TEST CASE 1:* PARALLEL TRANSMISSION OF MULTIPLE PACKETS, ON COLLISION FREE PATHS**

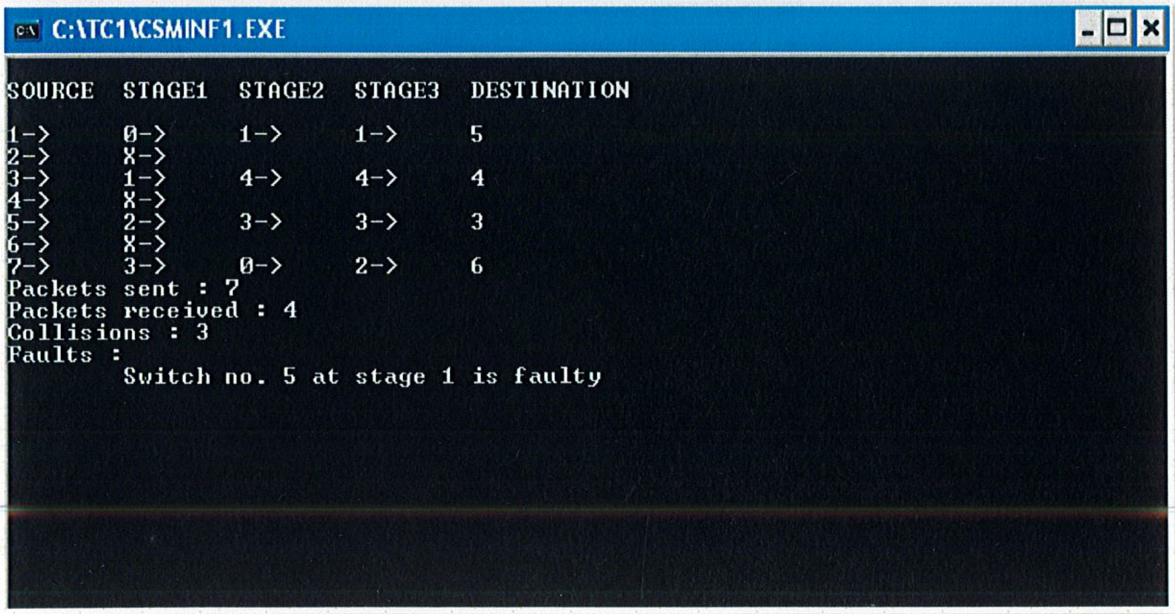


```
C:\TC1\CSMINF1.EXE

SOURCE   STAGE1   STAGE2   STAGE3   DESTINATION

1->      0->      2->      2->      6
3->      1->      4->      6->      2
5->      2->      6->      0->      4
6->      3->      7->      1->      1
Packets sent : 4
Packets received : 4
Collisions : 0_
```

***TEST CASE 2:* SWITCH FAULT AT STAGE 1**

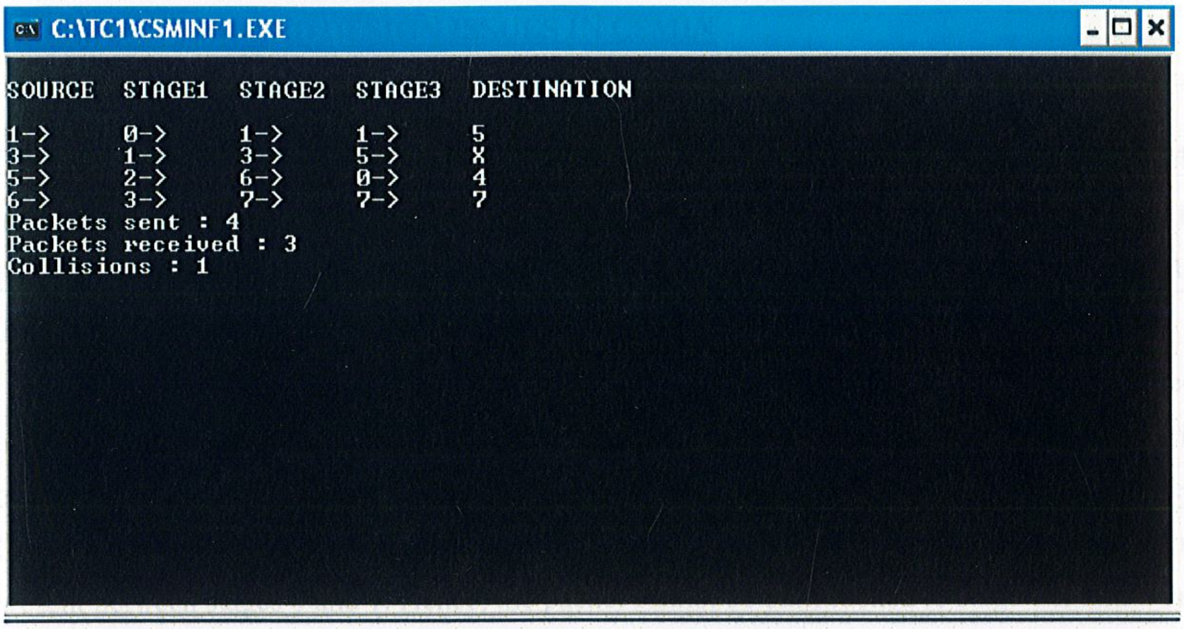


```
C:\TC1\CSMINF1.EXE

SOURCE   STAGE1   STAGE2   STAGE3   DESTINATION

1->      0->      1->      1->      5
2->      X->
3->      1->      4->      4->      4
4->      X->
5->      2->      3->      3->      3
6->      X->
7->      3->      0->      2->      6
Packets sent : 7
Packets received : 4
Collisions : 3
Faults :
         Switch no. 5 at stage 1 is faulty
```

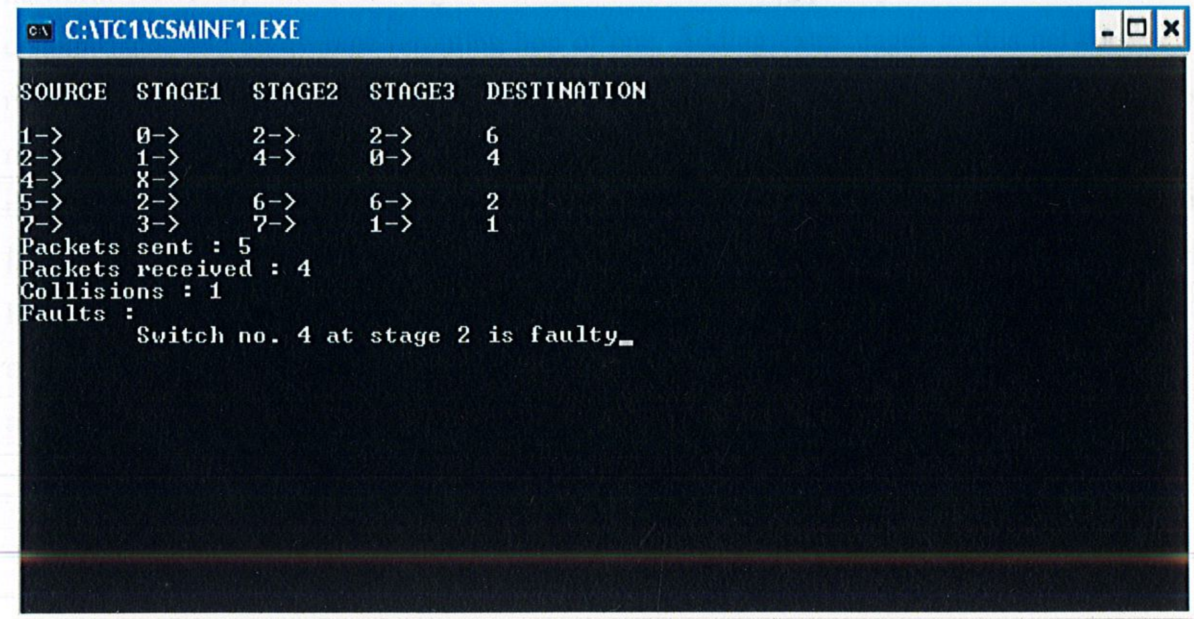
***TEST CASE 3**: COLLISION AT STAGE 3*



```
SOURCE  STAGE1  STAGE2  STAGE3  DESTINATION

1->     0->     1->     1->     5
3->     1->     3->     5->     X
5->     2->     6->     0->     4
6->     3->     7->     7->     7
Packets sent : 4
Packets received : 3
Collisions : 1
```

***TEST CASE 4**: SWITCH FAULT AT STAGE 2*



```
SOURCE  STAGE1  STAGE2  STAGE3  DESTINATION

1->     0->     2->     2->     6
2->     1->     4->     0->     4
4->     X->
5->     2->     6->     6->     2
7->     3->     7->     1->     1
Packets sent : 5
Packets received : 4
Collisions : 1
Faults :
        Switch no. 4 at stage 2 is faulty_
```

3.9 ALGORITHM AND DESIGN ISSUES IN CSMIN

The backtracking mechanism used in CSMIN results in increased system latency or delay in receiving the packets at the destination. The switch hardware complexity of CSMIN is high because bidirectional switches are used between stages $1\ to$ in order to backtrack a packet to the previously traversed switch $i -$ in case of a fault or collision at stage

The use of these switches to solve the backtracking purposes increases the hardware cost of the network. Distance tag routing algorithm also involves some amount of pre-processing overhead as the Upward and Downward routing tags are computed before the packet is sent. CSMIN cannot deal with faults or collisions at stage 1 because the bi-directional switches are not present between stages 0 and 1. Hence, the packet cannot be backtracked to stage 0 if a link fault occurs between stage 0 and 1. Hence, CSMIN does not have strong reroutability i.e. a packet cannot find an alternate path at each stage whenever a fault is encountered.

One alternative to solve these problems faced in CSMIN is to change the design of the network. However, this is not a good idea as CSMIN is completely 1-fault tolerant with two disjoint paths and an average rerouting hop of one. Adding extra stages to this network is also not a good option as it will further increase the hardware cost and increase the delay time and routing conflicts. The solution to overcome the difficulty involved is in modifying the distance tag routing algorithm. For CSMIN to be cost-effective in terms of hardware and to tolerate link faults between all stages, we introduce chaining in CSMIN resulting in a new design called Fully-chained Combining Switches Multistage Interconnection Network (FCSMIN). We propose a combination of distance tag routing and destination tag routing for FCSMIN and hence do away with the backtracking mechanism and the eventual degradation in system performance.

3.10 DESIGN OF FCSMIN

Fully chained Combining Switches Multistage Interconnection Network (FCSMIN) has multiple paths between any source-destination pair to provide better fault tolerance capability. The FCSMIN changes one of the original non-straight links of CSMIN at stage $i = 1\ to\ n -$ to a chained link.

A FCSMIN of size $N =$ consists of $n +$ stages labeled from $0\ to$ The first stage of FCSMIN is similar to CSMIN using $2 \times$ crossbar switches. For stages 1 to n-1, each switch of FCSMIN is augmented with a chain-in link and a chain-out link. The switches at intermediate stages are replaced by 3 × 3 switches. We also remove either of the non-straight links between the last two stages so that the final stage has 2 × 1 switches.

We have introduced this chaining at all stages except for the last stage. Either the Upward non-straight link or the Downward non-straight link of CSMIN can be changed to a chained link for stages $0\ to\ n -$. At the last stage n, we can remove either the Upward non-straight link or the Downward non-straight link. Thus, we present two models of FCSMIN in this paper namely, UpRoute-function based FCSMIN and DownRoute-function based FCSMIN. The UpRoute and DownRoute are two functions of the destination tag routing algorithm which is used by FCSMIN in order to resolve the algorithmic and design issues faced by CSMIN. These functions are discussed at a later stage in this paper.

In FCSMIN, since destination tag routing algorithm does not involve backtracking, we have done away with the bi-directional switches between stages $1\ to$ and thus brought down the hardware cost of the network as compared to CSMIN design.

In UpRoute function based FCSMIN, the chaining scheme is that switch is chained to switch $(j - 2^{i})mod2^{n-}$ where is the stage number from $1\ to\ n -$ and $n = \log_{2}$. For example, at stage 1 the chain-out link of switch 2 is connected to the chain-in link of switch 0. In the last stage, we remove all the Downward non-straight links. Figure (3.7) shows the topology of UpRoute function based FCSMIN.

In DownRoute function based FCSMIN, the chaining scheme is that switch is chained to switch $(j + 2^{i})mod2^{n-}$ where is the stage number from $0\ to\ n -$ and $n = \log_{2}$. For example, at stage 1 the chain-out link of switch 2 is connected to the chain-in link of switch 4. In the last stage, we remove all the Upward non-straight links. Figure (3.8) shows the topology of

DownRoute function based FCSMIN.

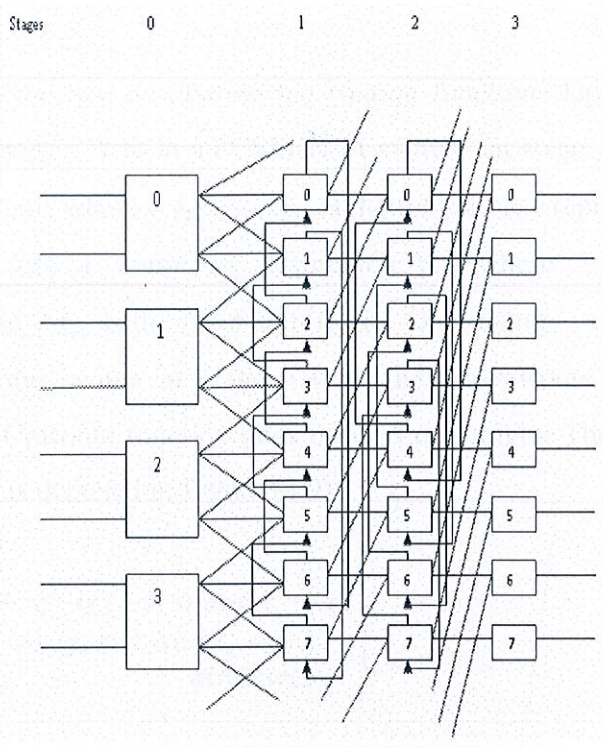


Figure 3.7 UpRoute Function based FCSMIN of size 8

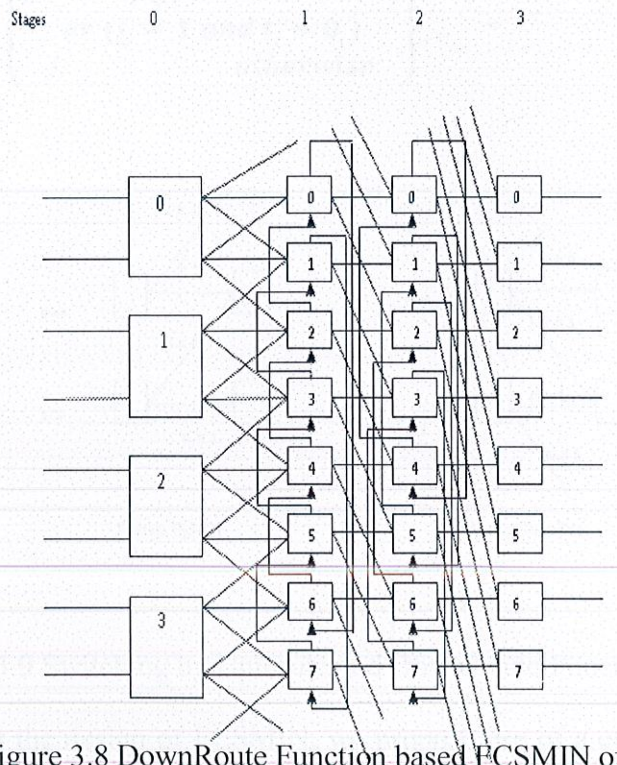


Figure 3.8 DownRoute Function based FCSMIN of size 8

3.11 ROUTING SCHEME IN FCSMIN

We have used the two destination tag routing functions UpRoute and DownRoute for routing packets from stage $1\ to$ in a FCSMIN. A switch at stage is an even switch if $j_i =$ or an odd switch if $j_i =$, where $j_0 j_1 j_2 \dots j_n$. is the n-bit binary representation of , and j_n. is the most-significant. Let T denote a destination tag where $t_0 t_1 t_2 \dots t_n$. is the binary representation of and t_n. is the most significant. Only using can decide routing from the switch at stage to the switch at stage $i +$ The DownRoute function goes straight or downward, while the UpRoute function goes upward or straight. The behavior of UpRoute and DownRoute functions is depicted in Figure (3.9).

$$UpRoute(j, t_i) = \begin{Bmatrix} j + 2^i \ \ if (j_i = 0\ and\ t_i = 1) \\ or\ (j_i = 1\ and\ t_i = 0\ ) \\ j \qquad\qquad otherwise \end{Bmatrix}$$

$$DownRoute(j, t_i) = \begin{Bmatrix} j - 2^i \ \ if (j_i = 0\ and\ t_i = 1) \\ or\ (j_i = 1\ and\ t_i = 0\ ) \\ j \qquad\qquad otherwise \end{Bmatrix}$$

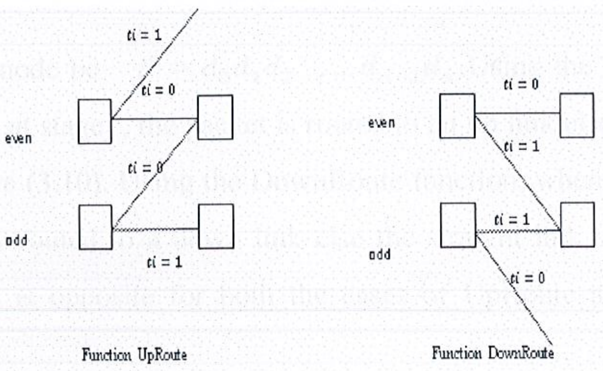


Figure 3.9 Switching by UpRoute and DownRoute Function at stage i

To implement the design of FCSMIN, we propose use of a combination of distance tag routing and destination tag routing algorithm. The distance tag routing algorithm is used for transfer of packets between stage 0 and stage 1 while the destination tag routing algorithm is used for transfer of packets from stage 1 to stage n. Under this combinational scheme, the distance tag algorithm generates a two bit routing tag for transferring packets between stage 0 and stage 1. The four links to stage 1 from a $2\times$ switch in stage 0 are downward non–straight, downward straight, upward straight, upward non–straight. These links are represented by the two bit combinations (00, 01, 10, 11) respectively.

For stages $1\ to$ with chaining links, the routing functions can be derived from the pre-defined UpRoute and DownRoute destination tag routing functions as

$$UpRoute(j,t_i)\begin{cases} (j-1)mod\ N\ at\ stage\ i \\ \qquad if(j_i=0\ and\ t_i=0) \\ \qquad if(j_i=1\ and\ t_i=0) \\ (j-1)mod\ N\ at\ stage\ i+1 \\ \qquad if(j_i=0\ and\ t_i=0) \\ \qquad if(j_i=1\ and\ t_i=0) \end{cases}$$

$$DownRoute(j,t_i)\begin{cases} (j+1)mod\ N\ at\ stage\ i \\ \qquad if(j_i=0\ and\ t_i=0) \\ \qquad if(j_i=1\ and\ t_i=0) \\ (j+1)mod\ N\ at\ stage\ i+1 \\ \qquad if(j_i=0\ and\ t_i=1) \\ \qquad if(j_i=1\ and\ t_i=1) \end{cases}$$

Let Destination node be $D=d_0d_1d_2\ldots\ldots d_{n-2}d_n$.Using the UpRoute function, when $d_i=$ in an odd switch at stage , the packet is routed to an up link else the chained link will be taken as shown in Figure (3.10). Using the DownRoute function, when $d_i=$ in an odd switch at stage , the packet is routed to a down link else the straight link will be taken. And when switch is even, routing is opposite for both the cases of UpRoute and DownRoute function applications.

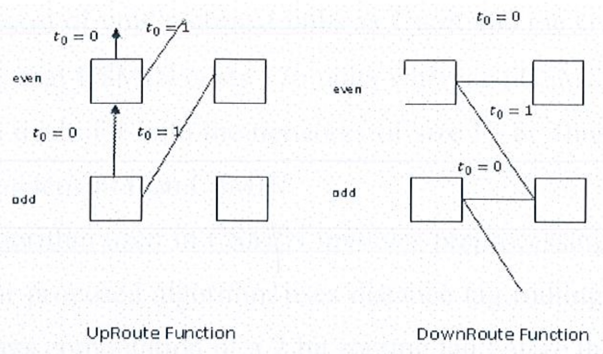


Figure 3.10 Switching by UpRoute and DownRoute Function for stages 1 to n

When faults occur, the chain link can provide alternate paths. When faults occur, the chain links provide alternate paths. Suppose a packet at a switch $j = j_0 j_1 j_2 \cdots j_n$ encounters a fault in link between stage and stage +1, the packet should be routed via chain link at stage to switch $j -$ mod i, for $1 \leq i \leq n -$ where $n = \log_2$. Thus, the FCSMIN is able to tolerate link faults at each stage. Thus, it has strong link reroutability.

3.12 COMPARATIVE ANALYSIS OF FCSMIN OVER CSMIN

The issues related to CSMIN which resulted in increased system latency and high hardware costs have been addressed. The switch hardware complexity of CSMIN is high because bidirectional switches are used between stages 1 *to* in order to backtrack a packet to the previously traversed switch $i -$ in case of a fault or collision at stage The use of these switches to solve the backtracking purposes increases the hardware cost of the network. FCSMIN does not make use of backtracking mechanism and hence the requirement of bidirectional switches has been eliminated. In addition to this, one of the non-straight links have been removed from the switches in stages 1 to N depending upon the functionality of FCSMIN being followed using either of UpRoute or DownRoute functions. This results in further cost reduction of FCSMIN.

If we compute the cost of CSMIN and FCSMIN, taking the cost of k, $m \times r$switches as $k \times m \times n$ units, the cost of unidirectional links as 1 unit and the cost of bidirectional link as 2 units, then we observe that CSMIN costs 376 units whereas FCSMIN costs 264 units. The above calculation has been made for both the networks of size N=8. Thus, our proposed design of FCSMIN is cheaper to implement than CSMIN.

The distance tag algorithm used in CSMIN involves preprocessing overhead to compute the n+1 bit routing tag. Our proposed algorithm uses distance tag routing only between stage 0 and 1. This approach requires computation of a 2 bit routing tag before the packet transfer takes place. Thus, our proposed algorithm results in less amount of preprocessing overhead. Moreover, the use of destination tag routing frees our routing scheme of the need to use to backtracking to deal with faults or collisions. This results in a significant improvement in system latency. Thus, FCSMIN possesses the same 1-fault tolerant and two disjoint path features of CSMIN at a lower hardware cost and achieves routing and rerouting with better performance metrics.

3.13 <u>RESULTS AND DISCUSSION</u>

The 1–Fault tolerant CSMIN provides two disjoint paths to solve collision situation. In this chapter, we have presented a more efficient and comprehensive algorithm, to provide two disjoint paths between every source destination pair that reaches the correct destination. Since the vertical distance between these two paths at any particular stage i is 2^i, the switch can easily reroute the packet between these two stages, in case of a fault or collision. Since the proposed algorithm backtracks the packet in case of faulty switch or collision, the network operation cost is reduced.

We have modified the existing design of CSMIN and presented an alternative design FCSMIN. The need to lower the hardware costs and backtracking penalties drove us to make the necessary changes in CSMIN. Introduction of chaining links in FCSMIN for stages 1 to n-1 eliminates the need of other redundant hardware used in CMSIN. It also retains the two disjoint paths that the topology of CSMIN so successfully exhibited. The use of destination tag algorithm helps us to do away with backtracking in cases of faults or collisions. Our design and algorithm can achieve the 1-fault tolerance with lower hardware costs and better system latency than CSMIN.

Our proposed algorithm for routing packets on CSMIN achieves one–fault tolerance to solve the faults or collision situations. To arrive at a good arrival ratio in times of increasing traffic, it is imperative to keep the packets lost due to collisions and faults, as low as possible. To deal with such cases, we will be attempting at designing a comprehensive and efficient algorithm that will make the CSMIN 2–fault tolerant. In our attempts to achieve this objective in the future, we will have to look for three disjoint paths in the network. For that we will be making the necessary hardware changes in CSMIN that achieve the optimum balance between performance and cost. Our future study will then comprise of comparisons between our newly designed 2–fault tolerant network and the 1–fault tolerant CSMIN.

CONCLUSION

In our study on "Multistage Interconnection networks with Fault- tolerance and Collision Solving", we took into consideration various classes of multistage interconnection networks and then simulated and tested them for their fault-tolerance and collision solving capabilities. The observations made from the simulation results for various parameters of performance metrics like throughput, system latency, reliability, fault tolerance, collision solving have ultimately helped us to compare, determine, design and simulate the ideal multistage interconnection network which can achieve at least one fault-tolerance and lowest rates of collisions namely, Combining Switches Multistage Interconnection Network (CSMIN). The 1–fault tolerant CSMIN causes the two disjoint paths to have regular distances at each stage. As a major contribution, we also present a more efficient and comprehensive algorithm that always generates the two disjoint paths in CSMIN for every source-destination pair. Our proposed algorithm uses backtracking and dynamic rerouting to deal with faulty or busy switching elements and communication links and hence improve its performance by avoiding packet losses. The experimental results also substantiate that the arrival ratio and the collision ratio for CSMIN is better than other networks. To eliminate the backtracking penalties and to reduce hardware cost associated with design of CSMIN we propose a new design called Fully- chained Combining Switches Multistage Interconnection Network (FCSMIN) that makes use of destination tag routing for stages 1 to n. FCSMIN has the similar characteristics of 1-fault tolerance and two disjoint paths between any source-destination pair but it can tolerate at least one link or switch fault at each stage without backtracking. Our design and algorithm can achieve the 1-fault tolerance with lower hardware costs and better system latency than CSMIN.

BIBLIOGRAPHY

1. Ching-Wen Chen and Chung-Ping Chung . Designing a disjoint path interconnection network with collision solving and Fault tolerance.
2. G. B. III Adams, D. P. Agrawal, and H. J. Siegel. A survey and comparison of fault-tolerant multistage
interconnection networks. IEEE Transactions on Computers, 20(6):14–27, 1987.
3. C.W. Chen, N. P. Lu, T. F. Chen, and C. P. Chung. Fault-tolerant gamma interconnection networks by chaining In IEE Proceedings on Computers and Digital Techniques, 147(2):75–80, 2000.
4. P. J. Chuang. CGIN:Afault tolerant modified gamma interconnection network. IEEE Transactions on Parallel and Distributed Systems, 7(12):1301–1306, 1996.
5. P. J. Chuang. Creating a highly reliable modified gamma interconnection network using a balance approach. In IEE Proceedings of Computers and Digital Techniques, 145(1):27–32, 1998.
6. F. C. M. Lau and W. C. Poon. Throughput analysis of B-networks. IEEE Transaction on Computers,
47(47):482–485, 1998.
7. K. Y. Lee and H. Yoon. The PM22I interconnection network. IEEE Transactions on Computers, 38(Issue2):302–307, 1989.
8. K. Y. Lee and H. Yoon. The B-network: A multistage interconnection network with backward links. IEEE Transactions on Computers, 39(7):966–969, 1990.
9. R. J. McMillen and H. J. Siegel. Performance and fault tolerance improvements in the inverse augmented data manipulator network. In 9th Symp. Computer Architecture, pp. 63–72, April 1982.
10. D. Rau, J. A. B. Fortes, and H. J. Siegel. Destination tag routing techniques based on a state model for the iadm network. IEEE Transactions on Computers, 41(3):274–285, 1992.
11. K. Yoon and W. Hegazy. The extra stage gamma network. IEEE Transactions on Computers, 37(11):1445–1450, 1988.
12. H.K Hwang, Book on Computer Architecture and Parallel Processing

13. Subramanian, A., and Nitin, "On a performance of multi-stage interconnection network," IEEE ADCOM, pp. 73-79, December 2004.

14. Nitin, "On Analytic Bounds of Regular and Irregular Fault tolerant Multi-stage Interconnection Networks"

15. Jungsun Kim," Performance Study of Packet Switching Multistage Interconnection Networks "

16. P.K. Bansal , Kuldeep Singh, R.C. Joshi, G.P. Saroha in Department of Electronics an Computer Engineering, University of Roorkee. "Fault-tolerant Double Tree Multistage Interconnection Networks"

APPENDIX A

SOURCE CODE

Find the source code of various Multistage Interconnection Network simulations in the CD-ROM attached at the end.