Design and Analysis of a Fault Tolerant Switch for B-ISDN

by

Talha M. Al-Jarad

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER ENGINEERING

January, 1997

DESIGN AND ANALYSIS OF A FAULT TOLERANT SWITCH FOR B-ISDN

BY

TALHA M. AL-JARAD

A Thesis Presented to the
FACULTY OF THE COLLEGE OF GRADUATE STUDIES
KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE
In
COMPUTER ENGINEERING

January 1997

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS
DHAHRAN, SAUDI ARABIA
COLLEGE OF GRADUATE STUDIES

This thesis, written by

TALHA M. AL-JARAD

under the direction of his Thesis Advisor and approved by his Thesis Committee, has been presented to and accepted by the Dean of the College of Graduate Studies, in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER ENGINEERING

Thesis Committee

Dr. Mostafa Abd – El – Barr (Chairman)

Dr. Khalid M. Al – Tawil (Co – chairman)

Dr. Habib Youssef (Member)

Department Chairman

Dean, College of Graduate Studies

8-1-97
Date

Dedicated to

my beloved mother and my lovely father

whose prayers, sacrifice, inspiration and love

led to this accomplishment

Acknowledgments

In the name of Allah, Most Gracious, Most Merciful. Read in the name of thy Lord and Cherisher, who created. Created man from a *(leech-like)* clot. Read and thy Lord is Most Bountiful. He Who taught (the used of) the pen. Taught man that which he knew not. Nay, but man doth transgress all bounds. In that he looketh upon himself as self-sufficient. Verily, to thy Lord is the return (of all).

(The Holy Quran, Surah 96)

First, all praise to be to Almighty Allah, who gave me the patience and courage to carry out this work. I glorify his name in the sincerest way through this small accomplishment. I seek his forgiveness, favor, and mercy. And I ask him to accept my little effort (*Aameen*).

I wish to thank my parents for everything they have done for me. They have been encouraging and praying for me day and night to accomplish this work. I believe that every success i have done is just because of their prayers. I do appreciate their scarification and I owe so much to them.

I acknowledge King Fahd University of Petroleum and Minerals for providing the opportunity and support to this research work.

I am deeply thankful to my thesis chairman, Dr. Mostafa Abd-El-Barr for his guidance , help, and patience. Working with him was indeed a learning experience.

Thanks are due to the department chairman and my thesis co-chairman Dr. Khalid Al-Tawil for his interest, advice, and guidance.

I am grateful to my thesis member Dr. Habib Youssef for his help, encouragement, and deep cooperation. He deserves my true respect.

I am deeply appreciative of my brothers in the dormitory for all the kinds of help they have done for me. I was feeling their prayers always. I have had unforgettable times with them. I want to thank specifically the members of the Cultural Club in the University for their wonderful company. Special thanks to Naser Al-Qahtani, Majed Al-Sodairy, Saeed Al-Saeed, Abdullah Rizq. Thanks are also due to my roommates Saeed al-Khodri and Abdullah Al-Sowelem. I am also thankful for my fellow graduate students and all my friends in the campus.

I acknowledge with gratitude, the affection, prayer, and encouragement of all my family members for their support and understanding, especially my dear brother Al-Zubair and Mostafa. I also appreciate the great encouragement of my uncle Mr. Abdul-Kader Kaadn.

Lastly, but not the least, special thanks are due to my wife for her patience and encouragement throughout this work.

Contents

List of Figures

List of Tables

Abstract

Name: **Talha M. Al-Jarad**

Title: **Design and Analysis of**
a Fault Tolerant Switch for B-ISDN

Major Field: **Computer Engineering**

In this thesis, we present a high performance fault tolerant switch for B-ISDN, called the Reliable and Zealous Network (RAZAN). It consists of $NLog_2N$ *switching elements of size* $n+1 \times n+1$, *where* N *is the number of inputs and* $n = Log_2N$ *is the number of stages. The performance of the network is evaluated analytically as well as through simulation under uniform traffic load. RAZAN shows a high throughput in the presence or absence of faults compared to a number of recently introduced networks. Moreover, RAZAN has a high terminal reliability even for large network sizes. The architecture of RAZAN also offers high fault tolerance, in-sequence delivery of cells, simple routing, and regularity.*

Master of Science Degree
King Fahd University of Petroleum and Minerals
Dhahran, Saudi Arabia
January 1997

خلاصة الرسالة

الاسم: طلحة محمد صالح الجراد
عنوان الرسالة: تصميم وتحليل موزع محتمل للأخطاء للعمل في شبكات (BISDN) .
التخصص: هندسة الحاسب الآلي
تاريخ التخرج: يناير ١٩٩٧

في هذه الرسالة، نقدم موزعاً ذا أداء عالي ومحتمل للأخطاء للعمل في شبكات BISDN وتسمى الشبكة المتحمسة ذات الإعتمادية وتختصر إلى (د ز أ ه). تتكون الشبكة من N $\log_2$ N صمام كل منها ذو حجم (n+1x n+1) ، حيث أن N تمثل عدد المداخل للشبكة و n هو عدد المراحل في الشبكة. وقد تم تقييم أداء الشبكة تحليلياً وأيضاً باستخدام برنامج للمحاكاة تحت حمل مروري منتظم. وقد أظهرت د ز أ ه كفاءة عالية حتى في حالة وجود أخطاء في الشبكة مقارنة بعدد من الشبكات المقترحة حديثاً. إضافة إلى هذا، تقدم د ز أ ه إعتمادية عالية، إحتمالية كبيرة للأخطاء، توصيل للخلايا في ترتيبها الصحيح، تسيير مبسط، مع انتظامية في التصميم.

درجة الماجستير في العلوم
جامعة الملك فهد للبترول والمعادن
الظهران ـ المملكة العربية السعودية
يناير ١٩٩٧

Chapter 1

INTRODUCTION

In this chapter, we provide a background on Asynchronous Transfer Mode (ATM), the technique adopted for B-ISDN. The motivation behind ATM is clarified. The cell structure of ATM is explained. Then, the multiplexing of individual application streams together using ATM is provided. We conclude with an explanation of the routing for switched services in ATM networks.

1.1 ATM Technique

Asynchronous Transfer Mode (ATM) can be considered to be the ground on which B-ISDN is to be built [1]. In 1988, ATM was chosen as the switching and multiplexing technique for B-ISDN. The ATM standard is designed to efficiently support high-speed digital voice and data communications. The expectation is that by the next decade, most of the voice and data traffic generated in the world will be transmitted by ATM technology [2].

Voice traffic can tolerate only a limited delay, but can accept a moderate loss ratio. On the other hand, data might be very sensitive to loss, but their delay requirements are not so strict. In ATM networks, through statistical multiplexing, several individual sources may share a high transmission rate link of capacity less than the sum of their peak arrival rates. Through statistical bandwidth assignment, a significant multiplexing gain can be achieved, especially for bursty traffic sources such as video telephony, video retrieval and document retrieval. However, to achieve this higher efficiency, ATM requires effective congestion control strategies to guarantee a minimum Quality of Service in all connections [3].

ATM is a packet-oriented switching and multiplexing technique based on the use of cells which are of fixed length, having a payload field and a header. At high transmission rate, ATM is expected to offer full bandwidth flexibility and utilization. Cells are transmitted contiguously on a transmission link, and are not identified by their position in relation to a fixed time reference but by means of address information in the header defining a virtual channel. The technique is asynchronous in the sense that the cells carrying a particular address (i.e. within a particular virtual channel) may appear at irregular intervals within the cell-stream. The technique is connection oriented in that a virtual circuit is established at call set-up time, and this will associate the virtual channels used on a series of network links to form the end-to-end connection [3].

The ATM cell is the basic unit of information transfer within the ATM network. It consists of five octet header field and a 48-octet information field as shown in Figure 1.1a. The header field which contains mainly routing information is transmitted first, followed by the information field. The structure of the ATM cell header is depicted in Figure 1.1b. A description of each field of the ATM cell header is given below.

Generic Flow Control (GFC): this four bit field is used to support traffic flow control at the UNI on the network side to avoid overload situations.

Virtual Path Interchange (VPI)/ *Virtual Channel Interchnage* (VCI): (8 or 12 bits for the VPI and 16 bits for the VCI) contain the routing information of the cell. The VPI and VCI identify, respectively, one particular VP within the physical length on which the cell is transmitted, and one specific VC inside the VP.

Payload Type (PT): two bits are available for the identification of the payload of the cell. Two types of payload have been considered so far. The first is user information in which the payload of the cells contains user information and service adaption function information. The second is the network information in which the payload type field of a cell indicates network information, additional information regarding the type the network control is given in the information field of the cell.

Cell Loss Priority (CLP): one bit is available for the identification of the priority of the ATM cell. Cells with CLP bit set may be discarded by the network while

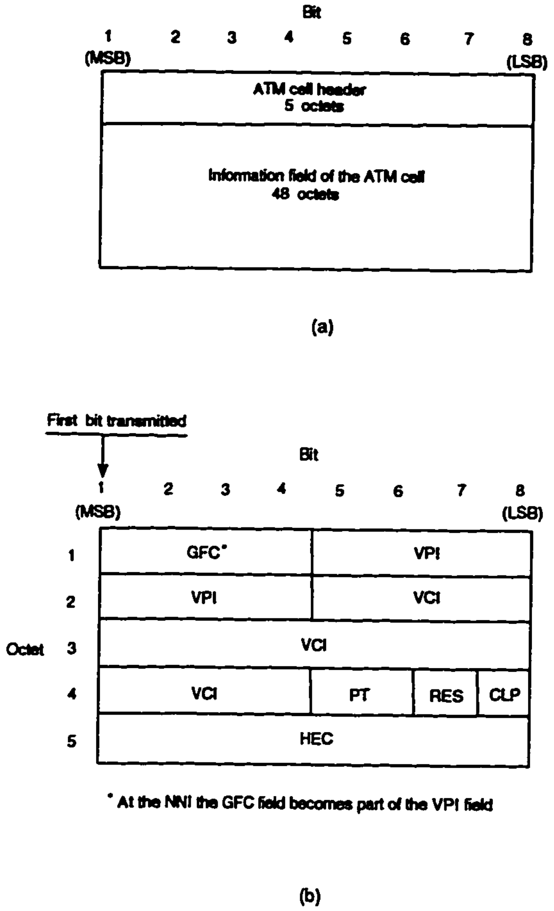


Figure 1.1: (a) ATM cell structure, (b) ATM cell header structure at the UNI.

cells with CLP is not set have high priority.

Header Error Control (HEC): this 8 bit field is used for cell header error protection, and provides recovery from single-bit errors occured in the header and a low probability of delivery of cells with errored headers.

Reserved (RES): the one bit of the RES field has been left for enhancement of the current cell header operation, and is left for future study.

The size of the ATM cell header (5 octet), as well as the size of the information field of the ATM cell, remain constant at both the User Network Interface (UNI) and Network Node Interface (NNI). The UNI is the interface between the user and the network node while NNI is the interface between network nodes. The only difference in the ATM cell format between the UNI and NNI is that the bits allocated to the GFC field of the ATM cell header at the UNI are allocated to extend the VPI field of the ATM cell header at NNI. Thus, while the VPI field at the UNI consists of 8 bits, at the NNI the same field consists of 12 bits [3].

1.2 Multiplexing in ATM

Two hosts can use a virtual path to multiplex many individual application streams together, using the VCI to distinguish between these streams. Figure 1.2 illustrates how ATM carries out cell switching [2]. The sequence of all cells from a particular user carry the same value in the header address field and this cell sequence constitutes a virtual channel. The rate of transmission of cells within a virtual channel can be variable, reflecting the source activity and source availability, and it is in this sense that the transfer mode is said to be asynchronous [3].

The basic routing entity for switched services in ATM networks is the Virtual Channel (VC). A VC is a logical unidirectional association between the end points of a link within a particular physical transmission path (Figure 1.3). The cells within a particular VC are identified by a particular combination of VCI and VPI. A specific value of VCI is assigned at each point where a VC is switched in the network. Routing of virtual channels is performed at a VC switch, and involves translation of the value of the incoming physical path and the VCI and VPI field in the headers to provide the identity of the outgoing physical path and the values to be inserted in the VPI and VCI fields of the cell for the next stage in its journey across the network. A Virtual Path (VP) is a logical association (bundle) of VCs characterized by the same VPI within a physical path connecting two VP switching nodes [3] (Figure 1.3).

Routing of Virtual Paths is performed at a VP switch, and involves translation of the values of the incoming physical path and the VPI field of the cell. The VCI field remains unchanged. It should be noted that two different VCs belonging to two different VPs may have the same VCI value. Thus, a VC is only identified by both the VPI and the VCI value. Also, a VCI has an end-to-end meaning only if the virtual channel is not switched in the network [3].

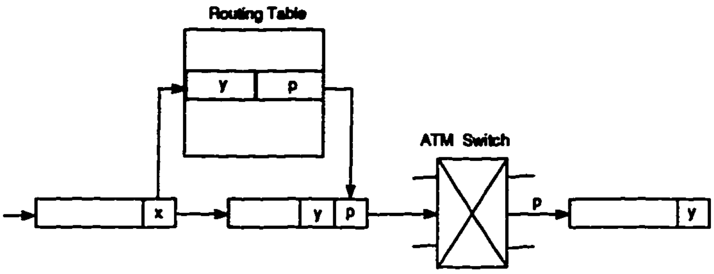


Figure 1.2: ATM cell switching

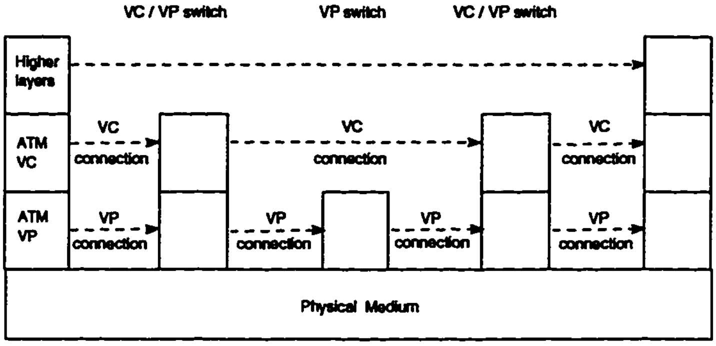


Figure 1.3: Types of ATM connections and switches.

Although ATM is defined as a packet-oriented technique, essentially, it is a connection-oriented switching technique in the sense that users first have to establish a virtual circuit (the set of translations in the appropriate network nodes) to their destination prior to the actual transmission of ATM cells. After the establishment of the virtual circuit, all cells of the same call follow the same established path. An important point to note is that the VP switch performs switching of VPs only, while the VC switch may perform switching to both VCs and VPs [3].

It is worthwhile to note that cells belonging to a connection do not always have to appear after one another in the data stream. Rather, the cells are statistically multiplexed with the amount of bandwidth allocated to a connection determined by the traffic requirement of the connection. ATM allows very efficient utilization of network bandwidth, as statistical multiplexing allows the total bandwidth available to be dynamically distributed among a variety of user applications. This is achieved by selecting virtual channel paths according to anticipated traffic and allocating the network resources needed. For guaranteed bandwidth applications, users must specify (before the virtual connection is set up) the amount of network resources they require. Users may specify their peak and average data rates, as well as maximum burst size. Using this information, it is up to the network to allocate resources in such a way that almost all information bursts are received intact. In theory, ATM should be able to ensure consistent performance to users in the presence of stochastically varying traffic [2].

1.3 Concluding Remarks

ATM is the switching technique adopted for B-ISDN. In this chapter, we presented a background on the ATM. The motivation behind ATM was clarified and the cell structure of ATM was explained. Then, the multiplexing of individual application streams together using ATM was provided. Finally, an explanation of the routing for switched services in ATM networks was given.

Chapter 2

FAST PACKET SWITCH ARCHITECTURES

Fiber optics technology provides the necessary bandwidth for transmission purposes. However, the challenge is to create a network that can provide high bandwidth services to the users with very high speed. The bottleneck comes primarily from switching. High- speed networks will carry all applications (voice, data, video, and images) in an integrated fashion. The suitable technique for such applications is packet switching. Several architectural designs have been proposed in recent years. In this chapter, an explanation of several architectures for fast-packet switching in the literature will be provided. These may be classified into three main categories; the shared-memory type, the shared-medium type, and the space-division type [4]. A survey on fast packet switch architectures for B-ISDN can be found in [4]. A

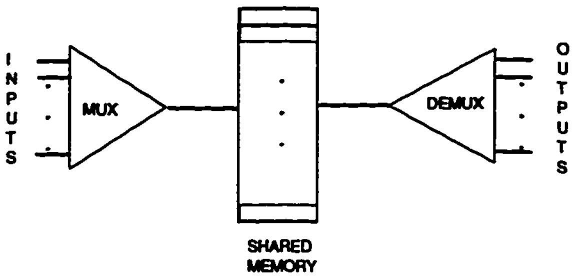


Figure 2.1: Basic structure of a shared-memory architecture.

comprehensive and specific survey on ATM switch architectures can be found in [5].

2.1 Shared-Memory Switches

Shared-memory switches consist of a single memory that is shared by all input and output lines. At the input side, all packets are multiplexed into a single stream which is fed to the shared memory. Inside the memory, packets are organized into separate output queues, one for each output line. Simultaneously, at the output side, packets are retrieved from the memory sequentially. The output stream is then demultiplexed, and packets are transmitted on the output lines. The basic structure of the shared-memory architecture is shown in Figure 2.1. Examples of the shared-memory fast packet switch architectures include the Prelude switch [6], the HITACHI's shared-buffer memory switch [7], the growable shared-buffer-based switch [8], and the output-buffer cell-based switch [9].

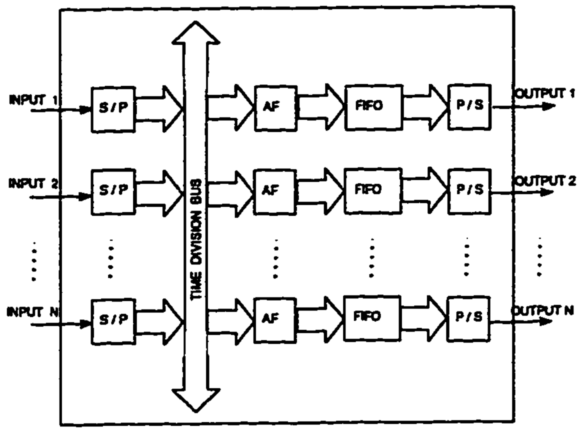


Figure 2.2: Basic structure of a shared-bus architecture.

2.2 Shared-Medium Switches

In shared-medium switches, all packets arriving on the input lines are multiplexed onto a common high-speed time-division bus of bandwidth equal to N times the rate of a single input line (Figure 2.2). Similar to the shared-memory type, the shared medium type switch is based on multiplexing all incoming packets into a single stream, and then demultiplexing the single stream into individual streams, one for each output line. The difference between this type and the shared-memory type is that in this type the memory is completely partitioned among the output queues. So, the queues can be organized as FIFOs [4]. Some examples of shared-medium fast packet switches include the ATOM switch [10] and the PARIS switch [11].

2.3 Space-Division Switches

In a space-division switch, multiple paths are established from the inputs to the outputs concurrently. Each path has the same data rate as an individual line. The control of the switch is distributed throughout the switching fabric. However, there is a problem associated with this type of architecture. The problem is that it may be impossible for all required paths to be set simultaneously. This characteristic is commonly referred to as internal blocking. The effect of this problem is that it limits the throughput of the switch [4].

Space-division switches have taken many forms, and may be classified into three categories: (I) crossbar fabrics, (II) banyan-based fabrics, and (III) fabrics with N^2 disjoint paths. All the above types follow an abstract reference model for space-division type switches as shown in Figure 2.3. For each input line i, there is a router (demultiplexer) which routes its packets to N separate pins, numbered $(i, 1)$ through (i, N), one pin for each output port. At the output side, for each output line j there is a concentrator (multiplexer) which connects all pins (i, j), $i = 1, 2, \ldots, N$, to output line j. At each time slot, the concentrator selects one packet, if any, from the output pins for transmission to output line j [4].

I) Crossbar Fabrics

Originally, crossbar switches were introduced for circuit switching. Basically, a crossbar fabric consists of a square array of N^2 crosspoint switches, one for each

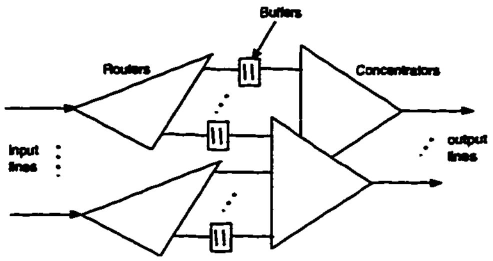


Figure 2.3: An abstract reference model for space-division switches.

input-output pair. To establish a connection between input line i and output line j, the switch at the (i, j)th crosspoint should be closed. It is thus easy to see that existence of N^2 crosspoint switches in the crossbar switch fabric. These crosspoint switches permit N disjoint pairs of input/output lines to be connected simultaneously .

Electronic versions of this fabric have been realized using transmission gates to implement the crosspoint switches. As shown in Figure 2.4, a transmission gate can assume two states: the cross state, which connects the horizontal input to the horizontal output and the vertical input to the vertical output, and the bar state, which connects the horizontal input to the vertical output and the vertical input to the horizontal output.

As long as there is no output conflict, all packets arriving at the inputs can be routed to their requested outputs. No global information regarding all packets and their requested destinations is required. This property is referred to as the self-routing property. The self-routing property is an important property because

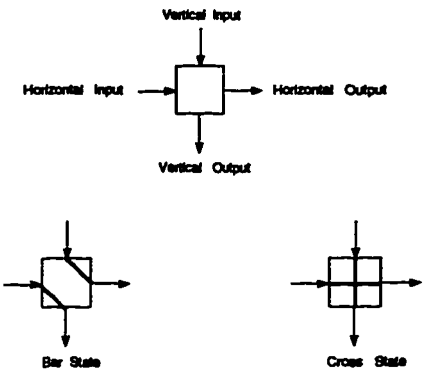


Figure 2.4: Transmission gate and its states.

it reduces the complexity of control needed at the switching fabric by distributing the control over all crosspoints. There are two schemes of buffering in the crossbar switches: buffering at crosspoints and buffering at the inputs of the crossbar [12] as they are illustrated in Figure 2.5 and Figure 2.6, respectively.

The crossbar fabric has some drawbacks which limit its use. First, it requires N^2 crosspoints. So, only limited size of the switch is realizable. Second, when self-routing is used, a knowledge of the complete output port address at each crosspoint is required. Third, the transit time is not constant over all input/output pairs.

II) Banyan-Based Space-Division Switches

In banyan-based space-division switches there are N inputs and N outputs using $N/2\ log_2 N$ elementary binary switches. The entire switching fabric is considered to be an $N \times N$ multistage interconnection network (MIN). One benefit of MINs is the reduction of the total number of crosspoints switches needed in the fabric compared to the crossbar switch. The negative point associated with this design is

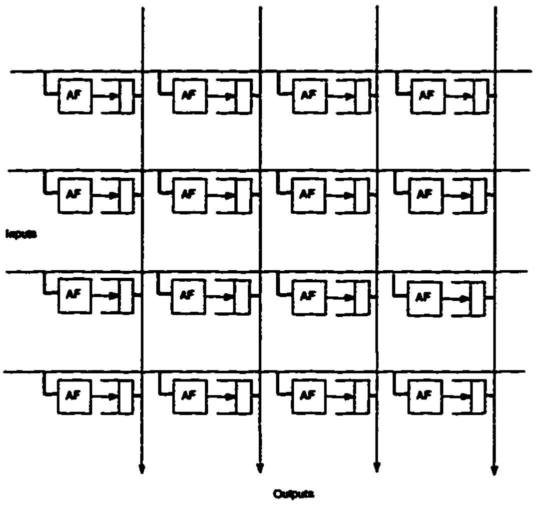


Figure 2.5: Buffering at crosspoints in a crossbar switch.

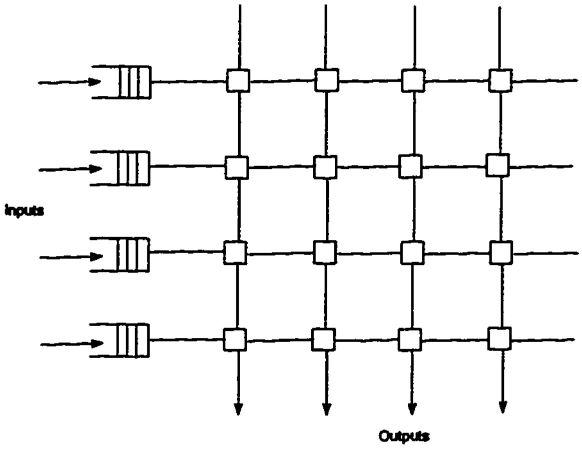


Figure 2.6: Input buffering in a crossbar switch fabric.

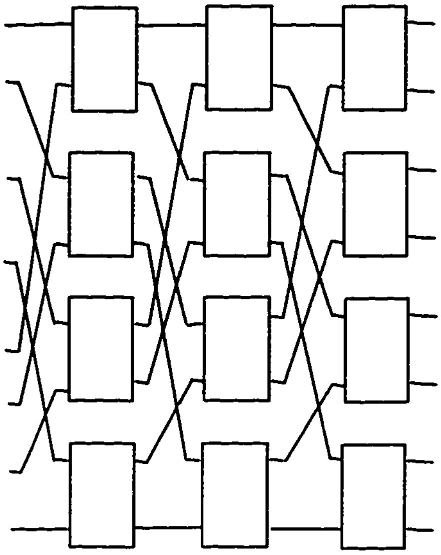


Figure 2.7: An 8×8 OMEGA network.

that internal conflicts may now arise. This is due to the fact that two packets may arrive at the same crosspoint and need to be switched to the same output, similar to the crossbar switch. The internal conflicts in MINs may arise even if no two packets are destined to the same output port. This introduces performance limitations and causes such an architecture to be blocking.

There are many forms that such interconnection networks can take, depending on the specific procedure used in constructing them. Figure 2.7 shows an example of Omega and Figure 2.8 shows an example of delta MINs. A common property to all banyan-based networks is that there is a single path between an input line and output line established using the self-routing procedure. All networks allow up to N paths to be established between inputs and outputs simultaneously [4].

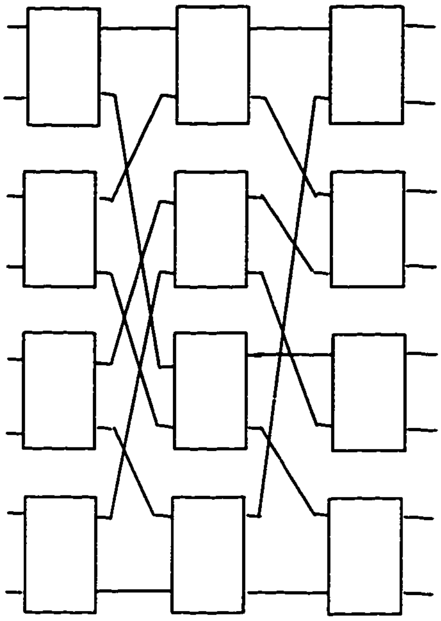


Figure 2.8: an 8×8 delta network.

The shortcomings of banyan-based networks are internal blocking and the throughput limitations. Simulation results have shown that the maximum throughput achievable is much lower than that obtained for the crossbar switch, and degrades with increasing network size. Thus, all switch architectures based on banyan networks are distinguished by the means used to overcome blocking and improve the throughput and packet loss performance [4]. Some of the solutions adopted include buffered-banyan switching fabrics [4], Batcher-banyan switching fabrics like the Starlite [13], Sunshine [14] and the modular approach based on divide-and-conquer [15], multiple banyan switching fabrics like the unbuffered shuffle exchange network [16] and the tandem banyan [17], dialated banyan network [18], open-loop shuffleout switch [19] and the closed-loop shuffleout switch [20].

III) Switching Fabrics with N^2 Disjoint Paths

One of the most obvious examples for this type of switches is the bus matrix switching architecture. It uses N broadcast input buses, N multiaccess output buses, and N^2 crosspoint buffer memories. Each crosspoint memory component contains an address filter corresponding to the output bus to which it is connected. Two other fabrics with similar architecture to the bus matrix fabric with output buffering are the Knockout switch [21] and the Integrated Switch Fabric [22].

2.4 Concluding Remarks

In this chapter, an explanation of several architectures for fast-packet switching was provided. These were classified into three categories: the shared-memory type, the shared-medium type, and the space-division type. The general idea of each category was explained and examples for each were provided. Space-division type was also classified into three categories; crossbar fabrics, banyan-based fabric, and N^2 disjoint paths fabrics. An explanation for each of the space-division categories was given with examples.

Chapter 3

FAULT TOLERANCE OF MINs

In large multiprocessor systems, the shared bus and the crossbar switch represent two extremes in the design of interconnection schemes. The shared bus is inexpensive, but it is too slow and has limited bandwidth when a large number of processors must rely on it for communication. At the other extreme, the crossbar switch provides high bandwidth and the fastest possible communication speed, but its cost grows with the square of the number of processors. For large systems, speed makes the crossbar desirable, but it is too expensive. Multistage Interconnection Networks (MINs) provide a compromise between the shared bus and the crossbar switch. It offers reasonable speed and bandwidth at a modest cost [4].

Various types of switching networks have been proposed to handle fast packet switching in B-ISDN. Most of them employ some form of MINs as the basic switching element. Although MINs allow for the use of a very simple routing algorithm, it is highly sensitive to blocking (hence possibly low throughput and high delay) because of its single path structure. Furthermore, because of its single path characteristic, a single fault in either a switching element (SE) or link may render the network incapable of performing its intended function. Therefore, the various networks that have been proposed usually involve mechanics to enhance the MINs so as to allow for multiple paths which in turn gives it a degree of fault tolerance. Fault tolerant network is one that can continue to correctly perform its specified tasks even in the presence of faulty switching elements or links. A considerable amount of work on MINs has been devoted to make them fault tolerant. In this chapter, some of the recent and important fault tolerant MINs are surveyed. A comprehensive survey on fault tolerant MINs can be found in [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34].

3.1 Delta Networks

Among the MINs, the delta network defined by Patel [35] possesses two properties: unique path for each input-output pair, and digit-controlled routing. The latter property, known as self-routing, makes delta networks very attractive to high-speed packet switch designers. However, it comes at the expense of internal blocking as discussed below. Many well-known MINs belong to the delta class, such as Banyan, Baseline, Reverse baseline, Omega, Modified Data Manipulator, and generalized Cube networks. These networks were shown to be topologically equivalent [36].

Some of these networks are shown in Figure 3.1 for $N = 8$ [5].

In general, a rectangular ($N \times N$) delta-b network is constructed using unbuffered ($b \times b$) SEs (crossbars), organized in n stages, where $N = b^n$ (n=1,2,...) and each stage has N/b SEs. Banyan networks have some desirable features such as self-routing, modularity, having the same latency for all input-output pairs, supporting synchronous and asynchronous modes of operation, suitability for VLSI implementation, and finally having a complexity of $O(Nlog_2N)$ compared to $O(N^2)$ for the crossbar switch. However, banyan networks have a major problem which is the internal blocking which arises when two cells destined to different output ports request the same outlets of an SE. In such situation, one cell is selected according to some policy, and the other cell is dropped [5].

There are various techniques reported in the literature to enhance fault tolerance of the Delta networks. These techniques can be classified into four categories: adding an extra stage, adding extra links, chaining of switches within a stage, and adding extra input/output ports.

3.1.1 Adding an Extra Stage

A) Extra Stage Cube Network

Adding extra stage to the network to make it fault tolerant was first proposed by Adams III and Siegel for their Extra Stage Cube (ESC) [37]. The ESC is derived from the generalized cube MIN. The generalized cube is an $N \times N$ MIN, $N =$

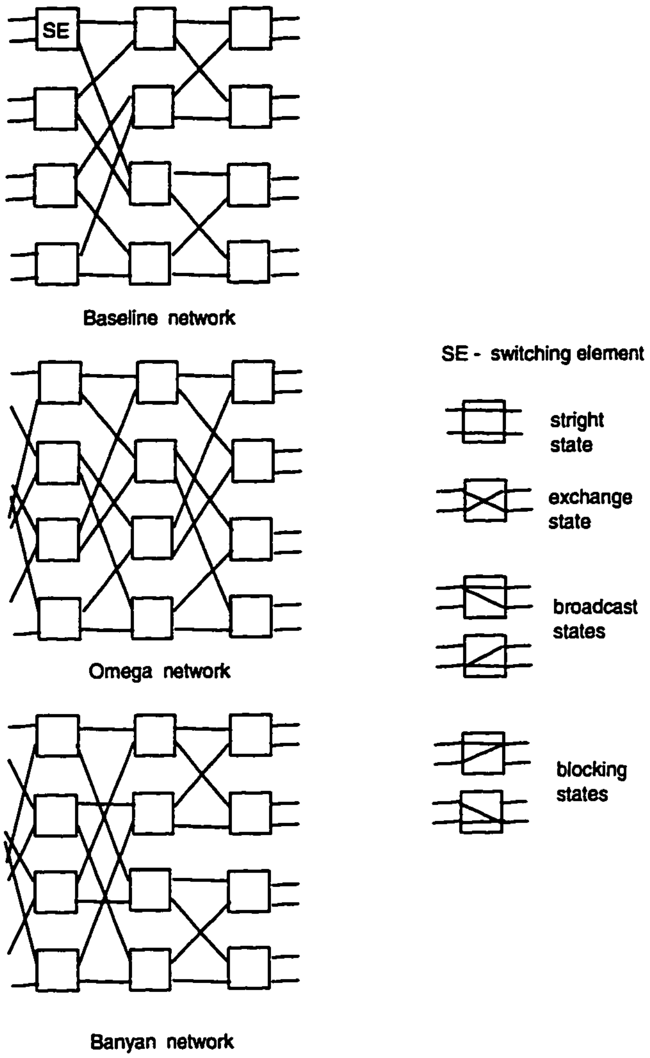


Figure 3.1: Three different topologies of banyan networks.

2^n, with $n = log_2N$ stages, each stage consisting of N links connected to $N/2$ switches as shown in Figure 3.2. The ESC is formed from the generalized cube by adding an extra stage to the input side of the network along with multiplexers and demultiplexers at the input and output stages, respectively. In addition, dual I/O links to and from the devices using the network are required. Stage n is connected like stage 0, that is, links that differ in the low order bit are paired. Figure 3.3 illustrates the ESC for $N = 8$ [37].

Stage n and stage 0 can be enabled or disabled (bypassed). A stage is enabled when its switches are being used to provide interconnection; it is disabled when its switches are being bypassed. Normally, the network will be set so that stage n is disabled and stage 0 is enabled. The resulting structure matches that of the generalized cube. After the fault is found, the network is reconfigured. A fault in stage n requires no change in network configuration; stage n remains disabled. If the fault occurs in stage 0, then stage n is enabled and stage 0 is disabled, i.e., stage n replaces the function of stage 0. For a fault in a link or in a switch in stages n - *1* to 1, both stages n and 0 will be enabled. Enabling both stages n and 0 provides two distinct paths between any source and destination, at least one of which must be fault-free given any single fault in the network.

B) Enhancing Fault Tolerance of the Shuffle-Exchange Network

Blake and Trivedi [38] proposed the Shuffle-exchange with Extra Stage (SEN+) to enhance the fault tolerance of the SEN. An $N \times N$ SEN+ network is an $N \times N$ SEN

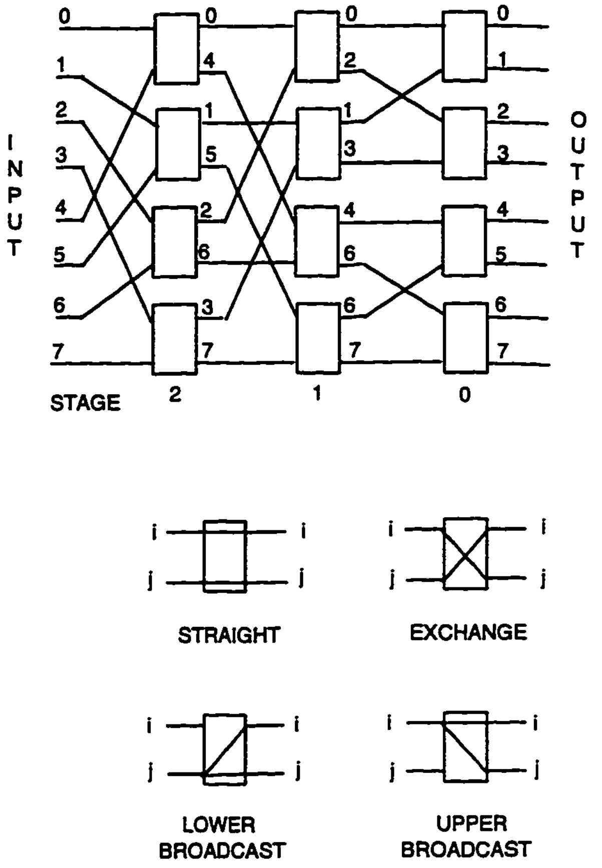


Figure 3.2: The generalized cube network.

with an additional stage. Figure 3.4 shows an 8×8 SEN+. The reason for adding a stage to the SEN is to allow two paths for communication between each source (S) and destination (D). While the paths in the first and last stages of the SEN+ are not disjoint, the paths in the intermediate stages are disjoint. The reliability analysis shows that the SEN+ has higher reliability than that of the SEN. This is achieved at a small increase in the number of switches used. As the network size increases, it becomes more advantageous to choose the SEN+ over the SEN [38]. The main problem associated with SEN+ is that it has critical first and last stages. So, any switch failure in the first or the last stage will affect the performance severely.

C) The Selective Extra-Stage Butterfly

In 1992, Konstatinidou [39] proposed the Selective Extra-Stage Buterrfly (SESB) to provide fault tolerance in MINs by providing multiple paths between any source and destination nodes. The SESB can be constructed by augmenting the regular butterfly with m extra stages. The outputs of each extra stage (ri) are connected to the inputs of the previous extra stage (ri-1). The outputs of the first extra stage (r0) is connected to the processors. The inputs of the first extra stage are connected to the outputs of any stage S of the regular butterfly such that every stage of S is repeated in the extra stages [39]. Figure 3.5 shows an example of a 16×16 selective extra stage butterfly with degree 2 switches.

The simulation results of the SESB show its competitiveness to other networks of comparable wiring complexity and hardware cost, namely the extra stage butter-

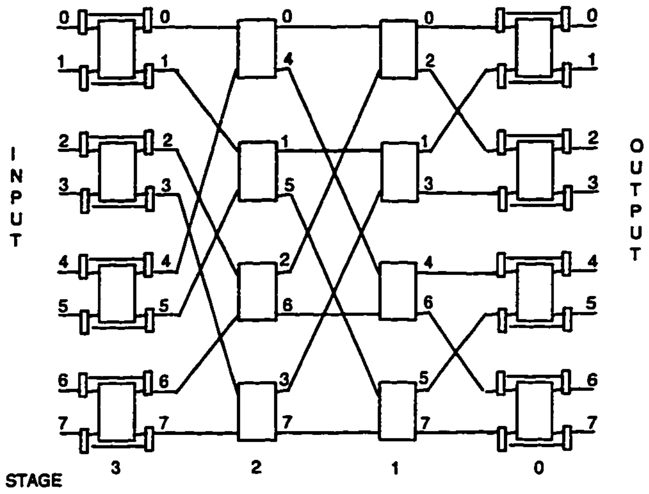


Figure 3.3: The Extra stage cube network for $N = 8$.

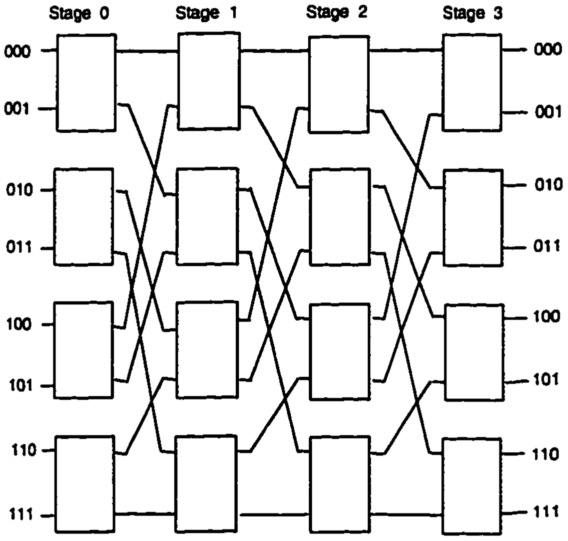


Figure 3.4: An 8×8 SEN with an extra stage.

fly with enabling/disabling and the extended extra stage butterfly. Moreover, the SESB outperformed the multibutterfly (a network of higher wiring complexity and hardware requirements) at high loads in the absence of faults. The SESB exhibited comparable performance to the basic baseline network in the presence of faults [39].

3.1.2 Adding Extra Links

Another possible way of enhancing fault tolerance to the delta networks is by adding extra links. Some of the networks that follow this technique are Augmented Data Manipulator (ADM), Enhanced Augmented Data Manipulator (EADM) [40], Improved EADM (IEDAM) [41, 42], Logical Neighbourhood (LN) [43], and Improved LN (ILN) networks [41, 42].

A) EADM and IEDAM Networks

Data Manipulator (DM) network is a class of MINs that is based on the *plus-minus* 2^i interconnection functions. A DM network with N inputs and N outputs consists of $n = log_2 N$ stages. Each stage is a column of N switches and 3N links. An extra column of switches is appended as the output stage. At stage i, switch j is connected to switch $(j + 2^i) mod N$, switch j, and switch $(j - 2^i) mod N$ of stage i+1. Switch control in DM networks is limited to one pair per stage, i.e. in stage i, switches that agree in the *ith* bits are set identically. DM networks whose switches can be set independently are called Augmented Data Manipulator (ADM) networks [40, 43].

Two interesting properties of the ADM network family are the existence of multiple paths between most of source-destination pairs and the ability of network switches to be set independently. Due to these properties, ADM networks can employ some dynamic routing tag scheme, i.e. each switch in the network derives its control signal from certain bits in the routing tag. This feature allows ADM network, in some cases, to avoid faulty switches or links. However, ADM networks are not fault-tolerant networks because there is only one disjoint path if the addresses of the source and destination are both even or odd [40].

One possible way to improve performance and to make ADM network fault-tolerant is by adding some extra links to the network. Such network is called Enhanced ADM (EADM) network. One version of EADM adds two half-links to each switch in stage 1 through stage n-1. In stage i, $1 \leq i \leq n-1$, the half-link connects switch j, $0 \leq j \leq N-1$, to switch $(j+2^{i-1}) mod N$ and switch $(j-2^{i-1}) mod N$. The name half-link comes from the fact that $2^{i-1} = (2^i)/2$, i.e. these links move routing tags half the distance of the existing nonstraight links. Motivation behind the addition of half-links is to reroute packets around a faulty straight link [40]. Figure 3.6 illustrates an EADM network with $N = 8$.

Recently, Abd-El-Barr and Abed [41] proposed improvements to the EADM to improve its fault tolerance capabilities and they call the new architecture the IEADM. The IEADM network is shown in Figure 3.7. The network has three paths for each source-destination pair from stages 0 to n-1, inclusive. Therefore, the

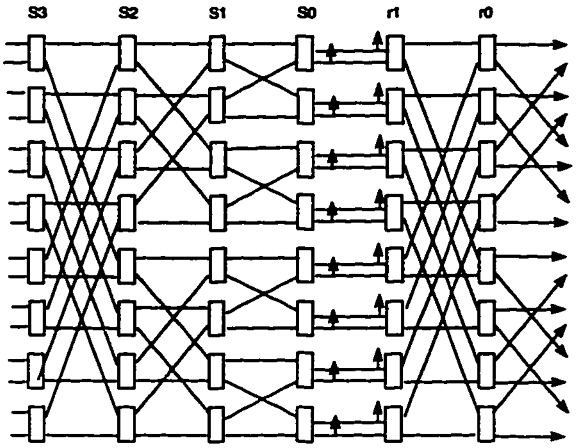


Figure 3.5: A 16×16 selective extra-stage butterfly.

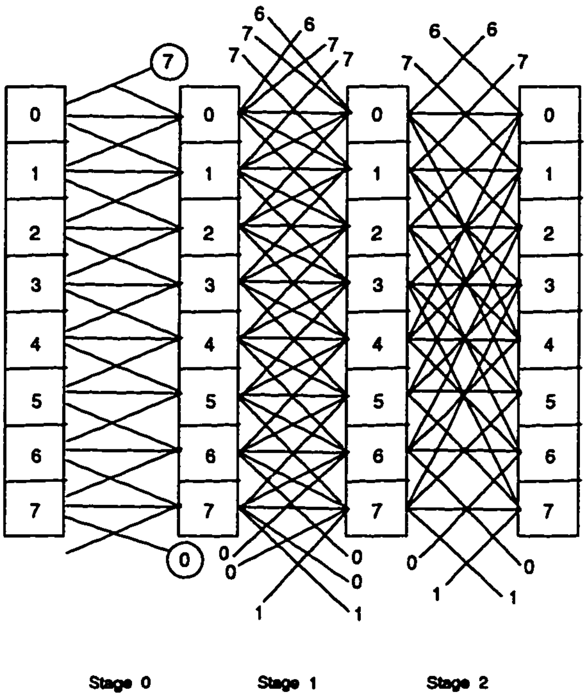


Figure 3.6: The EADM network with $N = 8$.

IEADM network can be considered as a two fault tolerant network for stages 0 to n-1. In stage n, two paths are available for each source destination pair. This will make the network a single fault tolerant for stage n [42].

B) LN and ILN Networks

A Logical Neighbourhood (LN) network with N inputs and N outputs consists of $n = log_2 N$ stages. Each stage is a column of N switches and nN links. An extra column of switches is appended as the output stage. In stage 0 through stage n-1, switch j is connected to the $n+1$ neighboring switches (in the successive stage) whose binary addresses differ by at most one bit from the binary address of switch j. A LN network with $N = 8$ is illustrated in Figure 3.8. The LN network has the property of multiple paths and it is a fault tolerant network with at least n disjoint paths from any source to any destination in the network [40].

The Improved Logical Neighbourhood (ILN) network [41] is an improved version of the LN network [40]. The ILN network is shown in Figure 3.9 and it has n paths for any source-destination pair. Thus an n-stage ILN network can tolerate n-1 faults in any stage [42]. The reliability of both IEADM and ILN were evaluated and compared. The results show that as the probability of failure increases, the terminal reliability of the IEADM network drops faster than the terminal reliability of the ILN network. This result is expected since the probability of finding a path in the ILN network is higher than the probability of finding a path in the IEADM network [42].

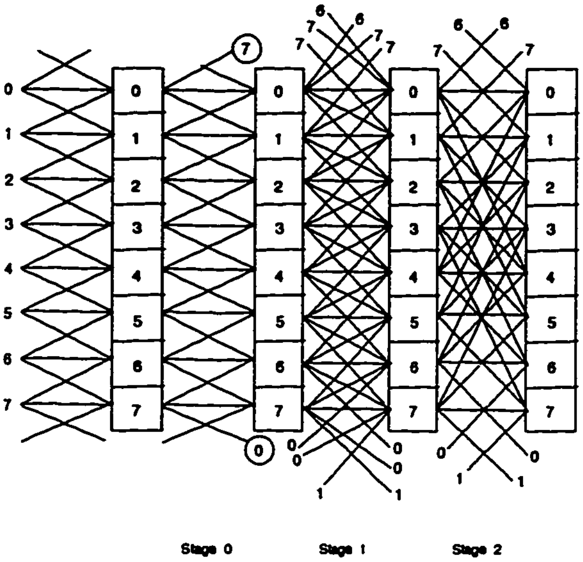


Figure 3.7: An 8×8 IEADM network.

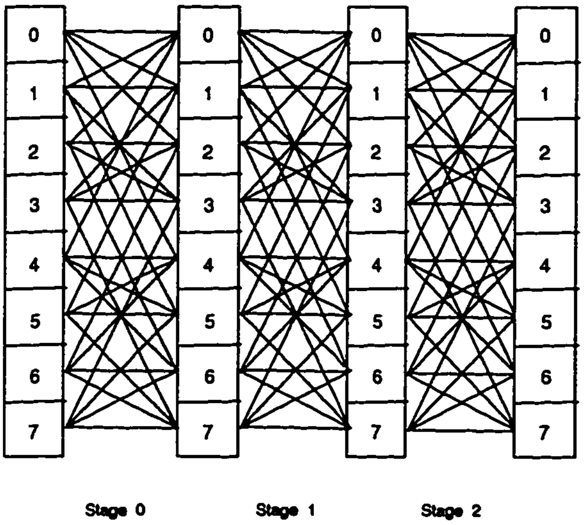


Figure 3.8: LN network with N = 8.

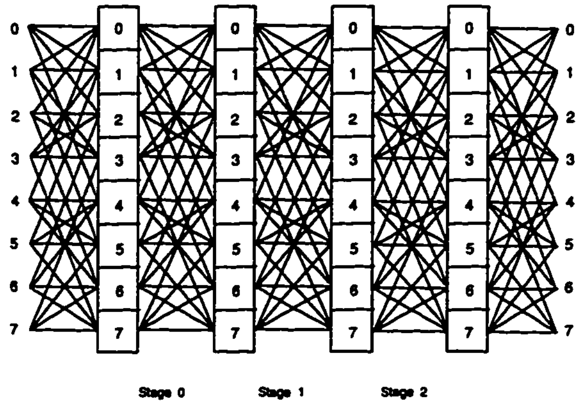


Figure 3.9: An 8×8 ILN network.

3.1.3 Chaining of Switches Within a Stage

The third approach to enhance fault tolerance to MINs is to chain switches within the same stage. This idea was first proposed by Kumar and Reddy in 1987 [33]. Starting from a shuffle-exchange MIN, an Augmented Shuffle Exchange Network (ASEN) is constructed by adding a stage of 2×1 multiplexer switches at the input side (for making multiple connections from the PEs to the MINs), replacing the last stage switches by 1×2 demultiplexer switches (for providing multiple connections from the MIN to each PE), and by adding links to connect certain groups of switches within each stage in loops (to provide an alternate way of routing in each stage). This makes more than one path available in the network to tolerate faulty elements. There are several versions of the ASEN, depending upon the number of switches included in each loop.

The ASEN-2, in which the loops contain exactly two switches, is shown in Figure 3.10 where N = 16. ASEN-2 has $n + 2$ stages, where $n = log_2 N$. Stage 0 consists of N switches with 2 inputs and one output (2×1 multiplexers); stages 1 through n consists of $N/2$ switches of size 2×2; and stage n+1 consists of N switches with one input and two outputs (1×2 demultiplexers) [33].

3.1.4 Adding Extra I/O ports

A typical example of this category is the Dynamic Redundancy Network (DRN) [44]. The DRN is based on the graph representation of the generalized cube network. The DRN has n stages, where $N = 2^n$. Stages are ordered from *n-1* to 0 from the input side to the output side of the network. Each stage has $N + S$ switches followed by *3(N+S)* links, as shown in Fig 3.11 for $N = 8$ and $S = 2$. In addition, there are $N + S$ output switches. This allows for an initial set of N normal inputs and S spares.

Each switch j at stage i of the network has three links to stage i - 1. One is connected to switch $(j - 2^i) mod(N + S)$, the second to switch j, and the third to switch $(j + 2^i) mod(N + S)$. The DRN output switches are its output ports. The S spares I/O ports allow S spares of the devices using the network, thus providing fault tolerance for devices. A row of a DRN contains all the switches having the same address, all links incident out of them, and the associated network input and output links. A row has the same address as its switches [44].

When there are no faults, rows 0 to N - 1 are used to emulate the generalized cube network. If a component of row j is found to be faulty, the network is reconfigured so that the switches physically numbered p are logically numbered $t(p)$, where $t(p)$ $= (p - j - S) mod(N + S)$. As long as N adjacent rows remain, the DRN can act as a fault tolerant generalized cube [44].

3.2 Concluding Remarks

In this chapter, some of the recent and important fault tolerant MINs were surveyed. Four methods to enhance fault tolerance to the delta networks; adding an extra stage, adding extra links, chaining of switches within a stage, and adding extra input/output ports. Each method was explained with examples. An explanation the fault tolerant double tree MIN was introduced.

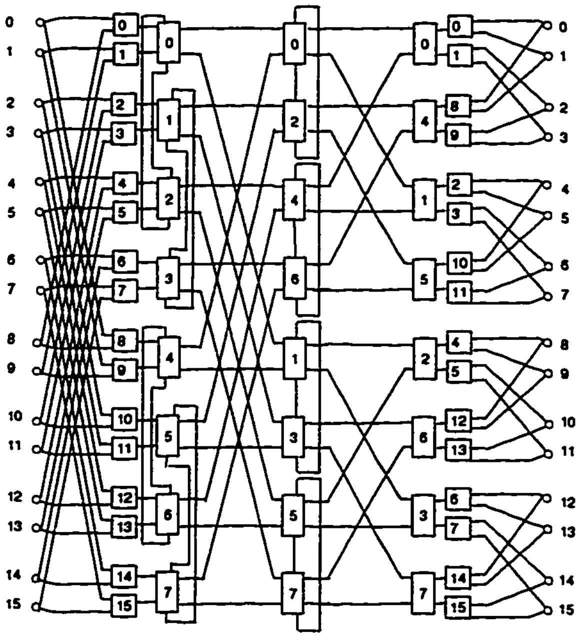


Figure 3.10: ASEN-2 for N = 16.

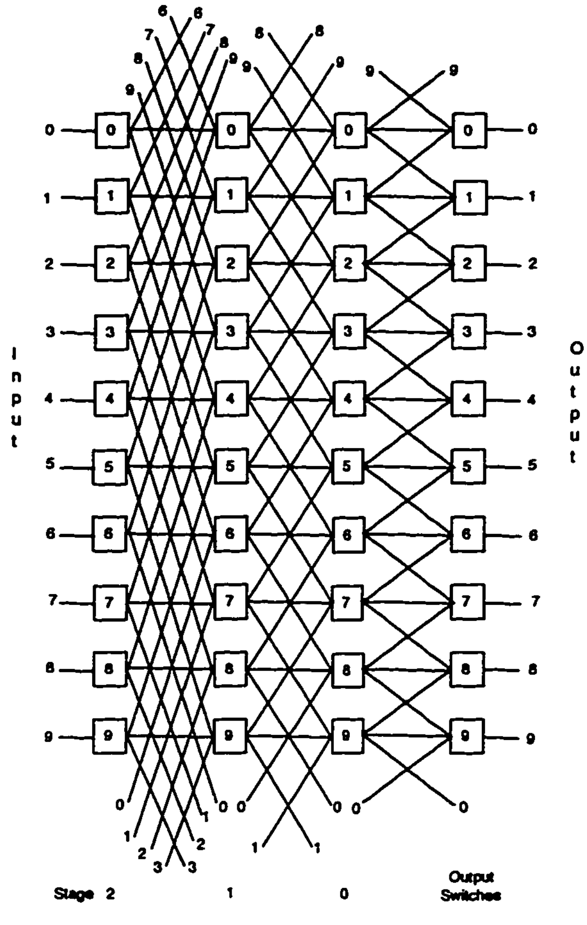


Figure 3.11: A DRN for N = 8 and S = 2.

Chapter 4

FAULT TOLERANCE OF B-ISDN SWITCH FABRICS

The fault tolerance of B-ISDN networks have been studied in recent literature. All the designs require extra hardware to enhance the fault tolerance of the networks. Fault tolerant B-ISDN networks can be classified into three categories. These are: networks which employ extra stage(s) of subswitches such as Itoh [45] and Lo [46] networks, networks which employ multiple copies of the network such as B-tree [47], Tagle [48], and FAUST [49, 50], and networks which employ extra stage and link dilation such as the FDB [51]. In this chapter, a survey of the above B-ISDN fault tolerant networks is presented. Each design is explained and its advantages and disadvantages are pointed out. We conclude the chapter with a comparison between all the above networks.

4.1 Extra Stage(s) of Subswitches Networks

The idea behind this scheme is to add an extra stage(s) of subswitches between the stages of base switches. The addition of subswitches provides the network with more redundant paths. Hence, faults in the switches or links can be tolerated. Two network architectures fall under this category; Itoh network [45] where there are multilevels of subswitches, and Lo network [46] where there is only one level of subswitches.

4.1.1 Itoh Design

The idea behind the Itoh network [45] is the addition of subswitches between stages of the original baseline network as shown in Figure 4.1. The total number of SEs in the network is $N(n-1)+1$ where N is the number of inputs and n is the number of stages. The channel graph of the network is shown in Figure 4.2. Each node in the channel graph represents an SE (either a switch or a subswitch) and each edge represents a link between two SEs in two successive stages.

The subswitches used have ranks which are labelled "R1" for rank-1, "R2" for rank-2, etc. The subswitches are connected to the base switches. If a cell is blocked in a base switch due to an internal contention, a link failure, or an SE fault in the next stage, the cell will be transmitted through the redundant path to a rank-1 subswitch. Then, based on its destination address, the cell will be routed to the next

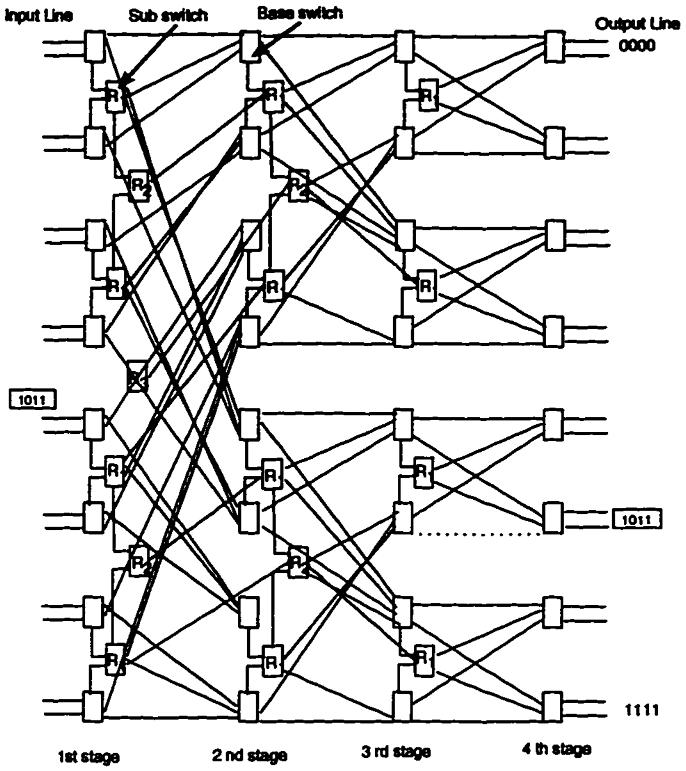


Figure 4.1: The Itoh network.

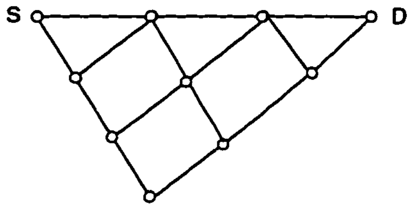


Figure 4.2: Channel graphs for Itoh network.

stage. If the cell is blocked again in the rank-1 subswitch, it will be transmitted through the redundant path of rank-1 subswitch to a rank-2 subswitch and the procedure is repeated again. There are up to rank-(n-i) in the ith stage, where the network size is $N = 2^n$. The routing algorithm used in the network is the self routing algorithm where the bits of the destination address are used to route the cells at each stage [45].

The addition of extra subswitches between the stages will lead to extra available paths in the network. The total number of redundant paths increases as network size increases. For a 16×16 network, the total number of paths available between each input and output pair is 14. The large number of redundant paths contributes to fault tolerance, reliability, and switching performance [45].

The network has four types of SEs: 3×3, 3×2, 2×3, and 2×2. The major part of the network is 3×3 SEs; 2×3 SEs are used in the first stage; 3 ×2 SEs are used in the last stage; 2×2 SEs are used only for subswitches in the first stage [45].

Performance Analysis

An element is considered faulty if any of its components, i.e. the output module, its connecting link, or the input module of the next SE, is faulty. Three assumptions were used in the analysis. First, faults happen in an independent random manner. Second, a network is considered failed when faults prevent the connection of any path between a source-destination pair. Third, the elements which are considered faulty are only those between the first and last stages. That is, the first and last stages are always considered non-faulty [45].

The throughput of the network is evaluated under two assumptions; one is that the cell arrivals at each input port of the network are random and independent; the other is that the input loads are uniformly distributed over all outputs of the network. When an internal cell contention occurs, then a cell will be selected at random as a winner for the link, and the other cell is routed to the redundant path. The maximum throughput the network can achieve is 75% for small network size (n=3). The throughput of the network decreases as the network size increases. The throughput of the network was estimated under faulty SEs. Every SE is assumed to have one buffer and the network has additional buffers at the input ports. The throughput of the network drops as the number of faults increases.

Comments and Observations

There are some comments that should be made about Itoh architecture. The design is a space-redundant, 2D, and self-routing. The network is modular but not regular. The availability of redundant paths between the source and destination makes the network fault-tolerant.

The most important shortcoming of this design is that the design does not preserve the cells order; a crucial requirement in ATM systems. If cells are not received in order, the receiver should have to reorder them. This will put a significant delay on the system and the extra delay is not preferable in ATM. The reason behind the out of order delivery of packets is that paths between each input and output pair have different lengths. The design has 4 different types of switches: 3×3, 3×2, 2×3, and 2×2.

Another problem associated with the switch is that the switches are buffered and they have a complicated architecture and complicated control procedures. The second drawback in the design of Itoh is the fact that the first and last stages of the design are very critical and highly sensitive to faults. Accordingly, there are no disjoint paths available between source-destination pairs. The maximum throughput in the Itoh design is 75% if the switching elements in the first and last stage are assumed to be fault-free.

4.1.2 The Lo and Chiu Design

This architecture is an $N \times N$ network, with N input ports and N output ports. The network is constructed by adding subswitches between base switches of the network. So, each switching stage except the last one contains both base switches and subswitches. Except for the SEs in the last stage (stage Log_2N), each base switch has a chain-in link, a chain-out link, and an extra link to the subswitch, in addition to the original input and output links [46] as Figure 4.3 illustrates.

There are three types of base switches: 3×4, 4×4, and 3×2 and one type of subswitches which is a 2×2. The network requires less elements than Itoh's network

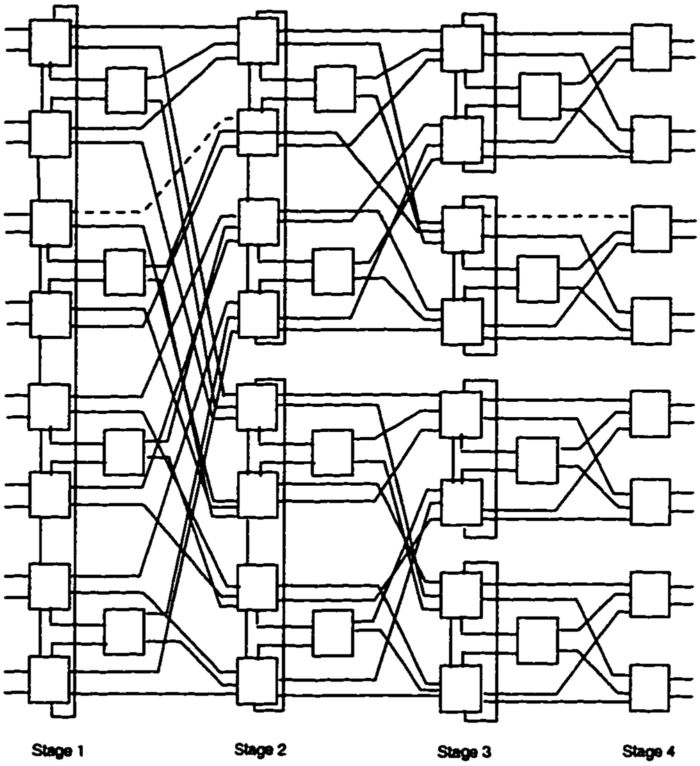


Figure 4.3: The architecture of Lo and Chiu.

because it uses only one extra stage of subswitches in front of each stage of base switches. The total number of redundant paths increases exponentially as the size of the network increases and it is much higher than the Itoh network because of the chaining applied in each stage. The routing algorithm applied is the self-routing algorithm used in banyan networks. If a cell is blocked in a base switch due to an internal contention, a link failure, or an SE fault in the next stage, the cell will be transmitted through the redundant path to the subswitch. Then, the cell will be routed to the next stage based on its destination address. If the extra link to the subswitch is also faulty or blocked, the cell will be routed through the chain-out link to another SE within the chain. The address is again used to route the cell in the new SE. If the new SE is also faulty the cell will be routed to another SE within the chain. The cell can be routed through as many SEs within the chain as required [46].

Performance Analysis

The performance analysis of the network was investigated under the following assumptions. An element is considered faulty if any of its components, i.e. the output module, its connecting link, or the input module of the next SE, is faulty. Five assumptions were used in the analysis. First, faults happen in an independent random manner. Second, a network is considered failed when faults prevent the connection of any path between a source-destination pair. Third, the elements which are considered faulty are only those between the first and last stages. That is, the first and last stages are always considered non-faulty [46]. Fourth, the cell arrivals at each input port of the network are random and independent. Finally, the input loads are uniformly distributed over all outputs of the network. When an internal cell contention occurs, one of the cells will be selected randomly to be transmitted through the link and the other is routed to the redundant path.

The maximum throughput the network can achieve is 81% for small network size (n=4) under workload of 0.8. The throughput of the network decreases as the network size increases. If the network encounters some faulty elements, the throughput of the network decreases as the number of faulty elements increases due to the decrease in the number of redundant paths. Also, the throughput of the proposed network decreases as the traffic load increases, since internal contention is more likely to occur when the traffic load is heavy. Moreover, for the same level of traffic load, the throughput of the network decreases as the network size increases [46].

Comments and Observations

The network is a modular but not regular, 2D, space-redundant, and self-routing architecture. The design has a larger number of redundant paths than the Itoh design. This leads to less sensitivity to the number of faults. The design shows good performance in terms of throughput.

On the other hand, some negative points in the design are listed below. The network has critical first and last stages; i.e. the first and last stages are sensitive to the faults, so any fault in them will lead to a disaster. The delay is variant for the cells depending on the path traversed. The faults in the system leading to the use of subswitches or chains of subswitches will also introduce variant delay. This introduces a problem which is the out-of-sequence delivery of cells. One more problem is that the network requires 4 types of switching elements. Finally, the network needs extra one-level of subswitches and extra links more than the ordinary banyan network.

4.2 Designs with Multiple Copies of the Network

In this category, extra copies of the network are used to enhance the fault-tolerance. There are three designs which fall under this category; the B-tree network [47], FAUST architecture [49, 50], and Tagle network [48].

4.2.1 B-tree Design

The basic structure of an N-input N-output ($N \times N$) network is given in Figure 4.4. It is usually assumed that N is a power of two ($N = 2^n$). Each stage of the B-tree is implemented with $N/2 = 2^{n-1}$ 4×4 SEs. Each 4×4 SE has four inputs, two formal

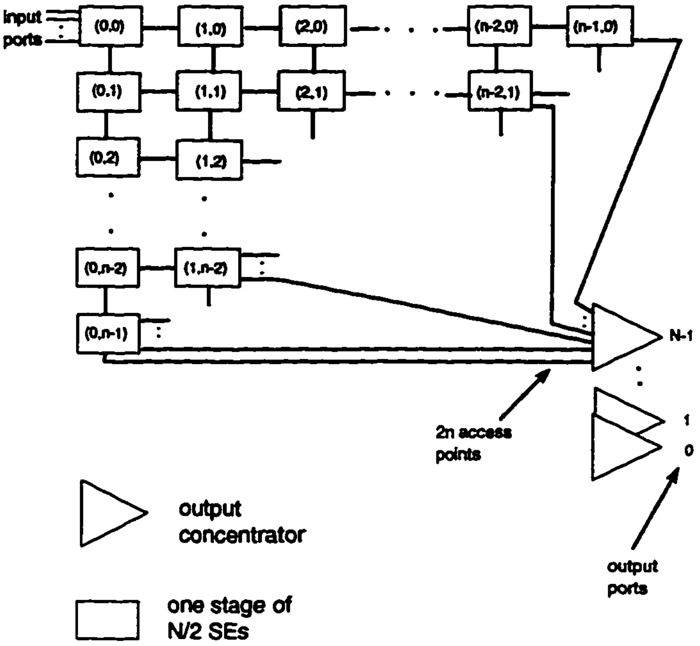


Figure 4.4: Basic structure of the $N \times N$ B-tree.

outputs, and two redundant outputs. There is no buffer employed in an SE of the B-tree.

The B-tree has the property of recursive construction. Thus, the B-tree is also suitable for easy expansion in a modular way. To double the size of the B-tree, two copies of the B-tree network (one below the other) are used. Then, a $2N \times 2N$ baseline network is added to the front of the two B-tree networks. At any stage of the network, the self-routing algorithm of the MINs is applied [47].

There are N=2^n baseline networks in the B-tree which means that there are N links between each input to output pair. Moreover, there are $2n$ access points for each output port. These access points are used to resolve the output contention if several cells are destined for the same output port simultaneously [47].

Performance Analysis

The performance of the B-tree was evaluated under the following assumptions. The arrival rate of cells at each input port of the B-tree is uniform, independent from all other input ports; the input loads are uniformly distributed over all output ports; only the SEs fail and the interconnecting links do not; and the input and output ports and the multiplexers directly connected to the output ports are always assumed to be functional. The throughput of the network is approximately 100% for small network size (n=3). An extremely small decrease in the performance of the B-tree is shown even though the network size increases. If faulty SEs are considered, the network has more than 90% throughput when it has three faulty SEs for networks of size $n \geq 5$. As the switch size increases, the influence of faults on the throughput is negligible [47].

To consider the faults in the first stage of the B-tree, an augmented network is derived from the basic B-tree by adding y baseline networks of 4×4 SEs. The new version of the B-tree is denoted as B-tree(y). Generally, the $N \times N$ B-tree(y+1) network can be constructed recursively with one $N \times N$ B-tree(y) network and one $N \times N$ baseline network. The number of alternative paths between each input and output pair is yN. The performance of the network improves as y increases. The simulation results for switch size $2^n \leq 9$ show approximately 100% throughput for B-tree(2) even up to 3 faulty SEs (with the assumption that SE faults occur randomly among all SEs) [47].

An $N \times N$ B-tree network provides $2n$ access points for each output port. Thus each output port has the capability of accepting up to $2n$ cells simultaneously if no internal conflicts are encountered. So, output contention resolution mechanism is not needed in B-tree [47].

Comments and Observations

The design of the B-tree is a 2D, space redundant and self-routing design. One of the most important positive points in the B-tree architecture is that the order of packets is reserved satisfying the need of the ATM networks. Another positive point is that only one type of switches is needed which leads to a regular design. There are yN alternative paths between each input/output pair for B-tree(y) which give the design high degree of fault tolerance. Also, there are $2n$ access points for each output port to resolve the output contention and increase the reliability of the switching elements in the last stage. Moreover, the B-tree design can grow theoretically to any size.

However, the B-tree has some negative aspects. In the first place, the extra number of switches needed is N times the number of SEs in the original baseline network. Accordingly, the extra number of links used is very large. Moreover, there are N output concentrators which represent extra hardware. In the case of B-tree(y) where $y \geq 1$, N extra multiplexors are needed which again represent extra hardware.

4.2.2 The FAUST Design

FAUST is a multi-stage multi-plane network as shown in Figure 4.5. The switch module could be only a crosspoint switching element or it could be a subnetwork of some kind. Each switch module may consist of a number of switch slices. The number of these slices can be determined based on a number of factors such as the complexity of the logic, the availability of pin-out, and the speed of pin/link. Reasonable values for the slices and modules are 4-16 for the z-dimension and 4-32 for the y-dimension [49, 50].

There are two schemes in FAUST. First is the master-slave scheme where there is one master slice and the rest are slaves as shown in Figure 4.5. The other scheme is the nonmaster- slave scheme in which all the slices are identical with distributed control as shown in Figure 4.6 [50]. Two-dimensional sparing is provided by FAUST; i.e., the z-dimensional sparing (one or two spare slices per switch module) and y-dimensional sparing (one spare switch module per column). In a master-slave design, one spare is provided for the master and another spare is provided for the slaves. For the nonmaster-slave design only one slice is provided for the entire switch module. An important feature of FAUST is that the extra spare components do not modify the core configuration of the system [49].

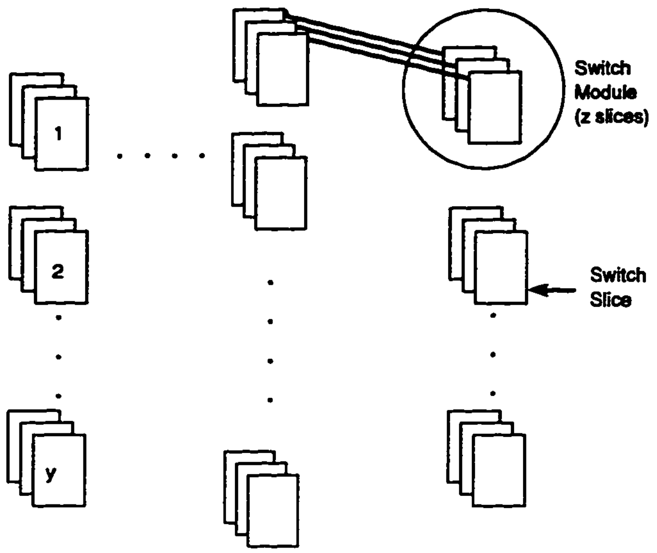


Figure 4.5: Core architecture in FAUST.

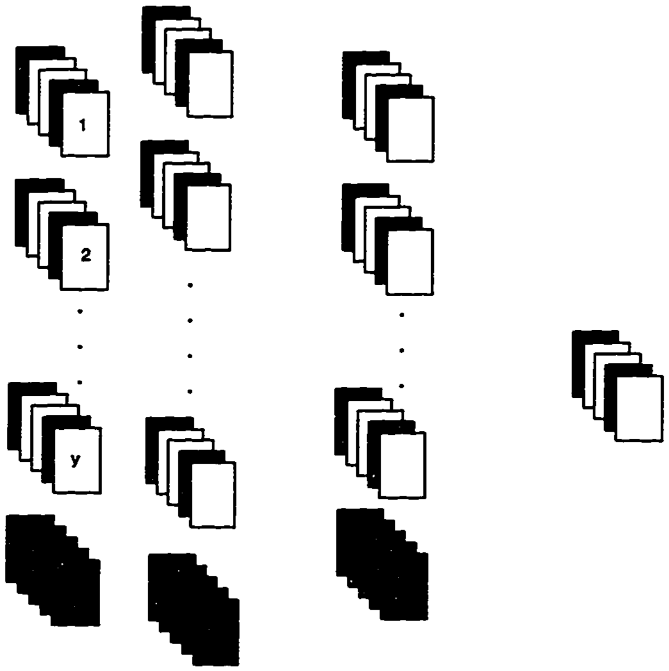


Figure 4.6: Master-slave idea in FAUST.

Fault Tolerance of FAUST

A spare operates in a hot standby mode and the commutation of the spares into the system is done hierarchically as follows:

(1) When the first slice fails, the spare slice in the module will replace it,

(2) When a second slice fails, the module will be declared as faulty and the spare module in the column will replace it. The spare module in the y-dimension can tolerate one faulty slice also. So, FAUST can tolerate three failures of slices in each column and this will give enough time to repair the system. The commutation of the spare slices is done automatically on the fly. The overhead in FAUST is computed as the extra number of spare slices and it is equal to: $(1/y+2/z+2/yz)$ for the master-slave design and $(1/y+1/z+1/yz)$ for the nonmaster-slave scheme [50].

Comments and Observations

The FAUST design is a 3D, space redundant, and self-routing design. The design is general and it can be applied to any type of networks. So, the design can be considered as a platform for fault tolerant ATM networks. The FAUST design preserves the order of packets so its suitability for ATM networks is apparent. Regarding the degree of fault tolerance, the FAUST architecture can tolerate 3 slice failures per column. This provides sufficient time to replace the failed module in most cases before the spares are exhausted. Also, the FAUST design exhibits on the fly distributed reconfiguration of system around failed units.

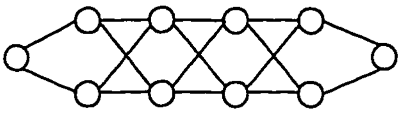


Figure 4.7: Channel graph for the Tagle and Sharma design.

4.2.3 The Tagle and Sharma Design

The network employs two banyan networks, labelled 0 and 1. The difference between this design and the design of parallel banyan network is that in this network the two banyan networks are closely linked. In the parallel banyan network once a cell is sent to a particular plane, it stays in that plane. In this network it is possible to transfer to and from the two planes to bypass faults and conflicts. An 8×8 network is shown in Figure 4.8. It has two complete 8×8 banyan networks in planes 0 and 1 which are represented by plain and shaded SEs, respectively. Plane 1 will be used for fault-tolerance purposes. So, plane 1 would be in a stand-by mode, meaning that it won't come into use until a fault in plane 0 is detected. Thus, the message can be transferred to and from different planes to bypass faults [48].

4×4 SEs are used to link a switch in stage i with 2 SEs in stage $i+1$ in the same plane and 2 SEs in stage $i+1$ in the other plane. A cell always has two paths to choose from in each stage. Moreover, the network allows for fault-tolerance in every stage of the network as can be seen from the channel graph in Figure 4.7. Each node in the channel graph represents an SE and each edge represents a link between two SEs in two successive stages. Furthermore, unlike Itoh's network a cell has to go

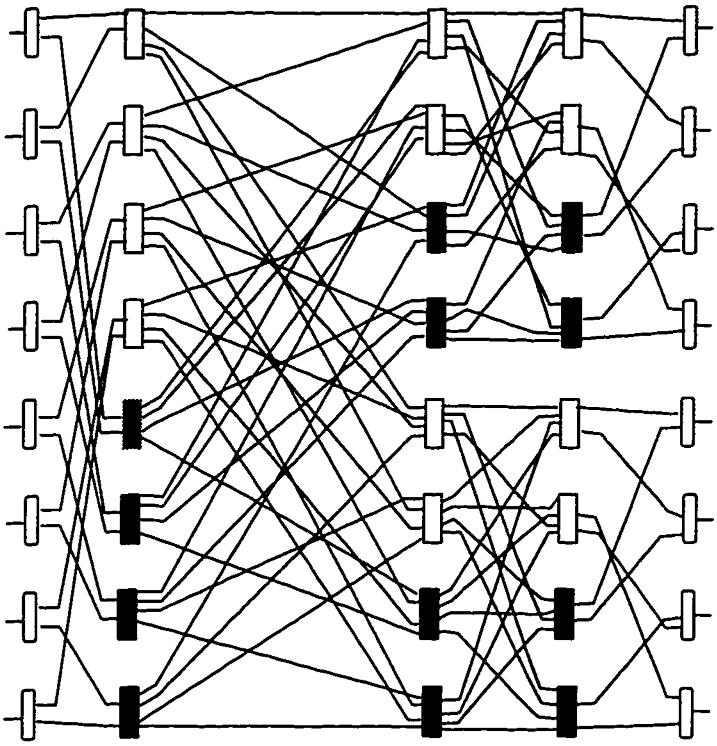


Figure 4.8: An 8×8 version of Tagle network.

through a fixed number of stages to reach an output line whether or not faults are present. The basic routing algorithm is similar to the self-routing algorithm with minor modification. A routing tag is simply the destination address of the output with one more bit appended to the front. This bit specifies the plane at which the cell should be routed to [48].

Performance Analysis

The network was analyzed under the following assumptions. The destination of each cell is generated randomly. No buffers are available in any of the SEs of the network. Hence any cell which can not win the link because of contention is discarded. Moreover, all switch inputs are assumed to be identical and independent. Simulation is done for the network under full load and uniform distribution [48].

Regarding the throughput, simulation of the network shows the following results. The rate of decrease of the throughput of the network is low even with the increase in the network size. When faults are injected randomly in the network, the network shows high performance even in the presence of high number of faults. The proposed network's throughput rate stays above 60% even in the presence of up to 20 faults in the system for network sizes n =6, 8, 10 [48].

Comments and Observations

This design shows some positive and negative points that will be pointed out in this section. This design is a self-routing, 2D, space redundant, and modular design. This design offers two paths for the cell to choose from in each stage. The network offers fault tolerance in every stage of the network and the design can tolerate one fault in the network without any degradation in the throughput of the system. The cell sequence of the input cells is preserved in this architecture.

Some of the negative points of this design is that it uses more complex routing algorithm than the simple routing algorithm used in MINs. The network needs double the number of switching elements of the banyan network. There are $2N$ muxes and demuxes required in the network. The SEs used in the network are the 4×4 SEs and they are used as 2×4 and 4×2 in the first and last stages respectively. The maximum throughput this network can achieve is 65% for small network size (n=3). As the network size increases, the throughput of the network decreases smoothly.

4.3 Networks with Extra Stage and Link Dilation

In this category, enhancement of fault tolerance is done by adding extra stage to the network and by increasing the dilation factor of the switching elements. Hence, extra redundant paths will be available for each source-destination pair. This of course gives the network a degree of fault tolerance. An example of this category is the FDB network [51].

4.3.1 FDB Design

There are three ideas behind the design of the FDB (Fault-tolerant Double-link Buffered) network. First, the links of each switching element is doubled so the network is 1-fault tolerant. Second, buffers are enhanced in the switching elements to be used as a storage for the conflicting cells. Third, an extra stage of SEs is added to the network to make the number of stages $n+1$ where $n=Log_2N$ [51].

There are three different switching elements used in the FDB network. The switching elements used in the FDB network depend on the stage in which they are used. The resulting network is shown in Figure 4.9. In the FDB network, there are two paths: reserved path and backup path. Specific path selection rules are used to select one of the paths in each stage [51].

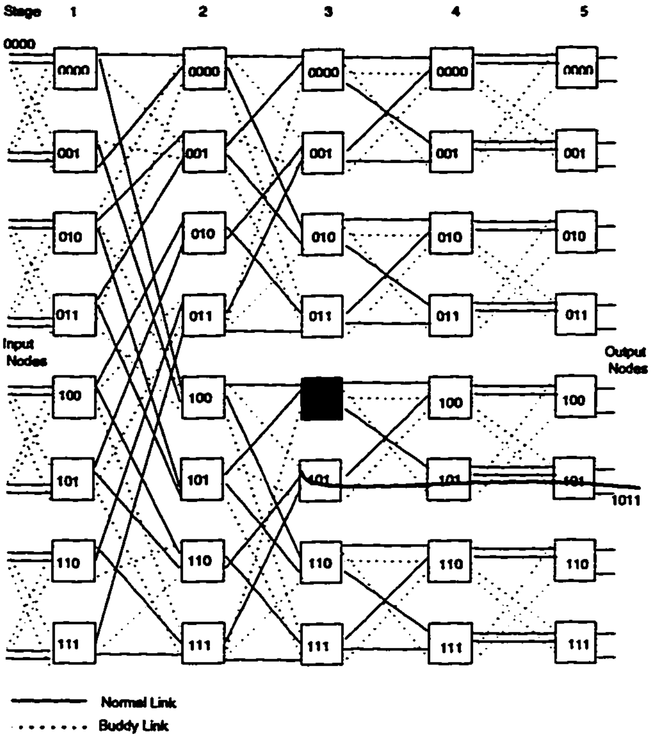


Figure 4.9: An example of the 16×16 FDB network.

Performance Analysis of the FDB Delta Network

The network was analyzed under the uniform traffic load. The following assumptions were made:

(1) Packets are generated with equal probability at each source node,

(2) Packets are of fixed size and are directed uniformly over all of the output ports,

(3) Conflicts are resolved randomly,

The maximum throughput the network can achieve is 87% for a small network size (n=3) with 2 buffers inside each SE. The throughput decreases rapidly as the network size increases. The buffers have a notable effect on the throughput of the network. As the number of buffers inside the SEs increases, the throughput of the network improves.

Comments and Observations

The FDB design is a 2D self-routing architecture which can tolerate one fault only. The order of packets in this design is preserved which makes the network suitable for ATM. The FDB is a modular design which employs space redundancy.

One of the drawbacks is that there are three types of switches in the design. Secondly, the design needs extra stage with all its switching elements and links. Third, the number of buffers inside the switching elements depends on which stage the switching element is located. Finally, the number of links inside the switching elements varies depending on the stage number.

Table 4.1: Comparisons of B-ISDN architectures

Type	Modularity	Routing	Dimensions	Size of Switch(es)
Itoh(1991)	not modular	self-routing	2D	2×2, 3×3, 2×3, 3×2
FDB (1991)	modular	self-routing	2D	4×4(2&4 buffers), 4×2
B-tree(1994)	modular	self-routing	2D	4×4
FAUST(1995)	modular	self-routing	3D	NA
Tagle(1995)	modular	self-routing	2D	4×4
Lo(1995)	not modular	self-routing	2D	3×4, 4×4, 3×2, 2×2

NA : Not Applicable.

Table 4.2: Comparisons of B-ISDN architectures (Continued)

Type	Complexity	Extra SE	Redundancy
Itoh	$N(n-1)+1$	$N(n-Log_2N-1)+1$	space
FDB	$N/2(Log_2N+1)$	$N/2$	space
B-tree(y)	$Nn(n+1+y)/4$	$Nn(n+y+1)/4$	space
FAUST(MS)	$xyz+(1/y+2/z+2/yz)$	$(1/y+2/z+2/yz)$	space
FAUST(NMS)	$xyz+(1/y+1/z+1/yz)$	$(1/y+1/z+1/yz)$	space
Tagle	Nn	$Nn/2$	space
Lo	$5/2N(n-1)$	$N/4(Log_2(n-1))$	space

SE: Switching Element.
MS: Master-Slave.
NMS: NonMaster-Slave.

4.4 Comparison and Observations

In this section, we compare the fault tolerant B-ISDN in many aspects. Tables 4.1, 4.2, 4.3 give the comparison between the different architectures. In Table 4.1, we compare the networks based on their modularity (the ability of building bigger networks out of small networks), routing (algorithm used to route cells from

Table 4.3: Comparisons of B-ISDN architectures (Continued)

Type	Sequence	Degree of faults tolerated	extra disjoint paths	buffering
Itoh	out-of-order	0 fault	zero	yes.
FDB	in-order	1 fault	one	yes.
B-tree($y \geq 2$)	in order	1 fault	one	no.
FAUST	in-order	3 faults	NA	NA
Tagle	in-order	1 fault	one	no.
Lo	out-of-order	0 fault	zero	yes

NA: Not Applicable.

the source to the destination), dimensions (whether the VLSI fabrication of the network is 2D or 3D), and the size(s) of SEs used (types and sizes). In Table 4.2, we compare the networks based on their complexity (the number of SEs used in a network), extra SEs (the number of extra SEs added to a network to make fault tolerant), and redundancy (type of redundancy added to the network to make it fault tolerant). In Table 4.3, we compare the networks based on their way of delivering cells (whether the cells reach destinations in- or out- of-order), degree of faults tolerated (the number of faults between any source-destination pair the network can have and also deliver the cell properly), extra disjoint paths (the number of disjoint paths available between any source-destination pair), and buffering (whether the SEs have buffers inside or not).

From Tables 4.1, 4.2, 4.3, the following observations can be made regarding the fault tolerant B-ISDN networks. Obviously, any fault in the network is critical and can lead to severe problems. So, fault tolerant networks are being proposed to tolerate faults. Most of the work done on fault tolerance is done using MINs with some modifications. There are many reasons for using MINs; namely, self routing, modularity, reasonable cost, etc. More hardware is always required to enhance the fault tolerance aspects of MINs. Some researchers provide extra switches, extra links, or a combination of both. The extra hardware is the trade off for the fault tolerance enhancement.

The number of faults that can be tolerated vary among different designs. Some designs can afford one fault only like the FDB, B-tree, and Tagle networks while some can afford more like the FAUST. In general, the more the number of faults tolerated, the better the network is. This is because the system can have more time before failure. Certainly, the hardware increases as the number of faults tolerated increases. Increasing the number of faults tolerated while having moderate hardware increase is a challenge.

Moreover, preservation of the order of packets is critical and crucial in ATM because real-time applications like video-conferencing require the integration of pictures and voice. In most of the previous research, the order of packets is preserved. On the other hand, some of the proposed networks do not preserve the order of packets like Itoh [45] and Lo [46]. Any future work on the ATM systems should preserve the packets order.

As the dimension of any proposed network increases, the complexity of the VLSI implementation increases. The 3D networks such as the FAUST are more difficult to implement than the 2D networks. The 2D networks are currently preferred because of ease of implementation using today's technology.

Another important factor is the use of buffers inside the switches. Some of the networks mentioned earlier do use internal buffering like Itoh, and FDB networks. Using internal buffering leads to more complex switching elements and more costly networks. The networks that do not need internal buffering could be more cost-effective and faster in handling the packets. A recommendation for the future designs is to try to avoid internal buffering as much as possible.

Finally, the size of switches in the previous networks differs from a design to another. The design of Tagle and the B-tree use only one size of switches which is a 4×4 SE. All the other kinds, except FAUST, use different sizes of switches in the same network. Using the same switch size in the network is important for many reasons. Mainly, the VLSI implementation is easier if one size of switches is used. The recommendation is to try to use one size of switches only to produce cost-effective design.

4.5 Concluding Remarks

In this chapter, a survey on fault tolerant B-ISDN networks was presented. All the designs require extra hardware to enhance the fault tolerance of the networks. Fault tolerant B-ISDN networks were classified into three categories. First, networks which employ extra stage(s) of subswitches like Itoh and Lo networks. Second, networks which employ multiple copies of the network such as B-tree, Tagle, and FAUST. Third, networks which employ extra stage and link dilation like the FDB [51]. Each design is explained and its advantages and disadvantages are pointed out. We concluded the chapter by a comparison between all the networks of interest.

Chapter 5

The Proposed Architecture

In this chapter, we introduce a new high performance switch architecture called the reliable and zealous network (RAZAN). The proposed architecture is deemed suitable for fast packet switching. RAZAN is a 2-D architecture which offers high throughput, low and constant delay, in sequence cell delivery, simple routing, modularity, and fault-tolerance. First, we describe the architectural design of RAZAN. We then explain the routing algorithm used in the proposed architecture. A routing example using the proposed algorithm is also given for an 8×8 network. We conclude with an analysis the cost of the network and compare it with some other existing networks in terms of switching elements and links needed.

5.1 Architecture of Reliable and Zealous Network (RAZAN)

The Reliable and Zealous Network (RAZAN) is an $N \times N$ multistage interconnection network (MIN), with N input ports and N output ports. For simplicity it is usually assumed that N is a power of two ($N = 2^n$ or $n = log_2 N$). RAZAN is based on the ILN (Improved Logical Neighborhood) network [41] [42] first proposed by Mostafa Abd-El-Barr et al. The switching network has n stages of switching elements. Stages are numbered $0, 1, 2, ...n - 1$ starting from the input side. Each stage consists of a column of N switching elements and $(n+1)N$ links. Each switch has $(n+1)$ inputs and $(n+1)$ outputs. Every switching element j is connected to $n+1$ neighboring switches in the next stage. The switches to which switch j is connected are those whose binary addresses differ by at most one bit from the binary address of switch j.

The same procedure is followed in connecting the inputs and outputs of the network to switches in stage 0 and stage $(n - 1)$, respectively. So, each input is connected to all switches in stage 0 whose binary addresses differ by at most one bit from the binary address of that input. In addition, each output is connected to all switches in stage $n-1$ whose binary addresses differ by at most one bit from binary address of that output.

RAZAN has switching elements of size $(n + 1) \times (n + 1)$. An 8×8 RAZAN is shown in Figure 5.1. As evident from the figure, there are 3 stages ($log_2 8$) of switching elements. Each stage has 8 switches. There are 4 links which connect each switch in any stage to 4 neighboring switches in the next stage. Two switches in two successive stages are connected if their binary addresses differ by at most one bit. For example, switch 0(000) in an 8×8 RAZAN network is connected to switches 0(000), 1 (001), 2(010), and 4(100).

An 8×8 proposed network has switching elements with size 4×4. The number of stages the packet should go through in the 8×8 RAZAN is 3 ($log_2 8$). In general for $N \times N$ RAZAN, a packet has up to $(n+1)$ disjoint paths to follow and the number of stages the packet should go through in an $N \times N$ RAZAN is $n = log_2 N$.

5.2 Routing

A simple routing algorithm is proposed for the network. The routing algorithm uses both the destination address and the switch number to decide the link through which a packet is sent to the next stage. Assume that the switches of a given stage are labeled from 0 up to $N-1$, where N is the number of input ports. Each switch is considered as a source labeled $S = S_{n-1}S_{n-2}...S_1S_0$, where $S_{n-1}, S_{n-2}, ...S_1, S_0$ represent the binary address of the switch. Let $D = D_{n-1}D_{n-2}...D_1D_0$ represent the binary address of the destination. Figure 5.2 shows the routing algorithm used at each switch in the proposed network.

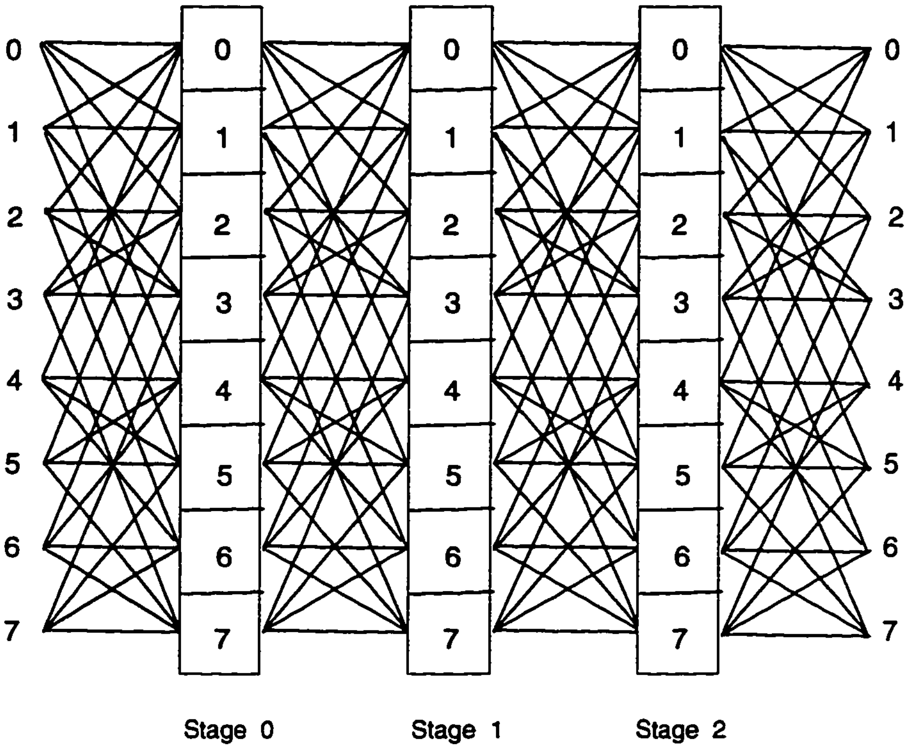


Figure 5.1: An 8×8 RAZAN Network.

Algorithm RAZAN_Route

```
Begin
  Let $S = S_{n-1}S_{n-2}....S_1S_0$     /* source port */
   $D = D_{n-1}D_{n-2}...D_1D_0$     /* destination port */
   $V = V_{n-1}V_{n-2}...V_1V_0$ where $V_i = D_i$xor $S_i$
   $S1 = S$.
   found = false.
    While not (found) do
      if there is an unexamined 1 position then
       Let i be the position of the unexamined 1.
       $D_1 = S_{n-1}...\bar{S}_i...S_1S_0$
       if $D_1$ is not faulty/congested/busy then
          Forward packet to $D_1$
          found= true
       endif.
       else
       $D_1 = S_1$
       if $D_1$ is not faulty/congested/busy then
          Forward packet to $D_1$
          found= true
       endif.
       else
         if there is an i where $D_i = S_i$ then
          $D_1 = S_{n-1}...\bar{S}_i...S_1S_0$
          if $D_1$ is not faulty/congested/busy then
           Forward packet to $D_1$
           found= true
          endif
         endif
        else
         Record packet loss
      endif
    endwhile
end (algorithm)
```

Figure 5.2: Routing Algorithm for RAZAN.

The idea behind the proposed algorithm is to reduce the hamming distance between the source and destination. The hamming distance between any two binary numbers is the number of bit positions in which the two numbers differ [52]. If there are more than one switch in the next stage whose hamming distances from the source are the same, the switch will forward the packet to the switch whose binary address has the minimum hamming distance from the destination's binary address.

For example, suppose that there is a packet at input port 3 (011) that should be forwarded to output port 6 (110). The hamming distance between 3 (011) and 6 (110) is 2. So, the algorithm forwards the packet to SE 7 (111) at the next stage. The hamming distance between 6 (110) and 7 (111) is 1. At the next stage, the algorithm forwards the packet to SE 6 (110). The hamming distance becomes 0. Once a hamming distance of 0 is achieved, the packet will be forwarded through the straight link to the requested destination.

The algorithm RAZAN-Route shown in Figure 5.2 is assumed to be inside each switch in the network and it works as follows. The binary representations of the source and destination addresses are exclusive-ored and designated as V. The algorithm searches for the first occurrence of 1 in the binary representation of V (from left to right). The bit in the binary representation of the source address corresponding to the first occurrence of 1 in V is flipped and the result is designated as $D1$. The packet is then forwarded to the switch in the next stage whose address (in binary) is equal to the address of the source after flipping the bit (i. e. to $D1$).

If for any reason, the link which connects the two switches (S and $D1$) is faulty, congested, or busy, or the next switch ($D1$) is faulty, then second occurrence of 1 in V is searched. Similarly, the corresponding bit of the binary representation of S at the second occurrence of 1 in V is flipped and stored in $D1$.The packet will be forwarded to the switch in the next stage whose address is equal to the address of the source after flipping the bit (i. e. $D1$). This process is repeated if the link between S and $D1$ is faulty, congested or busy or the next switch is faulty until no more 1's remain in V.

In such a situation, the straight link between the source and the destination is tested for not being faulty, congested, or busy. If the link can be used and the next switch to which the packet should be forwarded ($D1$) is not faulty, the packet is forwarded through that link to the next switch ($D1$).

If the straight link is faulty, congested, or busy or the next switch is faulty, then the algorithm searches for the first occurrence of 0 in the binary representation of V. The corresponding bit in the binary representation of the source address to the first occurrence of 0 in V is flipped and the result is designated as $D1$. If the switch in the next stage which has the address of the source after flipping the bit ($D1$) is not faulty and the link is not faulty, busy, congested, then the packet is forward to it. If a fault is encountered in the switch or the link is faulty, busy or congested, then the same procedure is repeated for the next 0 in V. This process will be repeated till no more 0's remain in V. In this case, the packet is considered lost because all links have been exhausted.

5.3 Example

To illustrate the algorithm Route, an example for routing between source and destination in an 8×8 network is given. Suppose that a packet at input port 0(000) should be routed to a destination 6(110). So, S = 0(000) and D = 6(110). Then, V = (000) xor (110) = (110). Let us assume that there are no faulty switches in this networks. Since, the leftmost bit in V is equal to 1, the leftmost bit in the source is flipped. The leftmost bit in the source is 0 and it is flipped to 1 which will give $D1$= 4(100). This means that the packet should be forwarded to switch 4(100) in the first stage as shown in Figure 5.3. Now, the new source address is 4(100) and the destination is 6(110). So, S= 4(100) and D= 6(110). The exclusive or between S and D is performed and the result is stored again in V which is equal to 2(010). Since the second leftmost bit in V is 1, then the second leftmost bit in S(100) is flipped. So, $D1$ = 6(110) and switch 4(100) should forward the packet to switch 6(110) in the next stage (stage 2) under the assumption that the path between the switches is not faulty, busy, or congested.

At switch 6(110) in the second stage, the values of S and D are both 6 and the exclusive-or operation between them yields V = 0(000). Since we have no 1's in V, the straight link between the two switches is used if it is not faulty, busy, or

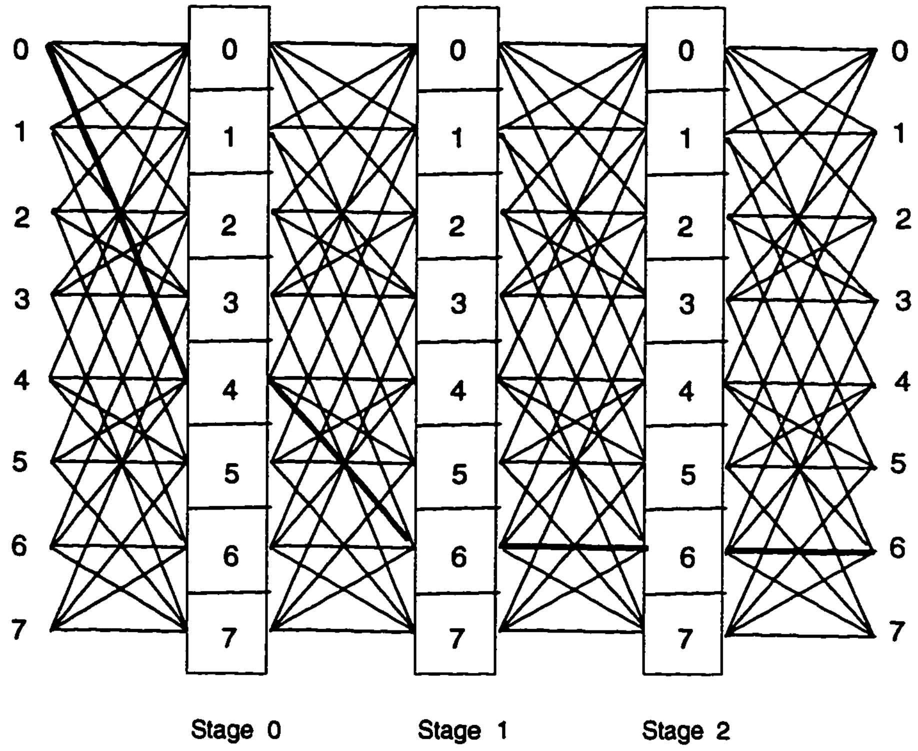


Figure 5.3: Routing Example.

congested. Again, at stage 3 the source and destination are both 6(110) and the straight link is used to forward the packet to the destination if the link is not faulty, busy, or congested. Figure 5.3 highlights the path that the packet should go through from 0 to 6. Note that the algorithm tries to deliver the packet as soon as possible to a SE which has an address equals to the requested destination address. This could happen at some early stages in the network after which the packet will follow the straight path to its destination.

5.4 Cost Analysis

In this section, we seek to compare the amount of hardware needed to implement the proposed network vis-a-vis Benes network, parallel banyan and Tagle's network [48]. The comparison will be made in terms of the complexity of each network and the number of links used in each network. To come up with equations for the complexity for Benes, parallel banyan, Tagle, and RAZAN networks, we assume that all the switching elements used are crossbar switches. So, the number of crosspoint switches in a $k \times k$ SE is k^2. We define the complexity of a network as the number of crosspoint switching elements required by that network.

The complexity of RAZAN can be deduced as follows. From Figure 5.3, for an $N \times N$ network, there are $log_2 N + 1$ stages of links and each stage has N SEs. Each SE in the network has $(log_2 N + 1)$ links. Therefore the total number of links in

RAZAN is $(log_2N + 1) \times (log_2N + 1) \times N = N(log_2N + 1)^2$.

The following formulae show the complexity of each of these networks.

$Complexity_{Benes} = 4(Nlog_2N - N/2)$

$Complexity_{Pbanyan} = 4Nlog_2N$

$Complexity_{Tagle} = 16Nlog_2N$

$Complexity_{RAZAN} = (log_2N + 1)^2Nlog_2N$

We see from the above equations that for $n \geq 5$, RAZAN becomes more complex than Tagle, and hence more complex than the other networks. Next, we compute the number of links for all four networks under consideration. The number of links of Benes, parallel banyan, Tagle [48], and RAZAN networks are as follows:

$links_{Benes} = 2N(log_2N - 1)$

$links_{Pbantyan} = 4N(log_2N - 1)$

$links_{Tagle} = 8N(log_2N - 1)$

$links_{RAZAN} = N(log_2N + 1)^2$

Figure 5.4 illustrates a comparison between the four networks in terms of the number of links used in each. The higher number of links in RAZAN implies higher complexity compared to others. On the hand, the higher number of links in RAZAN leads to higher throughput and higher fault tolerance among the aforementioned networks as will be shown in the following chapters.

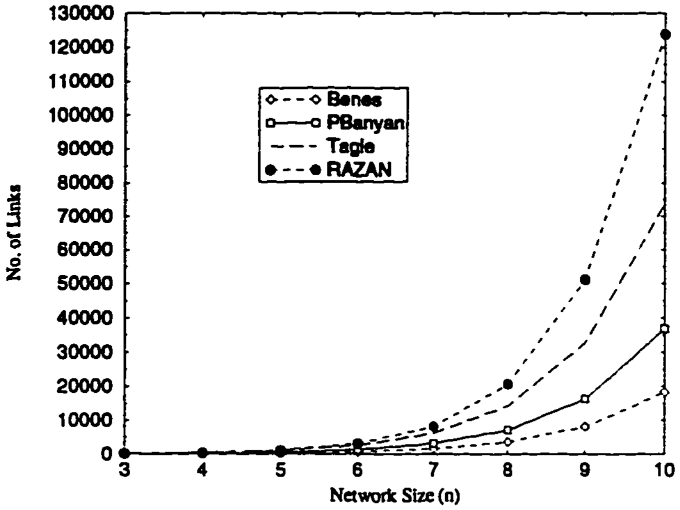


Figure 5.4: Number of Links in each Network.

5.5 Concluding Remarks

In this chapter, we proposed a new high performance switch architecture called RAZAN. The switch is mainly for fast packet switching. The architecture of the proposed switch was explained in detail. The simple routing algorithm adopted for the switch was fully discussed. An example to show how the algorithm works for an 8×8 network was also provided. Finally, we analyzed the cost of the proposed network and we compared it with other known networks; namely, Benes, parallel banyan, and Tagle networks. It has been shown that the complexity of RAZAN increases with its size. The other networks have fixed complexity for any network size. Moreover, the number of links used in RAZAN is higher than those used in the other three networks.

Chapter 6

Performance Evaluation of RAZAN

In this chapter, we present an analytical model for the throughput of RAZAN obtained under uniform workload. Then, the throughput obtained through computer simulation under uniform workload is presented for RAZAN assuming fault-free conditions. The results obtained for the throughput using the analytical model and the simulation program are compared. Finally, the throughput obtained from simulation of RAZAN is compared with that of Benes, parallel banyan, and Tagle networks.

6.1 Analytical Model of RAZAN

In this section, we derive the analytical expression for the throughput of RAZAN assuming uniform traffic at switch inputs. All destinations are assumed to be equally likely. We also assume that all the inputs are identical and totally independent of each other. To derive an expression for the analytical model, we refer to Figure 6.1. For an $N \times N$ network, the SEs are of size n+1 × n+1.

Let SE be a particular n+1 × n+1 switch at the ith stage. Let p_i be the probability that a cell is present at a given input of that switch (see Figure 6.1). Under uniform workload, each input will have the same cell presence probability p_i. Let p_{i+1} be the probability of cell presence at a particular output of the switch under uniform workload. All outputs will have the same cell presence probability p_{i+1}. Next, we derive the relationship between p_i and p_{i+1}.

Let j be a particular input of a switch at stage i, $1 \leq j \leq n+1$ and $1 \leq i \leq n$. Under uniform workload, the probability q_{jk} that the cell chosses a particular output k, $1 \leq k \leq n+1$, is

$$q = \frac{p_i}{n+1}. \tag{6.1}$$

The above probability is the same for any pair (j, k), that is

$$q_{jk} = q = \frac{p_i}{n+1}, \qquad 1 \leq j, k \leq n+1. \tag{6.2}$$

Therefore,

p_{i+1} = probability[at least one of the input cells selects that particular output]

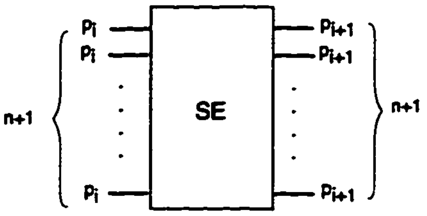


Figure 6.1: The n+1×n+1 SE used in RAZAN

= 1 - probability[none of the $n+1$ input cells select that particular output]

Hence,

$$p_{i+1} = 1 - (1 - \frac{p_i}{n+1})^{n+1} \tag{6.3}$$

The equation above represents the throughput of one stage of the network. To find out the overall throughput of the network, the equation above should be applied recursively from stage $i = 1$ to stage $i = n$, where n ($log_2 N$) is the number of stages in the network.

Let t_i be the throughput of stage i of RAZAN. The throughput of RAZAN for stage i is

$$t_i = (n+1) \times p_{i+1} \tag{6.4}$$

The equation above should be applied recursively from stage $i = 1$ to stage $i = n$ to get the overall throughput of the network.

Now, we give an example of the analytical model above for an 8×8 RAZAN network. If we assume that the workload of the network is $p = 1$, i.e. full load, then each link has a cell with probability $p_1 = 0.25$. So, the throughput of the first stage of RAZAN is as follows:

$$p_2 = 1 - (1 - \frac{0.25}{3+1})^{3+1} = 0.23 \tag{6.5}$$

Now, $p_2 = 0.23$ is the input for the second stage of RAZAN network and equation (1) is applied again where the result will be $p_3 = 0.21$. The procedure will be the same for the last stage and $p_4 = 0.194$. To get the overall throughput, we apply the equation 6.4 above to obtain,

$$t_4 = (3 + 1) \times 0.194 = 0.77 \tag{6.6}$$

6.2 Simulation of RAZAN

In this section, we present the throughput performance of RAZAN through computer simulation. The results are compared to that of Benes, parallel banyan, and Tagle networks. All results shown are under uniform workload distribution. We assume that the destination of each cell is generated randomly. No buffers are available for any of the SEs of all the networks. Hence any cell which loses the link because of contention is discarded. Moreover, all switch inputs are assumed to be identical and independent.

We designed a generic simulation program for any network size. The program has the following data structures.

N: The number of inputs and outputs of the network *m*: The number of link-stages used in the network ($m = log_2 N+1$)

loop: The number of iterations

No-Faults: The number of faults in the network

link-status[N][N][m]: A matrix which has the status of every link in the network(1 for used/faulty link and 0 for available link)

A pseudo-code like representation of the main routine of the simulation program is shown below.

RAZAN Simulator

```
Begin
  Initialize-Random-Number-Generator;
  For (i = 0, i ≤ loop, i++)
    BeginFor
      Intialize-Values;
      Generate-Random-Numbers(N);
      Inject-Fault (No-Faults);
      Route;
      Check-Outputs;
    EndFor
  Calculate-Throughput;
End
```

It consists of a set of function calls, each of which has a specific task. At the beginning of the program, the function Initialize-Random-Number-Generator is used to initialize the random number generator. Next, the simulation program enters a loop which has up to *loop* iterations. In each iteration, the function Initialize-Values is used to initialize all the variables and matrices. Next, the function Generate-Random-Number is used to generate N numbers which represent the destinations of cells. The function Inject-Fault is used to inject up to No-Faults faults in the network randomly distributed over the entire network. The function Route is then used to route the numbers generated at the inputs to the required destinations using the algorithm RAZAN Route shown in Figure 5.2. Then, the function Check-Outputs is used to calculate the number of cells that could not reach the required destination (i.e. cell loss). After the loop, the function Calculate-Throughput is used to find the throughput of the network by subtracting the cell loss from the total number of cells fed to the network divided by the total number of cells fed to the network.

The simulation program was written in C language and was run under UNIX. The simulation program was fed by 10^7 cells for each run and for each network size. Simulation is done for network sizes n=3 to n=10. The offered load for all the networks is 1; i.e. a full load. The results are shown in Figure 6.2.

Clearly shown in the figure is that, all three networks (Benes, parallel banyan, and Tagle) exhibit a decrease in throughput for increasing network size. RAZAN has approximately stable throughput even with the increase in the network size. RAZAN shows a very high throughput compared to the other networks. For large

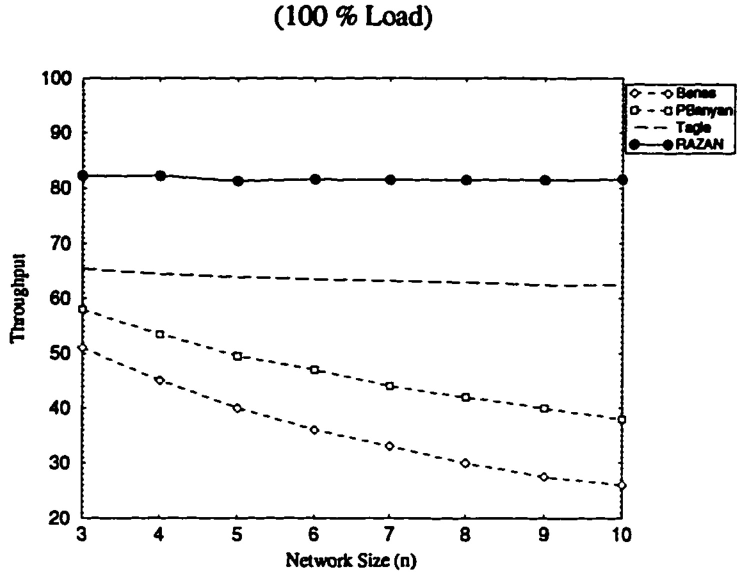


Figure 6.2: Throughput vs Network Size.

network sizes (n=8, 9, 10), the throughput of RAZAN is higher than the throughput of Tagle network by more than 18%.

A high throughput is one of the most important requirements for a high performance network if it is to handle B-ISDN. A low throughput means cells would have to be sent again. This leads to an increasing amount of delay and network traffic. RAZAN fulfills the high throughput requirement of B-ISDN.

6.3 Comparison Between the Analytical and Simulation Results

Figure 6.3 shows a comparison between the analytical and simulation results. The difference between the analytical model results and simulation results increases as the network size increases. A reason for this deviation is the independence between cells at all stages assumed in the analytical model. In the simulation, cells do depend on each other because a cell should take another path if the required path is used by another cell. We also assumed that all the SEs inputs are independent at each stage. In reality, this is not true since the routing from any SE in any stage depend on the routing used in the previous stage. This means that some of the SEs inputs will be busy or idle depending on the SEs in the previous stage.

Moreover, the inputs of a single SE also depend on each other. This is because a cell will be routed to the other input if the requested input of the SE is busy. Each cell, in reality, has a different path to follow from the source to the required destination. The simulation captures this feature while the analytical model does not.

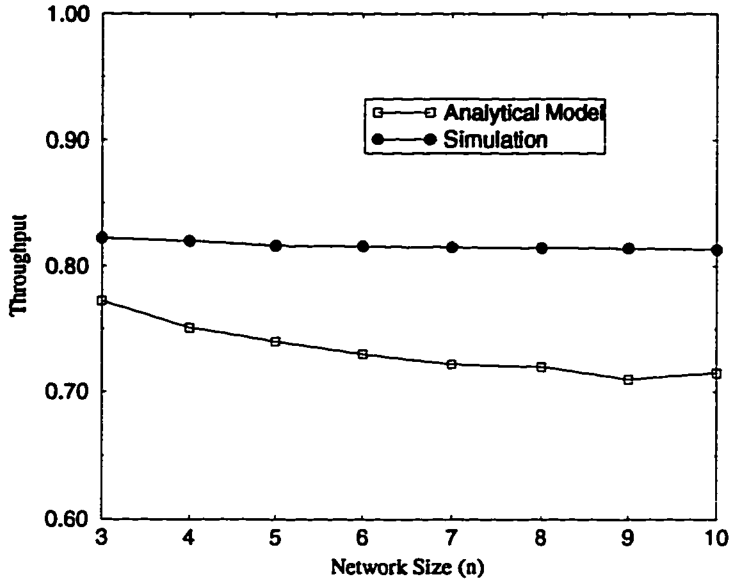


Figure 6.3: Comparison Between the Analytical and Simulation Results.

6.4 Concluding Remarks

In this chapter, we have evaluated the throughput performance of RAZAN analytically under uniform traffic. Then, the simulation results of RAZAN were presented in comparison with Benes, parallel banyan, and Tagle networks. RAZAN showed a superior throughput compared to other networks. At the end of the chapter, we presented a comparison between the throughput computed with the analytical model and simulation. The comparison indicated that the analytical model is a very good approximation of the behavior of RAZAN.

Chapter 7

Fault Tolerance Aspect of RAZAN

In this chapter, we evaluate the fault tolerance aspects of RAZAN. First, we find the number of redundant paths available for an $N \times N$ RAZAN network. Then, we compare that number with those of other known networks such as Benes, parallel banyan, and Tagle. We then find the number of disjoint paths of RAZAN and compare it to that of Benes, parallel banyan, and Tagle networks. The number of disjoint paths in a network is a measure of its fault tolerance. A network with m disjoint paths can tolerate up to m-1 faults between source and destination. Such network is called $(m$-$1)$-fault tolerant.

7.1 Redundant Paths

The number of redundant paths available between a source and destination affects system fault tolerance. A major drawback of banyan networks is that they have only one path between any source-destination pair. So, if two packets contend for the same link, one of the packets will be dropped. The throughput of the system therefore drops. The situation becomes much worse if any switch of the banyan network fails. This means that some of the output ports will be isolated and all packets destined to the device(s) connected to these ports will be lost. To overcome this problem, many architectures have been proposed in the literature [17, 18, 39, 40, 41, 33, 45, 46, 48, 51, 53, 54].

We will compare the number of redundant paths of Benes, parallel banyan, and Tagle to those of RAZAN. Assume that the number of redundant paths of any network is $R_{NETWORK}$. The number of redundant paths for the first three networks are taken from [48] and are as follows:

$$R_{Benes} = 2^{n-1} \tag{7.1}$$

$$R_{Pbanyan} = 2 \tag{7.2}$$

$$R_{Tagle} = 2^{n} \tag{7.3}$$

A lower bound for the number of redundant paths for RAZAN is $(n+1)!$, that is,

$$R_{RAZAN} \geq (n+1)! \tag{7.4}$$

Proof:

Let $Y_{SD} = SE^0_{i0}, SE^1_{i1}, \ldots, SE^j_{ij}, \ldots, SE^{n-1}_{i(n-1)}$ be a path from a source input S to destination D ($0 \leq D \leq 2^n - 1$). Y_{SD} is a legal path from S to D iff:

(a) $0 \leq i_j \leq 2^n - 1$

(b) $d(i_j, D) \leq n - j$

(c) $0 \leq i \leq n - 1$

(d) $0 \leq j \leq n - 1$

where $d(x, y)$ is the hamming distance between x and y.

For any RAZAN network of size $N \times N$, the worst case is from $S = 0$ to $D = 2^n - 1$ (refer to Figure 7.1). From 0, the cell can go to any SE connected to 0 which are 0, 2^0, 2^1, 2^2, ..., 2^{n-1}. This means that there are exactly $n + 1$ paths available for the cell to choose from at the input stage. At the first stage (stage 0), all the SEs are connected to SE 0 in the second stage (stage 1) because there is only one bit difference between these SEs and SE 0. But, SE 0 in stage 1 does not lead to destination $2^n - 1$. This is because the hamming distance between SE 0 and SE $2^n - 1$ ($d(0, 2^n - 1) = n$) is greater than the number of remaining stages which is $n - 1$. All the remaining SEs in stage 1 have legal paths to destination $2^n - 1$ because the hamming distance is less than or equal to the number of remaining stages.

Therefore, each SE of stage 1 connected to source 0 will have only n links that can lead to the required destination. The same procedure is followed in each of the remaining stages. So, at each stage, the number of links available for each SE to

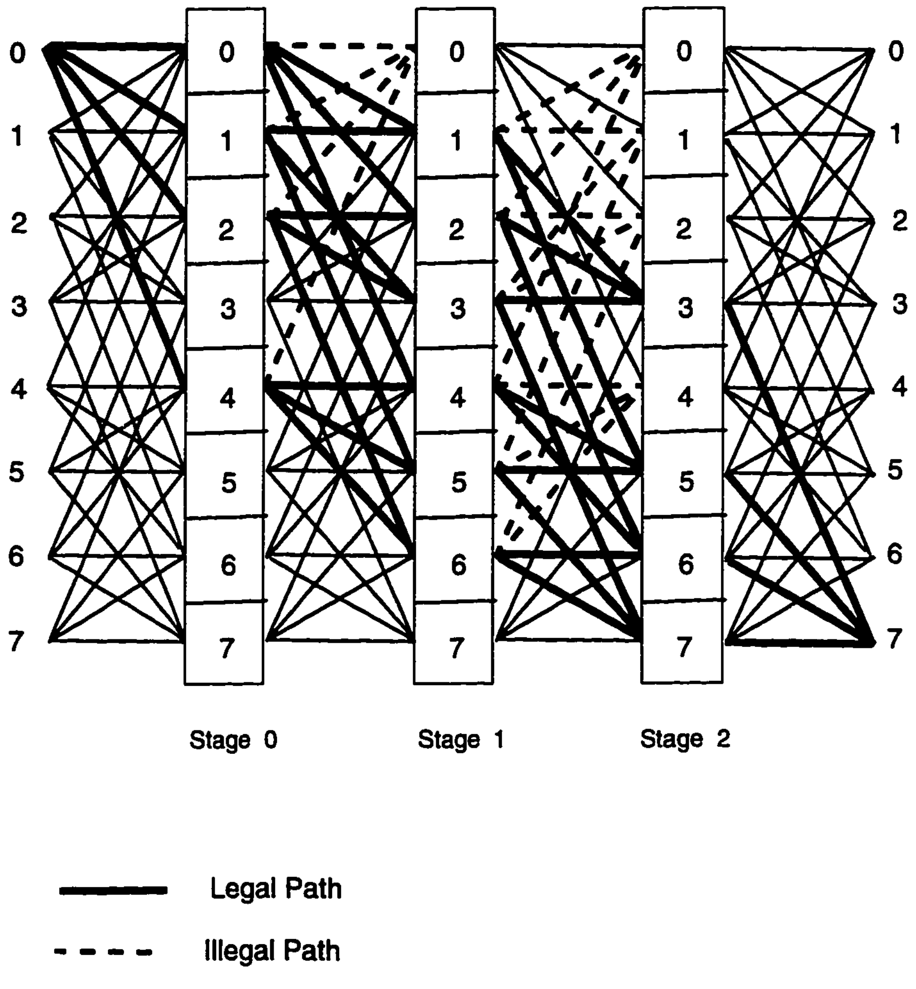


Figure 7.1: Legal and illegal paths available in an 8×8 RAZAN.

Table 7.1: The number of redundant paths for Benes, Parallel Banyan, Tagle, and RAZAN

Network Size (n)	Benes	Parallel Banyan	Tagle	RAZAN
3	4	2	8	24
4	8	2	16	120
5	16	2	32	720
6	32	2	64	5040
7	64	2	128	40320
8	128	2	256	362880
9	256	2	512	3628800
10	512	2	1024	39916800

the required destination is decresed by 1, i.e (n+1), n, (n-1), ... Therefore, the number of paths available from $S = 0$, to $D = 2^n - 1$ is

$$R_{RAZAN}(0, 2^n - 1) = (n+1)! \tag{7.5}$$

The above is a lower bound on the number of paths from any source-destination pair. Table 7.1 evaluates the above equations for various values of switch sizes. The very large number of redundant paths in RAZAN leads to a very highly reliable network. Also, a high number of redundant paths reduce the possibility of packet loss which therefore increases the throughput of the system.

7.2 Disjoint Paths

In this section, we will compute the number of disjoint paths available from any source to any destination in RAZAN and compare it to those of Benes, parallel banyan, and Tagle networks. The number of disjoint paths available from source to destination is a measure of the degree of fault tolerance of a given network. In general, a network which has m disjoint paths has a fault tolerance of degree m-1. In other words, the network is *m-1* fault-tolerant.

One of the interesting characteristics of RAZAN is that the number of disjoint paths increases as the network size increases. Hence, the fault tolerance of RAZAN improves as the network size increases. Generally, there are n+1 disjoint paths in a RAZAN of size n, where $n = log_2 N$. For n =10, there are up to 11 disjoint paths which makes the network 10-fault tolerant. This means that if we have up to 10 faults on the path between a source and a destination, RAZAN can still deliver the packet to its required destination successfully. Figure 7.2 shows the disjoint paths of an 8×8 RAZAN network between source 0 and destination 6. The channel graph of an 8×8 RAZAN is shown in Figure 7.3(d). Each node in the channel graph represents a SE and each edge represents a link between two SEs in two successive stages.

Parallel banyan has two disjoint paths between a source and a destination which makes it a 1-fault tolerant network. The problem with parallel banyan is that fault tolerance offered only at the first stage after which the packet has only one single path to reach its destination as can be seen in Figure 7.3(a). So, parallel banyan does not provide a solution to the banyan network single path weakness as each plane in just a baseline network.

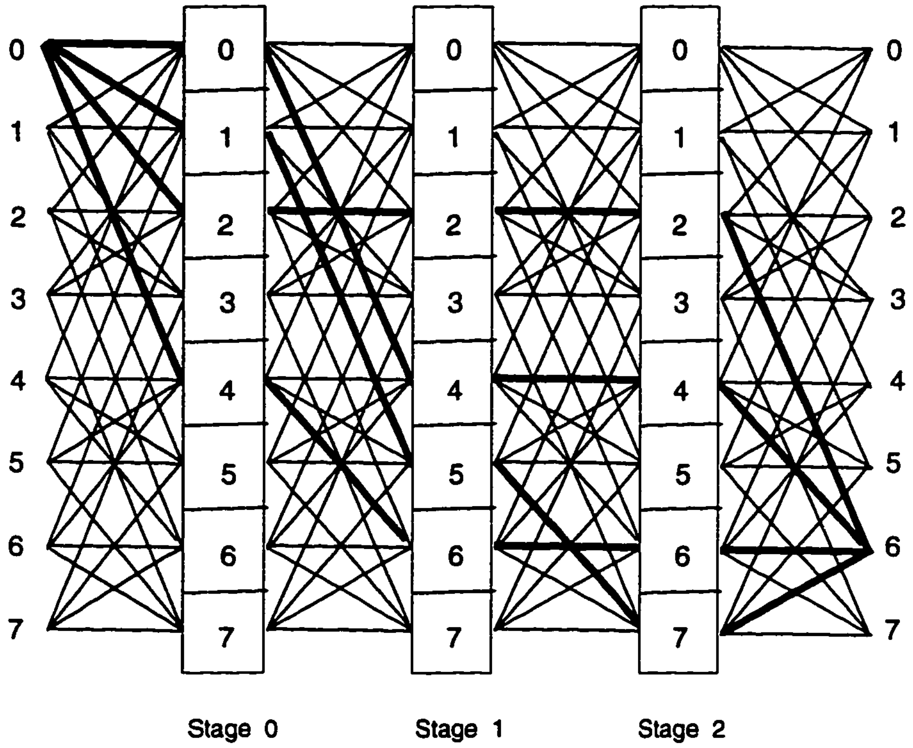


Figure 7.2: Disjoint Paths Available in an 8×8 RAZAN.

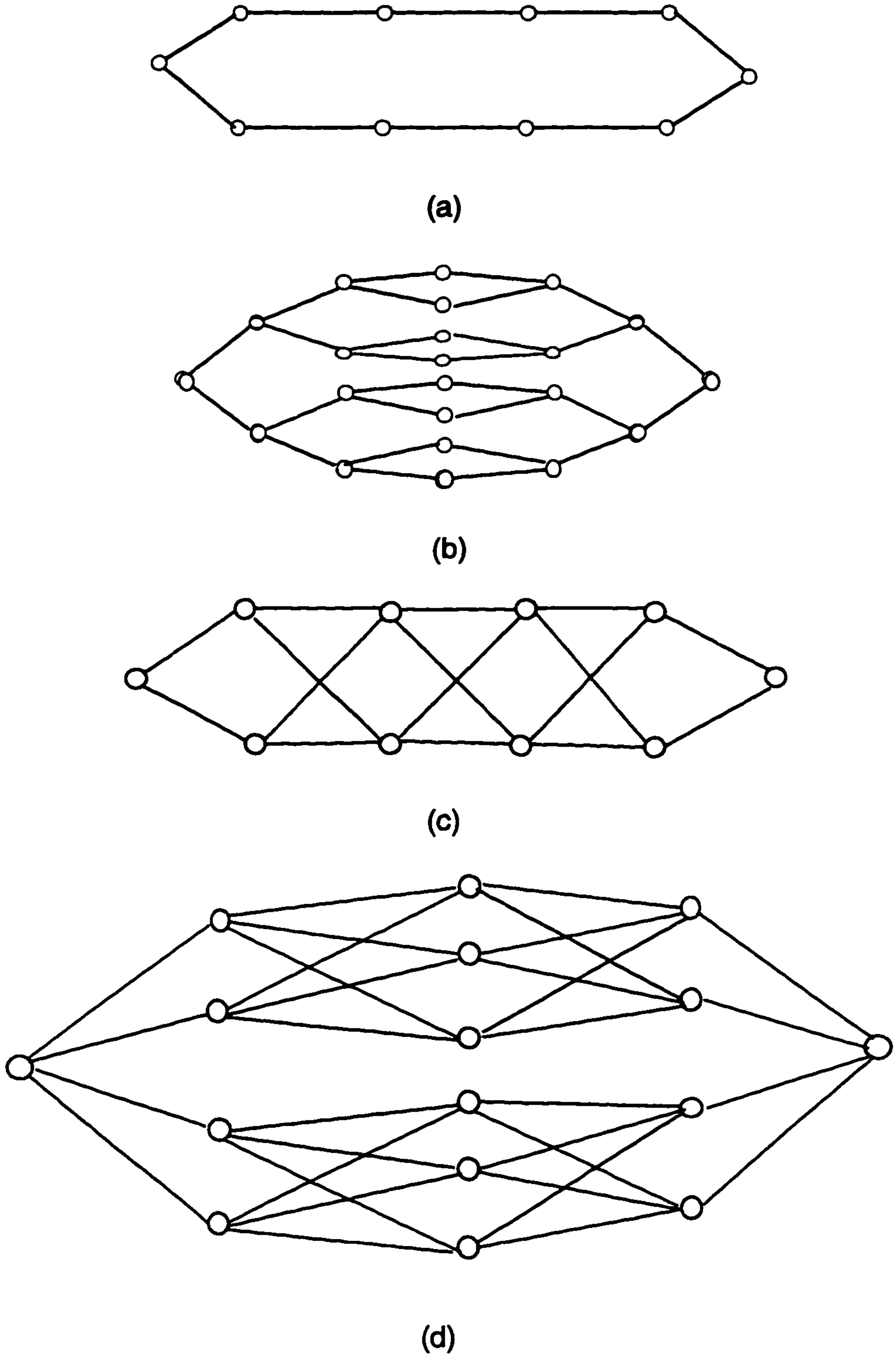


Figure 7.3: Channel graph for (a) parallel banyan (b) Benes (c) Tagle (d) 8×8 RAZAN.

Benes network eventhough it has $2n$-1 stages for the $n \times n$ network, it has only two disjoint paths between a source and a destination. So, Benes network is 1-fault tolerant. Moreover, a close examination of Benes network shows that fault tolerance is only available till the middle stage after which a packet has no choice but to follow a single path to its destination. This is clarified by the channel graph in Figure 7.3(b).

Tagle's network which has been proposed recently is based on banyan network but provides multiple paths at each stage of the network. Fault tolerance is offered at all stages since a packet always has two paths to choose from in each stage as shown in the channel graph in Figure 7.3(c). The number of disjoint paths in Tagle's network is two which makes the network single-fault tolerant.

To construct the channel graph of an 8×8 RAZAN network, we proceed as follows. Assume that a cell has to go from source 0(000) to destination 6(110). At the first stage, the cell has four links to go through. If the cell chooses the upper link then from stage 0 the cell has three links to choose from, where each of them will reach destination 6 as shown in Figure 7.4. From stage 1, the cell always has two links to choose from where each of which reaches the destination required. At the last stage (stage 2), the cell should go through only one link to the requested destination. Figure 7.3(d) shows the channel graph of an 8×8 RAZAN network.

Figure 7.5 shows the number of disjoint paths versus network size for all the four networks. The figure shows the superiority of RAZAN in the aspect of fault

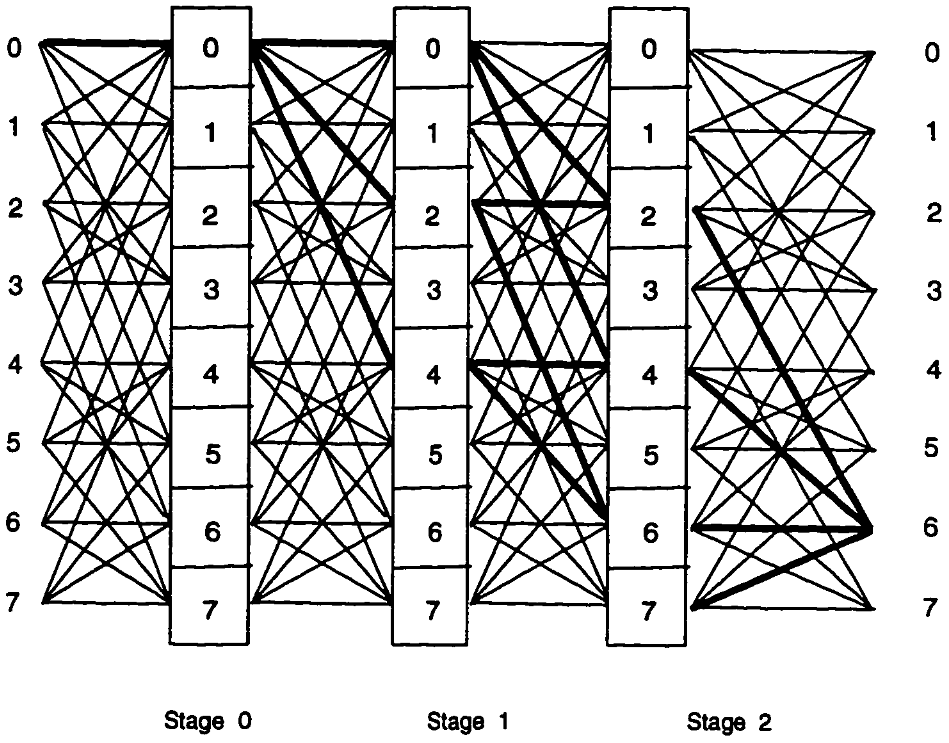


Figure 7.4: Construction of an 8×8 channel graph of RAZAN.

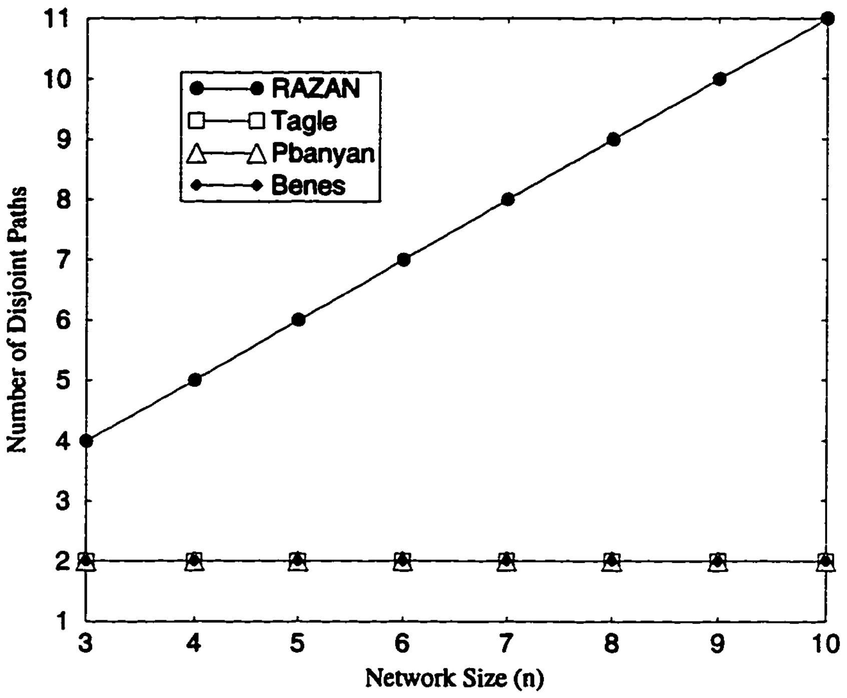


Figure 7.5: The number of disjoint paths vs network size.

tolerance over the other networks.

7.3 Concluding Remarks

In this chapter, we computed the number of redundant paths available in RAZAN. The results were compared with those of three other networks, mainly, Benes, parallel banyan, and Tagle. RAZAN has shown a high superiority in the number of redundant paths available between any source and destination compared with the three other network. The number of disjoint paths between a source-destination pair was also computed. As the network size of RAZAN increases, the number of disjoint paths also increases. The other three networks have a fixed number of disjoint paths, mainly, two. So, RAZAN has the highest fault tolerance compared to the other networks under consideration, Benes, parallel banyan, and Tagle networks.

Chapter 8

Reliability Analysis of RAZAN

In this chapter, we analyze the terminal reliability of RAZAN and compare it to the terminal reliability of three networks: Benes, parallel banyan, and Tagle. We will derive a formula for the terminal reliability of each network. We then estimate the complexity of RAZAN switching elements (SEs) in term of area required relative to the area required by the 2×2 SE. Numerical results of terminal reliability are provided for RAZAN, Benes, Tagle, and parallel banyan.

8.1 Terminal Reliability

The Terminal reliability of a MIN can be defined as the probability of having at least one available path between any designated source-destination pair [41]. If we assume that the probabilities of failure of different components of the system are independent, we can use the general relations for the terminal reliability of series and parallel subsystems in computing the terminal reliabilities of parallel banyan, Tagle, and RAZAN networks. For a system with C components, 1,2,3,.. C, we have,

$$R_{series} = R_1 R_2 .....R_C \tag{8.1}$$

$$R_{parallel} = 1 - (1 - R_1)(1 - R_2)....(1 - R_C) \tag{8.2}$$

For reliability analysis, only SE-Faults are considered; links are assumed to be highly reliable. So, we shall assume that all the links reliabilities are equal to 1 (i.e. $R_l = 1$). We shall use the following notations for all networks of interest.

TR: Terminal reliability,

R_Y: Reliability of a path,

$R_{k \times k}$: Reliability of switch of size $k \times k$,

R_{demux}: Reliability of a demultiplexer,

R_{mux}: Reliability of a multiplexer,

R_l: link reliability and it is assumed to be 1 (i.e. $R_l = 1$)

For parallel banyan, the reliability block diagram for the network is shown in Figure 8.1. As can be seen from the figure, there are only two redundant paths between any source and destination. So, we can use equation (2) above to find the terminal reliability of the parallel banyan network as follows.

$$TR_{Pbanyan} = 1 - (1 - R_Y)(1 - R_Y) \tag{8.3}$$

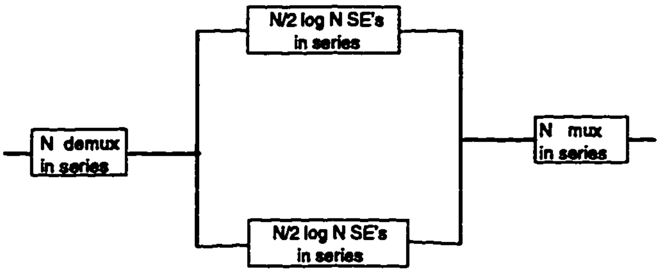


Figure 8.1: Reliability block diagram for parallel banyan network.

Hence,

$$TR_{Pbanyan} = 2R_Y - R_Y^2 \tag{8.4}$$

There are n 2×2 switches connected in series on any path in the parallel banyan network, so

$$R_Y = R_{2\times 2}^n \tag{8.5}$$

From the above equations,

$$TR_{Pbanyan} = 2R_{2\times 2}^n - R_{2\times 2}^{2n} \tag{8.6}$$

Since we have the multiplexers and demultiplexers in series with every path in parallel banyan network, the terminal reliability of the parallel banyan network will be :

$$TR_{Pbanyan} = R_{demux}.[2R_{2\times 2}^n - R_{2\times 2}^{2n}].R_{mux} \tag{8.7}$$

To calculate the terminal reliability of Tagle's network, we see from the reliability block diagram of Figure 8.2 that a cell always has two switching elements to go to at any stage in the network. Hence, we have switching element pairs in series. The

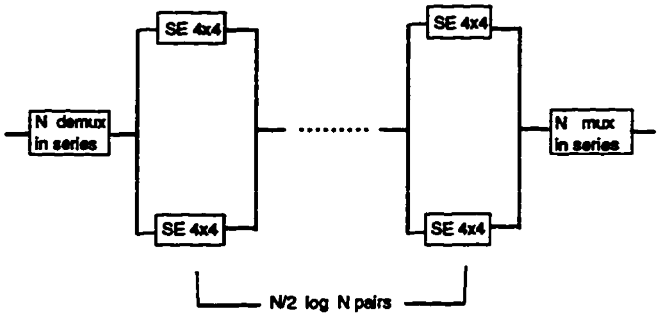


Figure 8.2: Reliability block diagram for Tagle network.

reliability for the switching element pairs is simply the product of the reliability of each pair of 4×4 switching element.

The terminal reliability of Tagle network is therefore as follows:

$$TR_{Tagle} = TR^{n}_{stage} \tag{8.8}$$

where,

$TR_{stage} = 2R_{4\times4} - R^{2}_{4\times4}$

Again, since we have the multiplexer and demultiplexer in series with every path, the terminal reliability of Tagle's network is:

$$TR_{Tagle} = R_{demux}.(2R_{4\times4} - R^{2}_{4\times4})^{n}.R_{mux} \tag{8.9}$$

To compute the terminal reliability of RAZAN, we refer to Figure 8.3. The terminal reliability of RAZAN is found based on the lower bound of the number of lpaths. The lower bound on the number of paths was found in Section 7.1 to be:

$R_{RAZAN} \geq (n+1)!$.

In general, at the first stage, the cell has $n+1$ SEs to choose from. At the second stage, the cell has n SEs to choose from, and so on. Figure 8.3 shows the reliability block diagram of RAZAN network.

The SEs at every stage are parallel and their reliabilities are just the product of the unreliability of each SE of size $k \times k$ where $k = n + 1$, and $n = log_2N$. The overall terminal reliability then is the product of the n stages since the stages are in series. The terminal reliability of RAZAN can be found as follows:

$$TR_{stage0} = 1 - (1 - R_{k\times k})^{n+1} \tag{8.10}$$

$$TR_{stage1} = 1 - (1 - R_{k\times k})^{n} \tag{8.11}$$

In general,

$$TR_{stagei} = 1 - (1 - R_{k\times k})^{n+1-i} \qquad 0 \leq i \leq n-1 \tag{8.12}$$

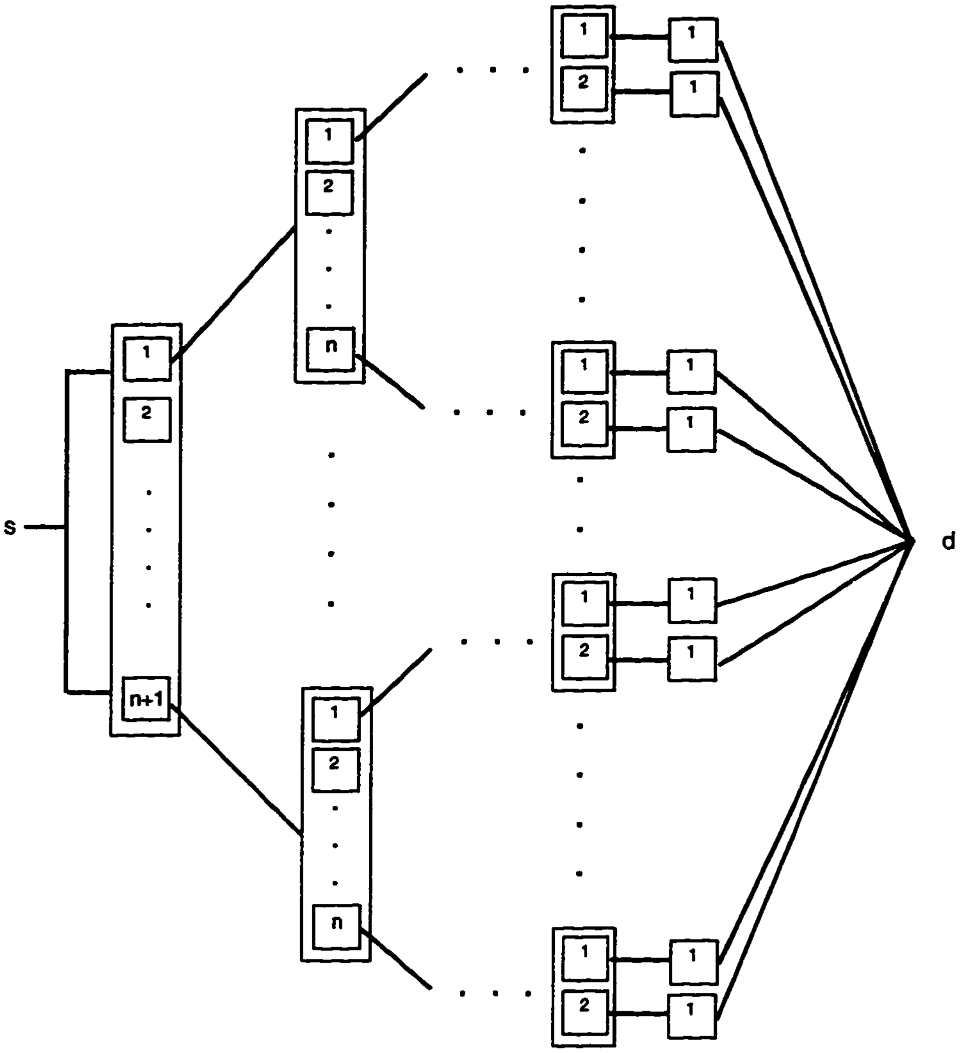
Therefore,

$$TR_{RAZAN} = \textstyle\prod_0^{n-1}[TR_{stagei}]$$

$$TR_{RAZAN} = \textstyle\prod_0^{n-1}[1 - (1 - R_{k\times k})^{n-i+1}]$$

8.2 Complexity Estimation of RAZAN

Switching elements (SEs) in RAZAN have variable sizes depending on the network size. In general, a network of size $n = log_2N$ has switching elements of size $k \times k$ where $k = n + 1$. It is clear that the complexity of SE increases with its size. To



s: Source

d: Destination

Figure 8.3: Reliability block diagram for RAZAN network.

estimate the complexity of the SEs used in RAZAN, we will establish a relationship between the well known 2×2 SE and the SEs used in RAZAN. The relationship can be derived based on the yield equation [55] which is given by:

$$y = \exp^{-\sqrt{Ad}}$$

where A= area and d= no. of defects/cm^2.

To compute the reliabilities of SEs of different sizes, the following assumptions are made.

(1) The reliability follows the same function as the yield equation. That is, $R_{k\times k} = \exp^{-\sqrt{Ad}}$, where k is the switch size.

(2) All the SEs used are crossbar switches.

(3) The area of a crosspoint switch and its interconnection is a.

Then,

$$\begin{aligned} R_{2\times 2} &= \exp^{-\sqrt{4ad}} \\ &= \exp^{-2\sqrt{ad}} \qquad (7) \end{aligned}$$

For a general $k \times k$ switch,

$$\begin{aligned} R_{k\times k} &= \exp^{-\sqrt{k^2ad}} \\ &= \exp^{-k\sqrt{ad}} \qquad (8) \end{aligned}$$

From equations (7) and (8) above, the terminal reliability of SE of size $k \times k$ will be as follows:

$$R_{k\times k} = (R_{2\times 2})^{k/2} \qquad (9)$$

Equation (9) gives the relation between the reliability of a 2×2 SE and any SE

Table 8.1: Reliability of SEs of different sizes.

SE Size (k)	Reliability $R_{k \times k}$
2	0.96
3	0.94
4	0.922
5	0.903
6	0.885
7	0.867
8	0.849
9	0.832
10	0.815
11	0.799
12	0.783
13	0.767
14	0.752
15	0.736
16	0.721
17	0.707
18	0.693
19	0.679
20	0.665
21	0.651

of size $k \times k$. If we assume that the reliability of a 2×2 SE is 0.96, the reliability of any SE of size $k \times k$ is shown in Table 8.1. We notice from Table 8.1 that the reliability of the SE decreases as the size of SE increases.

8.3 Numerical Results

Using the results from Table 8.1, the terminal reliabilities of parallel banyan, Tagle, and RAZAN networks for different sizes are shown in Table 8.2.

A plot of the terminal reliability of the three networks versus network size is

Table 8.2: Comparison of Terminal Reliabilities

Network Size (n)	Parallel Banyan	Tagle	RAZAN(lower bound)
3	0.9470	0.9430	0.9934
4	0.9380	0.9370	0.9896
5	0.9270	0.9310	0.9851
6	0.9150	0.9250	0.9796
7	0.9000	0.9200	0.9732
8	0.8850	0.9140	0.9662
9	0.8690	0.9090	0.9583
10	0.8520	0.9030	0.9497
11	0.8340	0.8980	0.9405
12	0.8160	0.8920	0.9308
13	0.7970	0.8870	0.9195
14	0.7780	0.8810	0.9072
15	0.7590	0.8760	0.8946
16	0.7390	0.8710	0.8819
17	0.7200	0.8650	0.8678
18	0.7000	0.8600	0.8538
19	0.6800	0.8550	0.8383
20	0.6600	0.8500	0.8218

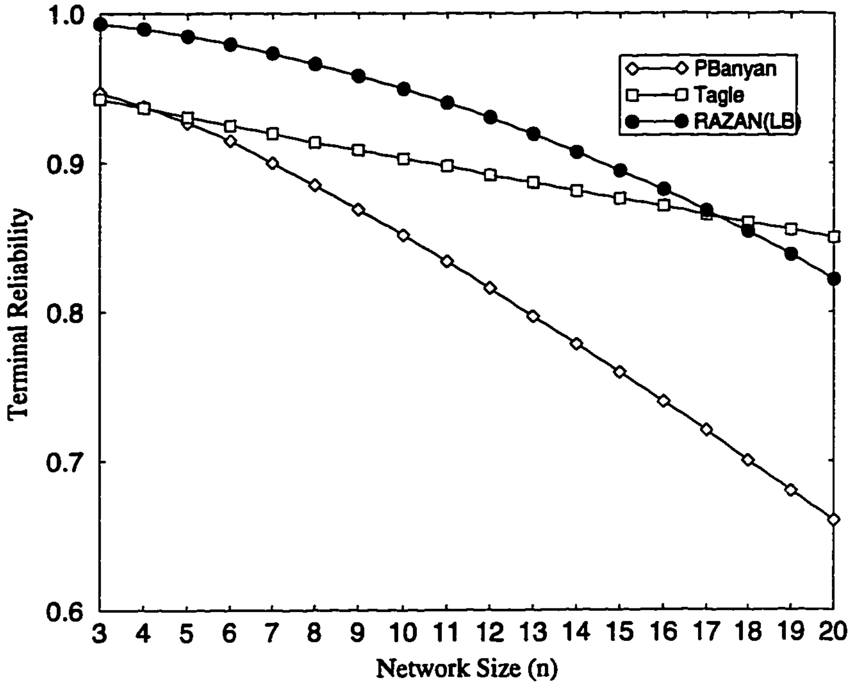


Figure 8.4: Terminal reliability vs network size.

provided in Figure 8.4. As can be seen from the graph and the table, the lower bound terminal reliability of RAZAN is higher than the the exact terminal reliability of parallel banyan network for any network size. Compared to the exact terminal reliability of Tagle network, the lower bound terminal reliability of RAZAN is better for small and medium network sizes. For network sizes 2^{18} and above, the lower bound terminal reliability of RAZAN is less than the exact terminal reliability of Tagle network.

8.4 Concluding Remarks

In this chapter, we analyzed the terminal reliability of parallel banyan, Tagle, and RAZAN networks. Terminal reliability formulae were derived for each network. Complexity estimation of SEs used in RAZAN was provided based on the yield equation of switches. Numerical results of the lower bound RAZAN terminal reliability were presented in comparison with the exact terminal reliabilities of parallel banyan and Tagle. RAZAN exhibited clear superior performance over parallel banyan network and small and meduim sizes of Tagle network. For very large network sizes, the lower bound terminal reliability of RAZAN is less than the exact terminal reliability of Tagle network.

Chapter 9

Performance of RAZAN in the Presence of Faults

In this chapter, we explain the behavior of RAZAN network under faulty switching elements. We illustrate by an example how the routing algorithm works if faulty switching elements are encountered. At the end of the chapter, we show the performance of RAZAN network under fault conditions. The results are compared with those of Benes, parallel banyan, and Tagle's networks.

9.1 Behavior of RAZAN under Faulty Switching Elements

Previously, we have explained the simple routing algorithm of RAZAN. An example of routing a cell from a source to a destination was provided under the assumption that no faulty SEs exist in the network. Here, we briefly summarize the routing algorithm and consider the example given before assuming some faulty SEs.

According to the routing algorithm, first, the binary representation of the source and the destination are exclusive-ored and the result is stored in V. The position in V which has the first occurrence of 1 (starting from the left) is used to flip the bit in the source having the same position. This gives the SE number to which the cell should be forwarded. If this SE is faulty or if the link between the two SEs is busy, faulty, or congested, then the position of the next occurrence of 1 is used and the procedure is repeated again.

This process is repeated until all 1s in V are exhausted. If so, the straight link between the source and the destination is used to forward the cell to the next SE. Again, if the next SE is faulty or the link between the two SEs is busy, faulty, or congested, the procedure is repeated to find a link by testing the occurrences of 0 starting from the first occurrence. If no path is found after exhausting all the occurrences of 0, the cell is considered lost since no more links can be used.

Each SE has a maximum of $n+1$ links which connect the SE to $n+1$ SEs in the next stage. So, the probability of cell loss is expected to be low even in the presence of faults. Hence, the expected throughput will be high even under some faulty SEs in the network.

9.2 Example

Previously, we showed an example of routing a cell from source 0 to destination 6 in an 8×8 RAZAN under no faults. In this section, we assume that some SEs on the path between 0 and 6 are faulty. Figure 9.1 shows an 8×8 network with faulty SEs. The algorithm works as follows refering to Figure 9.1 and Table 9.1. Since S=000 and D= 110, the xor of the two is V= 110. The first occurrence of 1 in V is at position 1 (counting from the leftmost bit). So, bit 1 in S is flipped which should give the number of the SE (D1=100, i.e. SE 4) in the next stage for the cell to go to. But Figure 9.1 and shows that this SE is faulty and can not be used. The algorithm then looks to the next occurrence of 1 in V which is bit 2 in this case. Bit 2 in S is flipped to give D 1=010 (i.e. SE number 2). Since this SE is not faulty, the cell will be forwarded to it.

In the first stage, SE 2 has the cell which has destination 6. This give S=010 and D=110. The xor of S and D gives V=100. Bit 1 has the first occurrence of 1 and its corresponding position in S is flipped to give D1 =110. Unfortunately, SE 6 is faulty. Because V does not have more 1s, the SE which has a straight connection

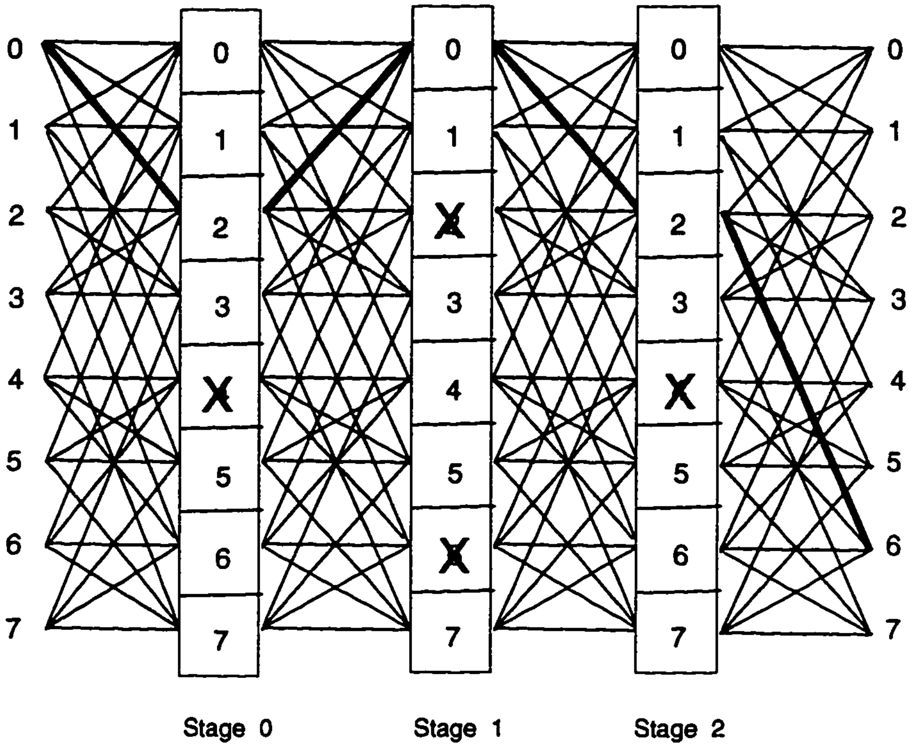


Figure 9.1: Routing under faulty SEs.

Table 9.1: Routing example under faulty SEs

S	D	V (SxorD)	Stage	SE No.	SE status	Action
000	110	110	0	100	faulty	Try next
000	110	110	0	010	available	Forward
010	110	100	1	110	faulty	Try next
010	110	100	1	010	faulty	Try next
010	110	100	1	000	available	Forward
000	110	110	2	100	faulty	Try next
000	110	110	2	010	available	Forward
010	100	110	-	110	-	Forward

with SE 2 is assumed. So, SE 2 in the next stage is the SE which the cell should go to. Again, SE 2 is faulty and can not be used. In this case, the routing algorithm finds the first occurrence of 0 in V which is bit 2. So, bit 2 in S is flipped to give D1 = 000. Fortunately, SE 0 in the next stage is not faulty. So, the cell will be sent to SE 0 in stage 2. At this stage, the values of S and D are 000 and 110 respectively which gives an xor value of 110.

The routing algorithm will use the value of V to give the SE number that the cell should be forwarded to as D1= 100. This is because bit 1 in V has the first occurrence of 1. Figure 9.1 shows that SE 4 in stage 3 is faulty and hence can not be used. The algorithm in this case decides to forward the cell to D1= 010 since bit 2 in V has the next occurrence of 1. The cell is forwarded to SE 2 in stage 3 because the SE is available.

The routing algorithm at SE 2 in stage 3 has S = 010 and D 110. The xor value of S and D is V = 100. The algorithm flips bit 1 in S to give the number of the output port the cell should go to. Finally, D1 which is equal 110 is the required destination of the cell. So, the cell reaches its destination successfully even though some faulty SEs have been encountered. The path the cell should go to from source S = 0 to D = 6 is shown in Figure 9.1.

9.3 Performance of RAZAN under Fault Conditions

As stated previously, RAZAN of size $n \times n$ has $n + 1$ disjoint paths for each source-destination pair. This makes the network n-fault tolerant, i.e.n faults can be tolerated in any of the n stages of RAZAN. To evaluate the performance of RAZAN under fault conditions, a computer simulation program has been developed. We are assuming the following fault model for RAZAN network:

1- Only SEs can fail.

2- A faulty SE is treated as unusable and no cells can be routed through it.

3- SE faults occur completely randomly and independently with equal probability.

Faults in interconnecting links can be accommodated in this model by treating them as part of the SEs. The result of simulating RAZAN under faulty conditions is compared to that of Benes, parallel banyan, and Tagle.

For Benes network, it is assumed that no faults can occur in the first and last stages. The same is assumed for parallel banyan and Tagle networks wherein the first and last stages are the demultiplexers and multiplexers, respectively. The same aforementioned fault model is assumed for the three networks. The networks are subjected to a varying number of faulty SEs ranging from 1 to as many as 20 faulty switches. For the three networks (Benes, parallel banyan, and Tagle), cells which are unable to win the link because of contention are considered lost.

For RAZAN network, cells will be lost only if they can not be forwarded to any SE in the next stage through all $n + 1$ links. Simulation is done for network sizes n=6 and 8. Figure 9.2 shows the throughput of RAZAN in presence of increasing number of faults in a network of size $n = 6$. The figure shows that RAZAN has stable high throughput even if the number of faults increases up to 50 faulty SEs in the network. The throughput of RAZAN drops after that because the increase in the number of faulty SEs will eliminate some paths between some source-destination pairs. Interestingly, for n=6, RAZAN has a throughput of approximately 50% even though it has 100 faulty SEs which corresponds to 52% of the total number of SEs. This is because of the availability of disjoint paths and the huge number of redundant paths between any source-destination pair.

Figure 9.3 shows the throughput of RAZAN in the presence of increasing number of faults in a network of size $n = 8$. We see that RAZAN has stable high throughput even if the number of faults increases up to 250 faulty SEs (approximately 25% of SEs are faulty) in the network which is much higher than 50 SEs in the case of n = 6. This is because the larger the network size is, the more disjoint paths and redundant paths there are. So, faulty SEs can be avoided with no deterioration in the throughput. The throughput gradually decreases after 250 faulty SEs in the network with increasing number of faulty SEs. Figure 9.3 shows that the throughput of RAZAN is more than 62% even with 325 faulty SEs in the network.

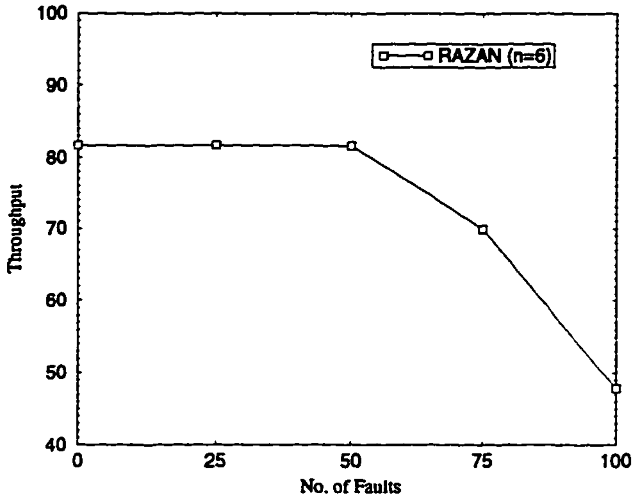


Figure 9.2: Throughput of RAZAN vs No. of faults for network size n=6

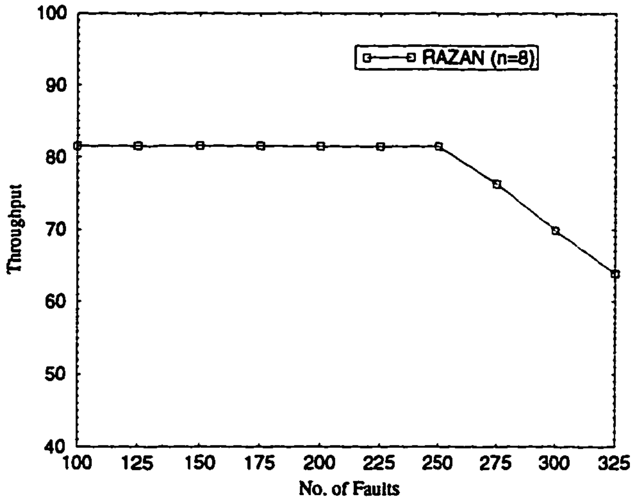


Figure 9.3: Throughput of RAZAN vs No. of faults for network size n=8

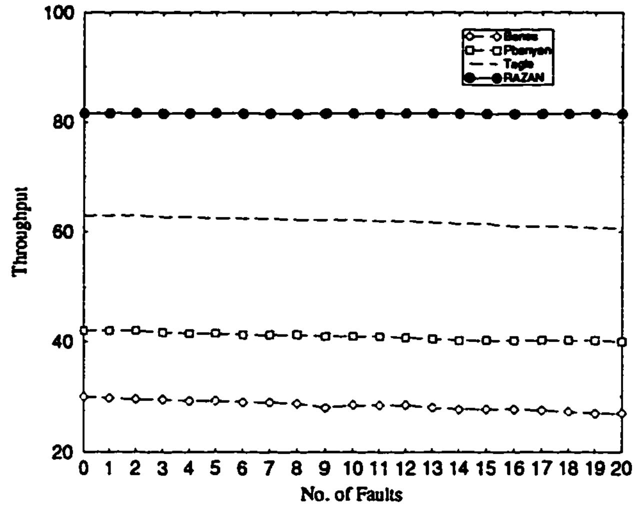


Figure 9.4: Throughput for all Networks vs No. of faults for (n=8)

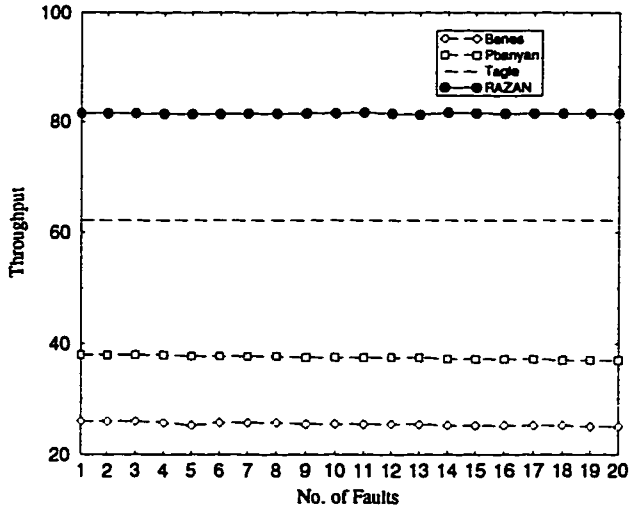


Figure 9.5: Throughput for all Networks vs No. of faults for (n=10)

The graphs shown in Figure 9.4 and Figure 9.5 show the superior performance of RAZAN over the other three networks even in the presence of large number of faults in the network for different network sizes. The throughput of Benes network deteriorates as the number of faults increases for any network size. This is because Benes network offers fault tolerance till the middle stage, after which a cell should follow a single path to its destination. Parallel banyan shows an improvement over Benes network but its throughput does not exceed 48% in the absence of faults. The throughput of parallel banyan also decreases as the number of faults increases. This can be justified by the fact that fault tolerance in parallel banyan is provided only at the first stage, i.e. the demultiplexers, after which the network is the ordinary single path banyan network.

Tagle's network shows a notable improvement in the throughput over both Benes and parallel banyan. This is because fault tolerance is provided at each stage of the network. There are always two paths at each stage of the network for the cell to choose from. The susceptibility of Tagle's network to faults is higher for small network sizes. So the throughput decreases smoothly for large network sizes. However, the throughput of Tagle's network does not exceed 65% under no faults.

The thoughput of RAZAN outperformes those obtained by the other three networks of interest. The throughput achieved is up to 82% if no faults are encountered in the system. More interestingly, the throughput of RAZAN network stays at about 81% even in the presence of up to 20 faults in the system. The susceptibility of

RAZAN network to faults is very low for networks of different sizes. This can be accounted for by the very high number of redundant and disjoint paths which increase as the network size increases. This implies that the effect of faults decreases.

High throughput is an important requirement for B-ISDN. Most networks have a decrease in throughput with an increase in the number of faults. Since B-ISDN is assumed to be used for very huge number of connections and under heavy loads, fault tolerance is essential. Throughput, of course, should be kept high even in the presence of faults. RAZAN network satisfies all the above requirements where it gives very high throughput compared to others even in the presence of a high number of faulty SEs.

9.4 Concluding Remarks

In this chapter, we explained the behavior of RAZAN network under faulty SEs. The routing algorithm was illustrated with an example to show how RAZAN accomodates faulty SEs. The throughput of RAZAN under faulty conditions was computed to show its superiority over other networks. RAZAN has exhibited a high throughput even under high number of faulty SEs compared to Benes, parallel banyan, and Tagle networks.

Chapter 10

Conclusions and Future Work

In this chapter, we summarize the results obtained in the thesis and the contributions to fault tolerance in B-ISDN networks. Future research based on this thesis in related areas is also described.

10.1 Summary and Conclusions

In this thesis, we presented a high performance fault tolerant switch for B-ISDN, called the Reliable and Zealous Network (RAZAN). It consists of $NLog_2N$ switching elements of size $(n+1) \times (n+1)$, where N is the number of inputs and $n = Log_2N$ is the number of stages. The performance of the network was evaluated analytically as well as through simulation under uniform traffic load. RAZAN shows excellent performance compared to other networks with respect to throughput in the fault-free network and in the presence of faults in the network. Moreover, RAZAN has a high terminal reliability even for large network sizes. The architecture of RAZAN also offers high fault tolerance, in sequence delivery of cells, simple routing, and regularity.

10.2 Contributions

In this section, we present the main contributions of this thesis in the area of fault tolerant B-ISDN networks.

- In this thesis, we presented a survey on fault tolerant B-ISDN networks. Then, we compared these networks together in terms of some aspects like, modularity, routing techniques, degree of fault tolerance, and complexity.
- We studied the ILN network [41] and we reduced the number of stages by one to come up with our network RAZAN.
- We proposed a new self-routing algorithm for RAZAN which uses only the binary bits of the source and destination to decide about the link that the cell should use.
- The thesis presented an analysis of the cost of RAZAN in terms of SEs and links used.

- In this research, we derived an analytical model for the performance of RAZAN. Then, a simulation program was developed for the network to explore the behavior of the network under uniform traffic. RAZAN has shown a high throughput performance of about 82% for different network sizes.

- We studied the fault tolerance aspects of RAZAN in terms of the number of disjoint and redundant paths available between any source-destination pair. RAZAN has $n + 1$ disjoint paths available between any source-destination pair. This makes RAZAN an n-fault tolerant network. The number of disjoint paths increases as the network size increases. The number of redundant paths of RAZAN was also analyzed. It has been shown that RAZAN has increasing number of redundant paths with increasing sizes.

- We then analyzed the terminal reliability of RAZAN which shows a very high reliability (approximately 1) for different network sizes.

- In the thesis, we studied the behavior of RAZAN under faulty SEs. RAZAN has shown a stable high throughput even with increasing number of faults.

- We compared all the results above to some other networks, namely, parallel banyan, Benes, and Tagle networks [48]. The superiority of RAZAN over these networks of interest was evident.

10.3 Future Work

In this section, we outline some future research topics in the area of fault tolerant B-ISDN networks. We demonstrated through our work the feasibility and advantages of RAZAN for B-ISDN networks. The future scope of our work involves the following.

- The SEs of RAZAN increases as the network size increases. So, for each network size, RAZAN's SE has different dilation factor for the number of inputs and outputs. We will try to fix the dilation factor of the SEs of RAZAN, i.e. to use SEs of fixed size in any network, and study the effect of that on the network.
- In the thesis, we have seen that the routing algorithm used in RAZAN tries to deliver the cell as fast as possible to the SE which has the same number as the destination. Another point for future research is to reduce the number of stages in RAZAN by one and investigate the behavior of RAZAN under this situation.
- Since RAZAN is mainly proposed for B-ISDN, we will try to study the behavior of the network under different types of workload distributions.

Bibliography

[1] Rainer Handel and Manfred N. Huber. *Integrated Broadband Networks - An Introduction to ATM-Based Networks.* Addison-Wesley, 1991.

[2] Ronald J. Vetter. ATM Concepts, Architectures and Protocols. *Communications of the ACM*, 38(10):30–38, February 1995.

[3] E.D. Sykas, K.M. Vlakos, and M.J. Hillyard. Overview of ATM Networks: Functions and Procedures. *Computer Communications*, 14(10):615–625, Dec. 1991.

[4] Fouad A. Tobagi. Fast Packet Switch Architectures For Broadband Integrated Services Digital Networks. In *Proceedings of the IEEE*, volume 78, pages 133–167, Jan. 1990.

[5] Ra'ed Y. Awedh and H. T. Moftah. Survey of ATM Switch Architectures. *Computer Networks and ISDN Systems*, (27):1567–1613, 1995.

[6] M. Devault, J. Cohennec, and M. Servel. The Prelude Experiment: Assessments and Future Prospects. *IEEE J. Selected Areas in Communications*, 6(9):1528–1537, Dec. 1988.

[7] H. Kuwahara, N. Endo, M. Ogino, and T. Kozaki. Shared Buffer Memory Switch For an ATM Exchange. In *Int. Conf. on Communications*, pages 4.4.1–4.4.5, June 1989.

[8] A. Jajszczyk and Kabacinski. A Growable Shared-Buffer-Based ATM Switching Fabric. In *Proceedings of the IEEE Global Telecommunications Conference*, pages 29–33, 1993.

[9] Kuei Y. Kou, Akira Arutaki, and Susumu Iwasaki. The Architecture and Implementation of ATM Switch for Broadband ISDN. In *Proceedings of the IEEE Global Telecommunications Conference*, pages 9–13, 1993.

[10] H. Suzuki et al. Output-Buffer Switch Architecture For Asynchronous Transfer Mode. In *Int. Conf. on Communications*, pages 145–149, June 1989.

[11] I. Sidi I. Cidon, I. Gopal and W. Liu. Real-Time Packet Switching: A Performance Analysis. *IEEE J. Selected Areas in Communications*, 6(9):1576–1586, Dec. 1988.

[12] M. Hluchyj and M. Karol. Queuing in High-Performance Packet Switching. *IEEE J. Selected Areas in Communications*, 6(9):1587–1597, Dec. 1988.

[13] A. Haung and S. Knauer. Starlite: A Wideband Digital Switch. In *GLOBECOM'84*, pages 121–125, Dec. 1984.

[14] J. Giacopelli, M. Littlewood, and W.D. Sincoslie. Sunshine: A High Performance Self-Routing Broadband Packet Switch Architecture. In *IEEE Integrated Solid-State Circuit Conference*, pages 599–606, 1990.

[15] T. T. Lee. A Modular Architecture for Very Large Packet Switches. *IEEE Transactions on Communications*, 6(9):1455–1467, July 1990.

[16] M. Kumar and J.R. Jump. Performance of Unbuffered Shuffle-Exchange Networks. *IEEE Transactions on Computers*, C-35(6):573–577, June 1986.

[17] F. M. Chiussi F. A. Tobagi, T. Kowk. Architecture, Performance, and Implementation of the Tandem Banyan Fast Packet Switch. *IEEE J. Selected Areas in Communications*, 9(8):1173–1193, Oct. 1991.

[18] T. T. Lee and Soung C. Liew. Broadband Packet Switches Based on Dialated Interconnection Networks. *IEEE Transactions on Communications*, 42(2/3/4):732–744, Feb./March/April 1994.

[19] Sandro Bassi, Maurizio Decina, Paolo Giacomazzi, and Achille Pattavina. Multistage Shuffle Networks with Shortest Path and Deflection Routing for High-Performance ATM Switching: The Open-Loop Shuffleout. *IEEE Transactions on Communications*, 42(10):3034–3044, October 1994.

[20] Maurizio Decina, Paolo Giacomazzi, and Achille Pattavina. Multistage Shuffle Networks with Shortest Path and Deflection Routing for High-Performance ATM Switching: The Closed-Loop Shuffleout. *IEEE Transactions on Communications*, 42(11):3034–3044, November 1994.

[21] Y.-S. Yeh, M. Hluchyj, and A. Acampro. The Knockout Switch: A Simple Modular Architecture for High Performance Packet Switching. *IEEE J. Selected Areas in Communications*, 5(8):1274–1283, Oct. 1987.

[22] H. Ahmadi, E. Ruffner, and S. Roy. A High Performance Switch Fabric For Integrated Circuit and Packet Switching. In *INFOCOM'88*, pages 9–18, Mar. 1988.

[23] G. B. Adams, D. P. Agrawal, and H.J. Siegel. A Survey and Comparison of Fault-Tolerant Multistage Interconnection Networks. *IEEE Computer*, pages 14–27, June 1987.

[24] Hiroyuki Fujii and Noriaki Yoshikai. Restoration Message Transfer Mechanism and Restoration Characteristics of Double-Search Self-Healing ATM Network. *IEEE J. Selected Areas in Communications*, 12(1):149–157, January 1994.

[25] M.A. Henrion, G.J. Eilenberger, G.H. Petit, and P.H. Parmentier. A Multipath Self-Routing Switch. *IEEE Communications Magazine*, pages 64–52, April 1993.

[26] H. Sakauchi, Y. Nishimura, and S. Hasegawa. A Self-Healing Network with Economical Space-Channel Assignment. In *GLOBECOM'90*, pages 438–443, 1990.

[27] W.D. Grover. The Selfhealing Trade Network: A Fast Distributed Restoration Technique for Networks Using Digital Cross-Connect Machines. In *GLOBECOM'87*, pages 1090–1095, 1987.

[28] R. Kawamura, K. Sato, and I. Tokizawa. The Self-healing ATM Network Technique Utilizing Virtual Paths. In *Networks'92*, pages 129–134, 1992.

[29] J. T. Blake and Kishor S. Trivedi. Reliability Analysis of Interconnection Networks Using Hierarchical Composition. *IEEE Transactions on Reliability*, 38(1):111–119, April 1989.

[30] Varma and C. S. Raghavendra. Reliability Analysis of Redundant Path Interconnection Networks. *IEEE Transactions on Reliability*, 38(1):130–137, April 1989.

[31] C. Booting, S. Rai, and D. P. Agrawal. Reliability Computation of Multistage Interconnection Networks. *IEEE Transactions on Reliability*, 38(1):138–145, April 1989.

[32] N. K. Sharma. Fault-Tolerance of a MIN Using Hybrid Redundancy. In *Proc. IEEE 27th Annual Simulation Symposium*, pages 142–149, 1994.

[33] V. P. Kumar. Augmented Shuffle Exchange Multistage Interconnection Networks. *Computer*, pages 30–40, June 1987.

[34] M. Belkadi and H. T. Mouftah. On a Class of High Performance Highly Reliable Switching Network for B-ISDN. In *Proc. IEEE Global Telecommunications Conference*, pages 104–109, 1995.

[35] J. H. Patel. Performance of Processor Memory Interconnections for Multiprocessors. *IEEE Transactions on Computers*, 30(10):771–780, Oct. 1981.

[36] C. L. Wu and Feng T.-Y. On a Class of Multistage Interconnections Networks. *IEEE Transactions on Computers*, 29(8):496–502, Aug. 1980.

[37] G. B. Adams III and H. J. Siegel. The Extra Stage Cube: A Fault-Tolerant Interconnection Network for Supersystems. *IEEE Transactions on Computers*, 31(5):443–454, May 1982.

[38] J. T. Blake and Kishor S. Trivedi. Multistage Interconnection Network Reliability. *IEEE Transactions on Computers*, 38(11):1600–1604, Nov. 1989.

[39] S. Konstantinidou. The Selective Extra-Stage Butterfly. *IEEE Transactions on VLSI Systems*, 1(2):167–171, June 1993.

[40] T. M. Bachtiar and M. H. Abd-El-Barr. Logical Neighborhood Network for Fault Tolerant in Packet Switching Networks. In *Proceedings of the 5th International Conference on Microelectronics*, pages 287–293, Dec. 1993.

[41] Mostafa Abd-El-Barr and Osama Abed. Fault-Tolerance and Terminal Reliability for a Class of Data Manipulator Networks. In *Proc. 37th Midwest Symposuim on Circuits and Systems*, pages 225–229, 1995.

[42] Mostafa Abd-El-Barr, Khalid Al-Tawil, and Osama Abed. Fault Tolerance and Reliability Analysis of Multi-stage Data Manipulator Networks. In *8th ISCA/IEEE International Conference on Parallel and Distributed Computing Systems*, pages 275–280, Sept. 1995.

[43] Robert McMillen and Howard Siegel. Performance and Fault Tolerance Improvements in the Augmented Data Manipulator Network. In *The 9th Annual Symposium on Computer Architecture*, pages 63–72, 1982.

[44] Menkae Jeng and Howard Jay Siegel. Design and Analysis of Dynamic Redundancy Networks. *IEEE Transactions on Computers*, 37(9):1019–1029, Sept. 1991.

[45] Arata Itoh. A Fault-Tolerant Switching Network for B-ISDN. *IEEE J. Selected Areas in Communications*, 9(8):1218–1226, Oct. 1991.

[46] Chi-Chun Lo and Chen-Yu Chiu. A Fault-Tolerant Architecture for ATM Networks. In *IEEE 14th Annual International Phoenix Conference on Computers and Communications*, pages 29–36, 1995.

[47] J.-J. Li and C. M. Weng. B-tree: A High-Performance Fault-Tolerant ATM Switch. In *IEE Proc. Commun.*, volume 141, pages 20–28, part II, 1994.

[48] Pierre U. Tagle and Neeraj K. Sharma. A High-Performance Fault-Tolerant Switching Network for B-ISDN. In *IEEE 14th Annual International Phoenix Conference on Computers and Communications*, pages 599–606, March 1995.

[49] Krishnan Padmanabhan. FAUST: A Fault Tolerant Sparing Technique for ATM Switch Architectures. In *Proc. IEEE Global Telecommunications Conference*, pages 1368–1374, 1993.

[50] Krishnan Padmanabhan. An Efficient Architecture for Fault-Tolerant ATM Switches. *IEEE Transactions on Networking*, 3(5):527–537, Oct. 1995.

[51] Wen-Shyen E. Chen, Young Man Kim, Yow-Wei Yao, and Ming T. Liu. FDB: A High-Performance Fault-Tolerant Switching Fabric for ATM Switching Systems. In *Proc. 10th Annual International Phoenix Conf. on Comp. and Comm.*, pages 703–709, 1991.

[52] Barry W. Johnson. *Design and Analysis of Fault Tolerant Digital Systems.* Addison-Wesley, 1989.

[53] V.P. Kumar, J.G. Kneuer, D. Pal, and B. Brunner. PHOENIX: A Building Block for Fault Tolerant Broadband Packet Switches. In *GLOBECOM'91*, pages 228–233, 1991.

[54] N. Tzeng, P. Yew, and C. Zhu. A Fault-Tolerant Scheme for Multistage Interconnection Networks. In *12th Intl. Symp. on Computer Architecture*, pages 368–375, 1985.

[55] Neil Weste and Karman Eshraghian. *Principles of CMOS VLSI Design*. Addison-Wesley, 1993.

Vitae

- Talha M. Al-Jarad
- Born in Deer-Ezoor, Syria in 1970
- Received Bachelor of Science (**B.S.**) degree in Computer Engineering from King Fahd University of Petroleum and Minerals (**KFUPM**), Dhahran, Saudi Arabia in 1992.
- Received Master of Science (**M.S.**) degree in Computer Engineering from KFUPM, Dhahran, Saudi Arabia in 1996.