On Performance Analysis of Fault-Tolerant Multi-Stage Interconnection Networks

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

Master of Engineering
in
Computer Science and Engineering

Thapar University, Patiala

By:
Karamjit Kaur Cheema
(80632007)

Under the supervision of:
Ms. Rinkle Aggarwal
Lecturer (SS)

MAY 2008

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

CHAPTER 1

Introduction to MINs

This chapter gives an introduction to MINs. Section 1 gives definition of MINs, whereas Section 2 explains the classification of MINs according to different parameters. Section 3 briefs about different types of connections in MINs. Section 4 gives an idea about routing in MINs, definition of routing tag and different types of routing possible in MINs. Section 5 explains a very important performance measure of MINs i.e. fault-tolerance. Section 6 gives an overview of how the thesis is organized.

1.1 Multi-stage interconnection networks

Multi-stage interconnection networks (MINs) consist of more than one stages of small interconnection elements called switching elements [5] and links interconnecting them [26]. Multi-stage interconnection networks are described by three characteristic features, the switching elements, network topology [18, 22] and control structures [23]. A multi-stage interconnection network is actually a compromise between crossbar [21] and shared bus networks, indicated in the following table; describing the properties of various types of multiprocessor interconnections.

Table 1.1 Comparison of bus, crossbar and MINs

Property	**Bus**	**Crossbar**	**Multi-stage**
Speed	Low	High	High
Cost	Low	High	Moderate
Reliability	Low	High	High
Configurability	High	Low	Moderate
Complexity	Low	High	Moderate

Multi-stage interconnection networks:

Attempt to reduce cost.

Attempt to decrease diameter, diameter is the longest path between any two nodes.

In a multi-stage interconnection network, as in a crossbar, switching elements are distinct from processors. However, fewer than O (P^2) switches are used to connect P processors. Instead messages pass through a series of switch stages.

Switching Elements: The switching element [5] may be viewed as a very small network. These switches are the devices having multiple inputs and multiple outputs. The number of inputs/outputs and the input-to-output connections supported within a switch can assume either the straight or the exchange states [9]. A four function switch box can be in any one of the following four states i.e. straight, exchange, upper broadcast and lower broadcast.

1.2 Classification of Multi-stage interconnection networks

Multi-stage interconnection networks can be classified according to different categories. The main classification categories are path, switches and control.

1.2.1 Classification according to path

MINs can be classified according to path as unique and multi-path networks, as described below:

1.2.1.1 *Unique path networks:* These networks provide a unique path between every source and destination [22]. The failure of any switching element along the path disconnects some source-destination pairs, so adversely affecting the capabilities of existing network. These networks are not reliable for a large multiprocessor system, as they cannot tolerate even a single fault. In case of multiple requests, a source destination connection may be blocked by a previously established connection, thus providing a poor performance [10].

1.2.1.2 *Multi path networks:* These networks provide more than one paths between a given source and a destination [22]. In case, there is a failure of one switching element [5] in the path, the request is routed through some alternative path. Unique path multi-stage interconnection networks can be made multi path by adding redundancy in the form of extra switching elements, links, stages, sub networks, by increasing the size of switching elements or using multiple networks [10].

Multi path multi-stage interconnection networks can be either static or dynamic [28]. For static networks, if a fault is encountered, then data has to backtrack, to the source or some fixed point to select an alternative path in the network. The implementation of backtracking is expensive in terms of the hardware. In dynamic networks, if a fault is encountered in a particular stage, a switching element in preceding stage will re-route data through an alternative available path.

1.2.2 Classification according to switches

MINs can be classified according to switches as regular and irregular networks, as described below:

1.2.2.1 Regular networks: Regular multi-stage interconnection networks have an equal number of switching elements per stage; as a result they may impose equal time delay to all the requests passing through them [6].

1.2.2.2 Irregular networks: Irregular multi-stage interconnection networks have unequal number of switching elements as each stage and thus they are inherently multi path in nature. For a given source destination pair, different path lengths are available [6].

1.2.3 Classification according to control

MINs can be classified according to control as flip controlled and distributed control networks, as described below:

1.2.3.1 Flip controlled networks: Flip controlled multi-stage interconnection networks have a common control signal for switching in various switching elements at a given stage. These networks are less complicated due to lesser number of control signals but have lesser bandwidth [10].

1.2.3.2 Distributed control networks: Distributed control multi-stage interconnection networks have a separate control signal for every switching element [23]. These have higher bandwidth due to selection of source destination pair at a given time and are quite complex.

1.2.4 Other classification:

1.2.4.1 Blocking networks: In blocking network, simultaneous connections of more than one terminal may result in conflict in use of network communication links [21]. Example of blocking network is Omega network.

1.2.4.2 Non blocking networks: A network is called non-blocking if it is possible to route data from any source to any destination, in presence of other established source-destination routes, provided no two sources have same destination [21]. In other words, a network that can handle all possible connections without blocking is called non-blocking network.

1.3 Types of connections in MINs

There are four types of connections which are commonly used in multi-stage interconnection networks. These are:

One to one connection: a one to one connection passes information from a source to a destination. The exact route taken by the information is determined by the path itself [17].

Multi path connection: Multi path means many one to one connections are active simultaneously.

Permutation connection: A set of one to one connections such that no two connections have the same source or destination. Such connections are meaningful only in cases of equal number of sources and destinations [7].

Broadcast connection: Information flows from source to various destinations either some or all. Thus a number of destinations simultaneously receive the information.

1.4 Routing in MINs

No decision regarding the routing in a network is perfect one. To acquire nearly complete knowledge for routing would require so much overhead that traffic throughout would be simultaneously reduced. For example, if there is minimal traffic,

the network path with minimum number of links will normally be the best. If a node or switching element fails, then the path with minimum number of links that bypasses the failures will be the best [25]. As traffic builds up, however this simple routing strategy can give poor result at times because the shortest path may happen to be congested. So the network as a whole should employ a routing strategy that would bypass areas of congestion.

There are several objectives of routing strategy [2]:

Minimize the transmit time

Minimize the costs

Maximize the network throughput capability.

To minimize the transmit times under conditions of changing load, many control signals or overheads would be sent so that network throughput [2] would be reduced. On the other hand, maximizing the throughput could be done at the expense of packet transmit times.

1.4.1 Routing tags

Routing tag is a way of describing the path through the network [22]. For multi-stage interconnection networks, these tags are generally expressed as a 4-bit binary of the destination. Each successive bit, encodes the settings for the switch in the next stage along a desired path. This control is called distributed if the devices, using the network switches can be set on their own based on the tag information [23].

1.4.2 Types of routing in MINs

The routing in multi-stage interconnection networks take place through the generation of routing tags, which specify a fault free path [22]. There are basically three types of routing that is commonly used in multi-stage interconnection networks.

1.4.2.1 Non adaptive routing: In this method a source leans about a fault only when the path it is attempting to establish, reach the faulty network component. A notice of fault is sent to the source. Then it tries next alternative available path. This method has poor performance though it requires little hardware [11].

1.4.2.2 Adaptive routing: The adaptive routing can be of following types [6]:

Notification on demand: with notification on demand, a source maintains a table of faults it encountered in attempting to establish paths and uses this information to guide the future routing.

Broadcast routing: With broadcast notification of a fault, all the sources are notified of the fault components as they are diagnosed [11].

1.4.2.3 Dynamic routing: A fault free path need not to be specified by a source if the routing tags are modified in response to the faults as a path is followed or established [6]. The dynamic routing can be accomplished in multi-stage interconnection networks constructed of switches, which are capable of performing the necessary tag revision.

1.5 Fault tolerance

A fault tolerant multi-stage interconnection network provides service even under the faults [28]. Fault can be permanent or transient in nature. Fault tolerance is a criterion that must be met for the network which has tolerated a given fault or faults [6].
A network is a single fault tolerant if it can function as specified by its fault tolerance criteria despite any single fault conforming to its fault models [9]. In general, if any set of i-faults can be tolerated by a network, then network is said to be i-fault tolerant. A network that can tolerate some instances of i-faults is robust although not i-fault tolerant [17].

1.6 Organization of thesis

A survey of existing regular and irregular multi-stage interconnection networks is covered in chapter 2. Chapter 3 deals with problem definition and Chapter 4 covers the construction and routing of one existing MIN (ABN) and two proposed MINS (MABN and IABN). Chapter 5 explains the performance analysis of three networks (ABN, MABN and IABN). Chapter 6 shows the experimental results along with graphs. The last Chapter is regarding the Conclusions and future scope of the work.

CHAPTER 2

A Brief Review of Existing MINs

This chapter gives a brief review of some of the existing MINs. Section 1 explains a single stage interconnection network with figure. Section 2 describes a 3-D binary cube, re-circulating cube and multi-stage cube network. Section 3 gives an overview of different types of shuffles and different shuffle exchange networks. In section 4 and 5, extra-stage cube and Benes network is explained with figures. Till this section all MINs were regular i.e. contained same number of switches in every stage. From section 2.6 to 2.10, few irregular networks are reviewed.

2.1 Single Stage Interconnection Network

To solve the problem of providing fast, reliable and efficient communication at a reasonable cost in large parallel processing systems [24], different networks between the extremes of single bus and the cross bar have been proposed. Such interconnection networks can be constructed from single or multiple stages of switches. In a single stage network, data may have to be passed through the switches several times before reaching the final destination [18]. In multi-stage network, one pass of stages of switches is usually sufficient.

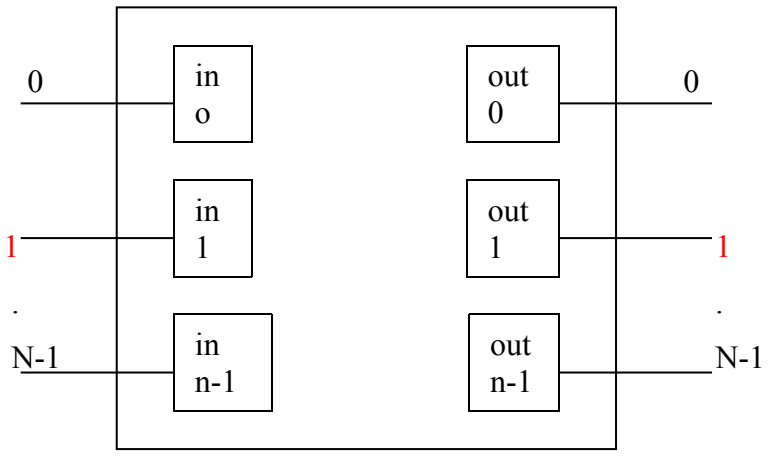


Figure 2.1: Conceptual view of a single stage interconnection network

A conceptual model of a single stage network to interconnect an N component parallel processing system is shown in Figure 2.1, which indicates that the single stage network can be viewed as an intra-connected set of N input and N output units. The way input units are connected with the output units, determine the functional characteristics of the network i.e. the allowable interconnections [26].

The single stage network is also called a re-circulating network. Data items may have to re-circulate through the single stage several times before reaching their final destination [22]. Number of re-circulations needed depends upon connectivity in a single stage network. In general, the higher is the hardware connectivity, the lesser is the number of re-circulations required [29]. Multi-stage interconnection networks are built from the stages of the basic single stage network.

2.2 Cube interconnection network

In a cube, vertical lines connect vertices whose addresses differ in most significant bit position, Vertices at both ends of diagonal lines differ in middle bit position [9]. Horizontal lines differ in least significant bit positions. The unit cube concept can be extended to an n-dimensional unit space, called n-cube, with n bits per vertex.

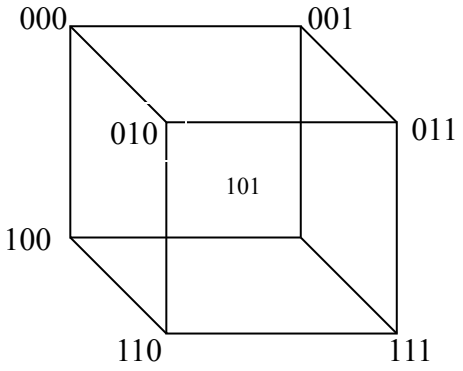


Figure 2.2: A three dimensional binary cube

The implementation of a single stage cube network is given in the Figure 2.3 for 8 nodes. The interconnection of the switching elements, corresponding to three routing functions is given separately in this Figure.

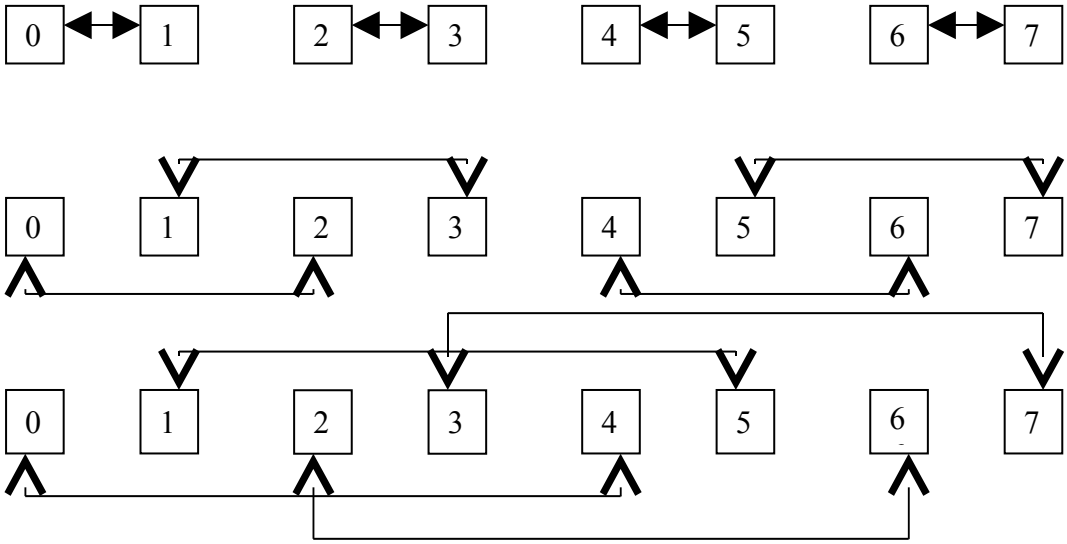


Figure 2.3: The re-circulating Cube network for N = 8

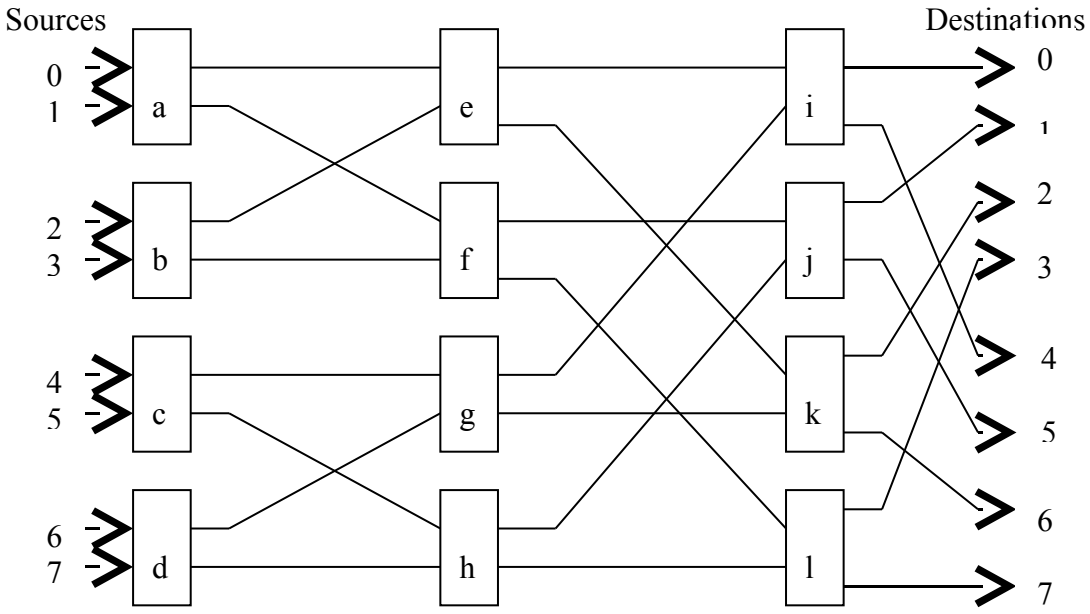


Figure 2.4: A multi-stage Cube network for N =8

The same set of cube routing functions, c_0, c_1, c_2 can also be implemented by using a three stage cube network [22]. Two functions switch boxes i.e. straight and exchange is used in constructing multi-stage cube network [9]. The stages are numbered as 0 at input end and increased to n-1 at output. The stage i implements C_i routing function for i = 0,1,2..(n-1). So, switch box at stage i connect an input line to output line that differs from it only at i^{th} bit position.

2.3 Shuffle Exchange Network

Shuffle exchange network is based on two routing functions that are Shuffle and Exchange [5]. A perfect shuffle of N = 8 is shown in Figure 2.5 Perfect shuffle cuts the deck into two halves from the center and intermixes them evenly [24]. Inverse perfect shuffle does the opposite to restore the original ordering as shown in Figure 2.6.

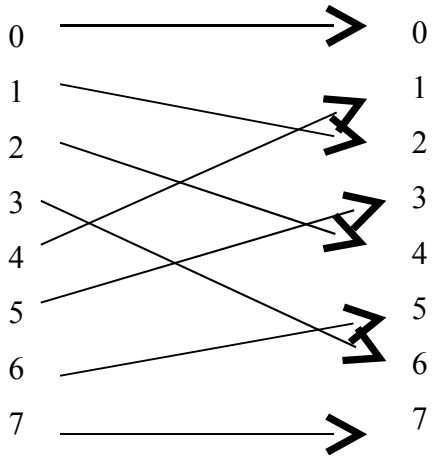


Figure 2.5: A perfect shuffle

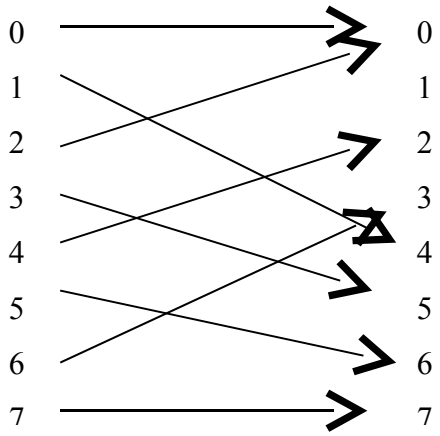


Figure 2.6: The inverse perfect shuffle

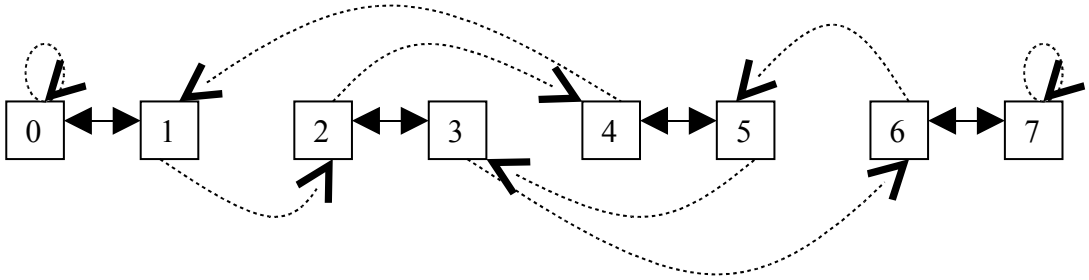


Figure 2.7: A shuffle exchange re-circulating network for N = 8

These shuffle exchange functions can be implemented as either re-circulating network or a multi-stage network. Figure 2.7 represent a single stage re-circulating shuffle exchange network, where solid lines indicate exchange and dashed lines indicate shuffle [5].The shuffle exchange has been implemented with multi-stage Omega network by Lawrie [24]. Figure 2.8 represents Omega network for N= 8. An N x N Omega network consists of $\log_2 N$ identical stages and between two stages there is a perfect shuffle interconnection. Each stage has N/2 switch boxes under independent box control [23]. Each box has four functions, straight, exchange, upper broadcast and lower broadcast.

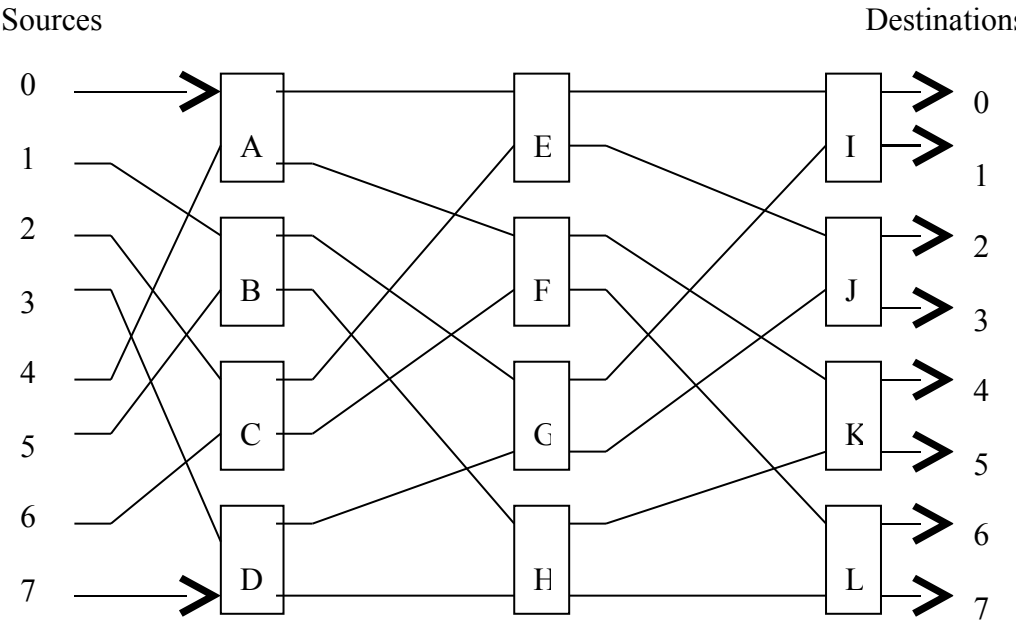


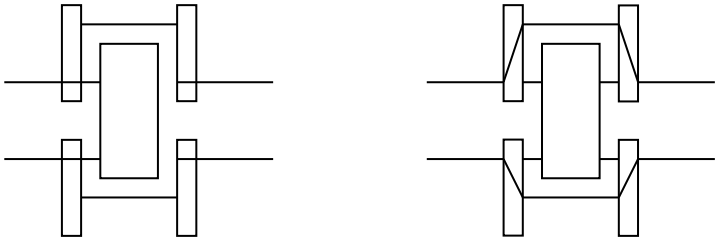
Figure 2.8: A shuffle exchange network for N = 8 (Omega network)

2.5 Extra stage cube network

There is only one path from a given input to a given output, in a generalized cube network. So, if there is a fault on that path, no communication is possible between corresponding source and the destination. However, by providing the generalized cube network with an additional stage, enough redundancy is created to allow for fault tolerant performance. This generalized cube network with an extra stage is known as Extra stage cube network [14]. The extra stage provides an additional path from each source to the destination.

The extra stage is placed on the input side of the network and implements C_0 cube interconnection function when its interchange boxes are set to swap [23]. The

extra stage and stage zero can each be enabled or disabled. A stage is enabled when its interchanged boxes can perform the corresponding cube interconnections and it is disabled when its interchange boxes are bypassed. The interchange boxes of stage m share common control signal to enable or disable them and the same is true for the interchange boxes of stage zero [21].



Interchange Box enabled Interchange Box disabled

Figure 2.9: Interchange boxes for Extra cube network

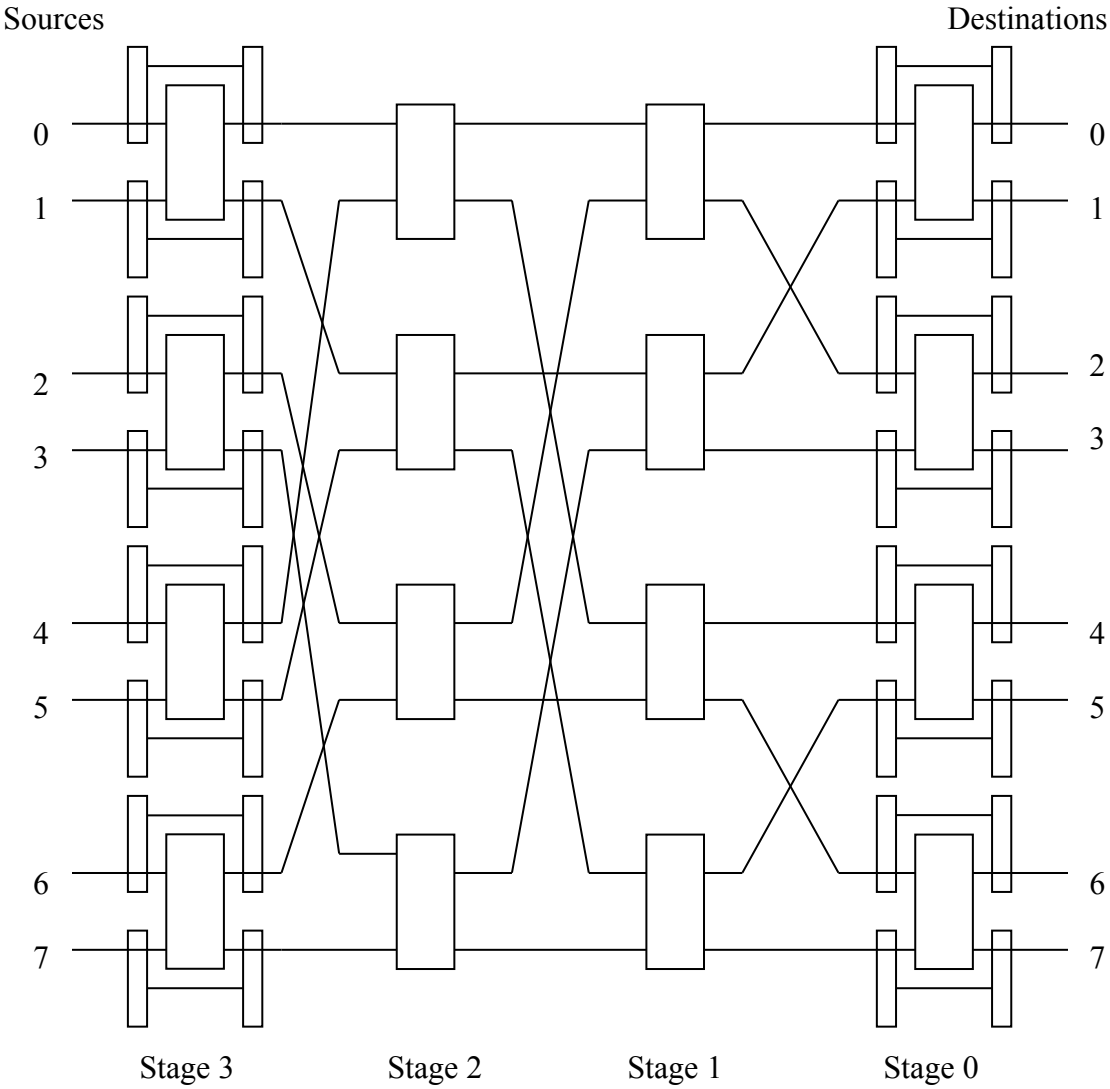


Figure 2.10: Extra stage cube network for N= 8

Enabling and disabling of stage m and stage zero is performed by the control processor. The normal operation of the network takes place with stage m disabled or bypassed and the stage zero enabled.

2.5 Benes network

Benes network is a re-arrangable network [7]. It performs all the possible connections between inputs and outputs by rearranging its existing connections so that a connection path for a new input output pair can always be established, Benes network is shown in Figure 2.11.The difference between Omega network and Benes network is that Omega network belongs to blocking network, where simultaneous connections of more than one terminal pair may result in conflict in the use of network communication links. But Benes network has more number of stages as compared to Omega networks [24]. So, unlike the Omega networks, in which there is exactly one path from any source to the destination, in Benes network, there are many alternatives. Routing in this network is much more complex as compared to Omega network, as the choice for the route is made at every step [22].

Benes network require O ($N\log_2 N$) switches. So, it has high cost [3] and overheads. This makes these networks impractical for parallel processing systems when their size n is large.

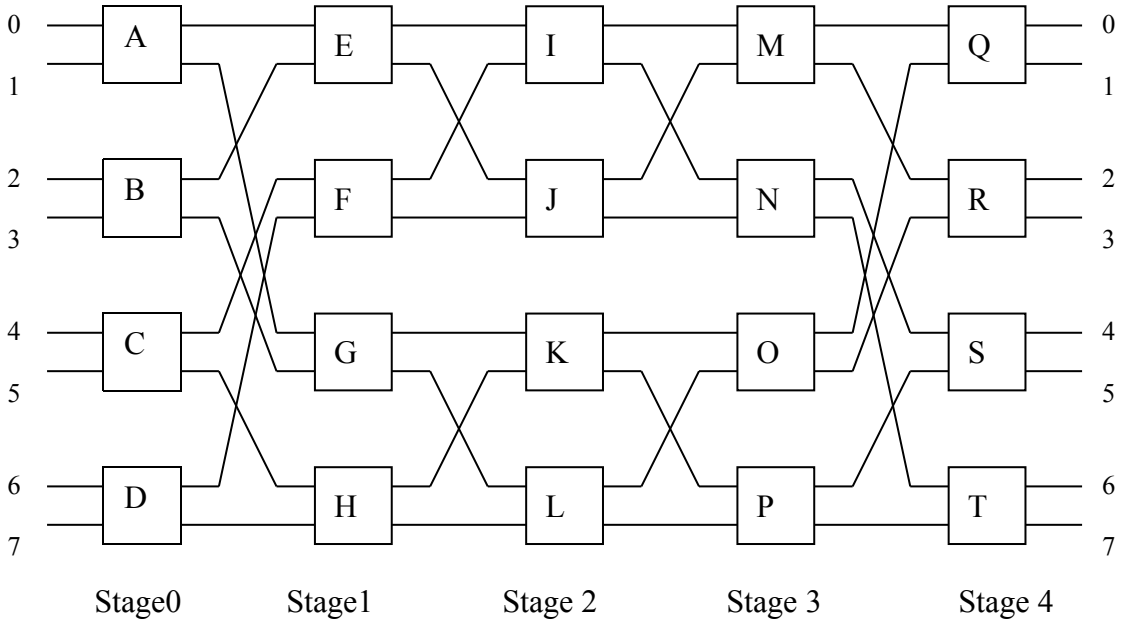


Figure 2.11: A Benes network for N =8

2.6 **Double Tree Network (DOT)**

The double tree (DOT) network was originally proposed as a fault-detecting and correcting network [3]. It is an irregular type of β network with the same number of inputs and outputs. It can be seen that there exists a direct link between β elements of an image pair. The structure of $2^m \times 2^m$ DOT network can be defined recursively as the $2^1 \times 2^1$ DOT network.

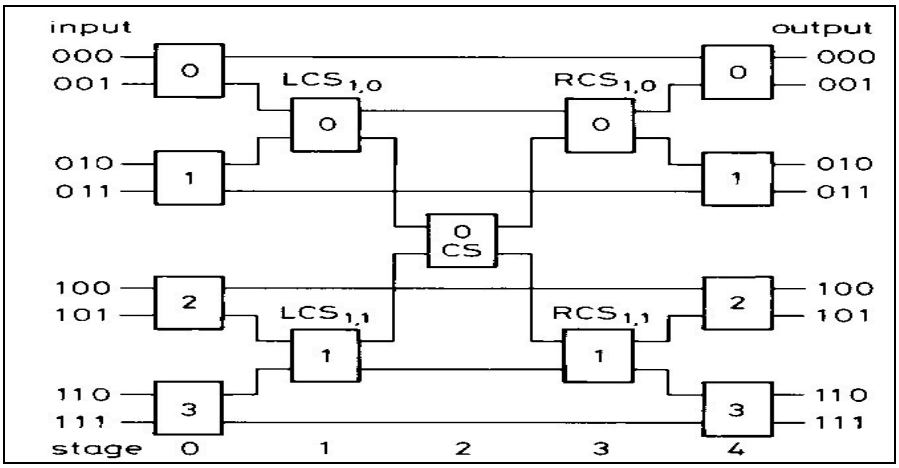


Figure 2.12: A DOT network for N =8

2.7 **Modified Double Tree Network (MDOT)**

MDOT is an example of non-fault tolerant irregular network, having different number of switches at each stage [3]. Total number of stages in this type of network are 2n-1, where n = $\log_2 N$ and the number of switching elements are $2^{n+1} - 3$. The number of switching elements at the last and first stages is equal.

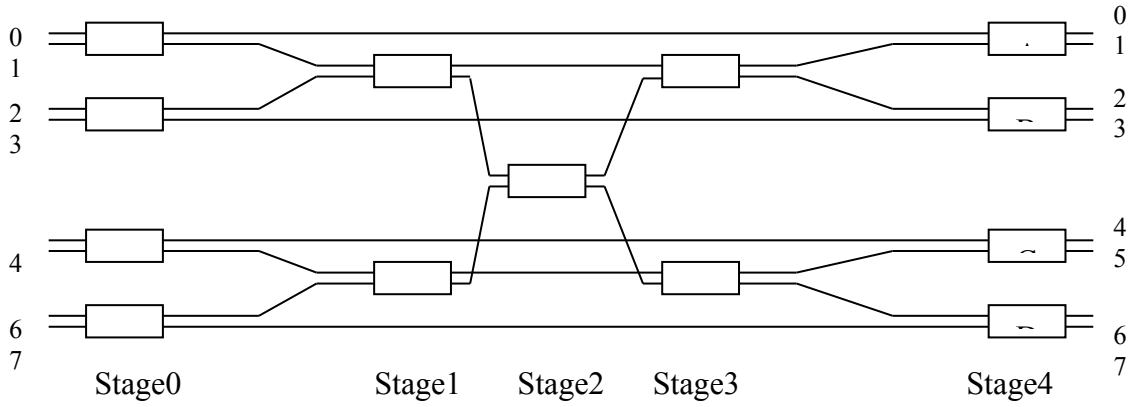


Figure 2.13: Modified double tree network for N=8

2.8 Four Tree Network (FT)

Four tree network is an irregular dynamic multi-stage interconnection network that supports multiple paths of different path lengths [20]. Two subgroups of identical sub-networks are used in its formation.

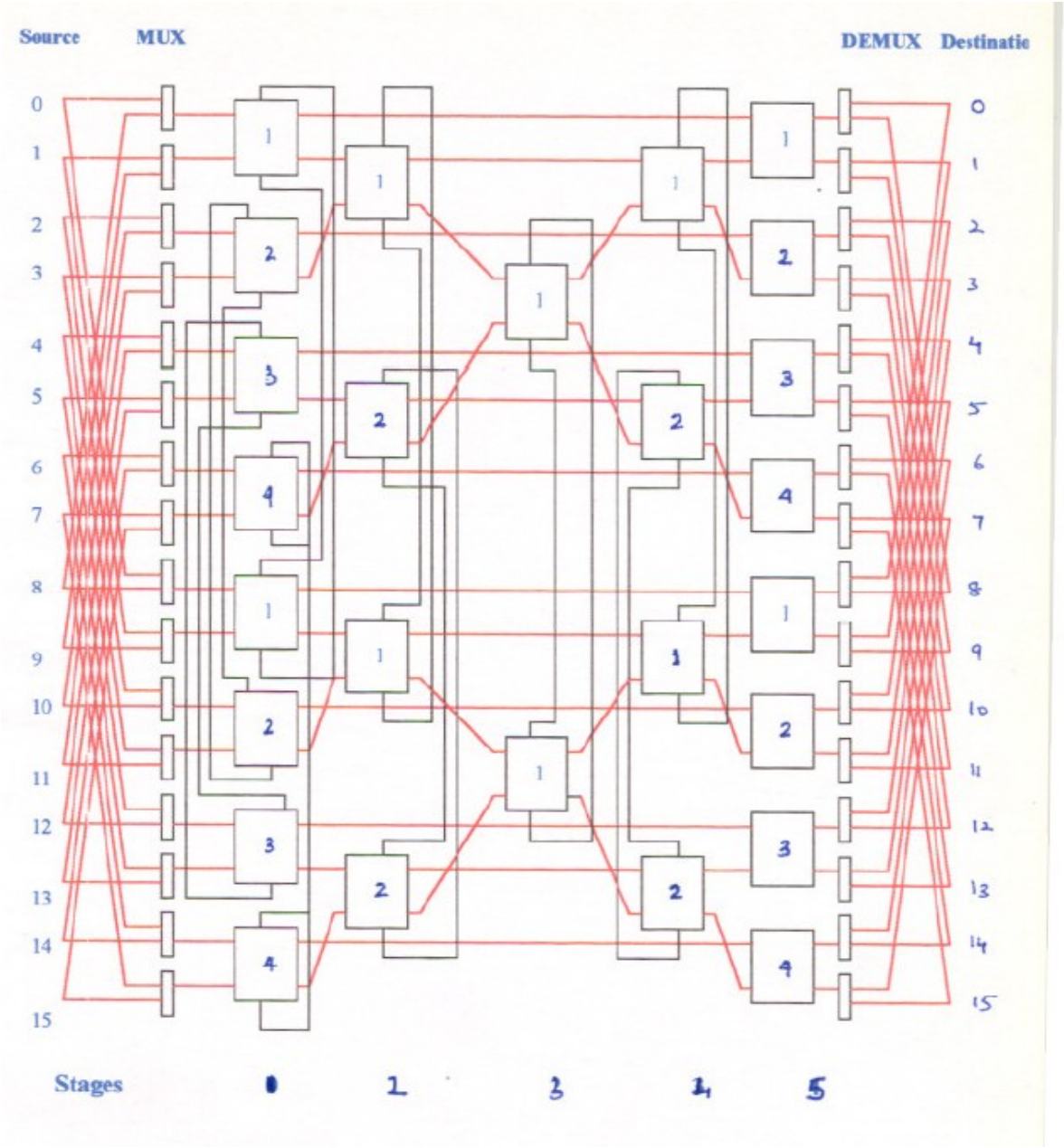


Figure 2.14: Four tree network for N=16

2.9 Modified Four Tree network

A modified FT of size 16 × 16 has one stage and two switches less as compared to the FT network. It contains total of $(2^{m+2} – 8)$ switches with 2^{n-1} of size 2 × 2 and rest of size 3 × 3. [20] It contains the same number of multiplexers and de-multiplexers as in FT network. The 2 × 1 multiplexers are connected in such a manner that each multiplexer between pair of sources have all the bits $(s_{n-2} \ldots s_2\ s_0)$ equal except the bit S_1. The 2 × 1 demultiplexers are so connected that each multiplexer is connected to the destinations that are having all bits same.

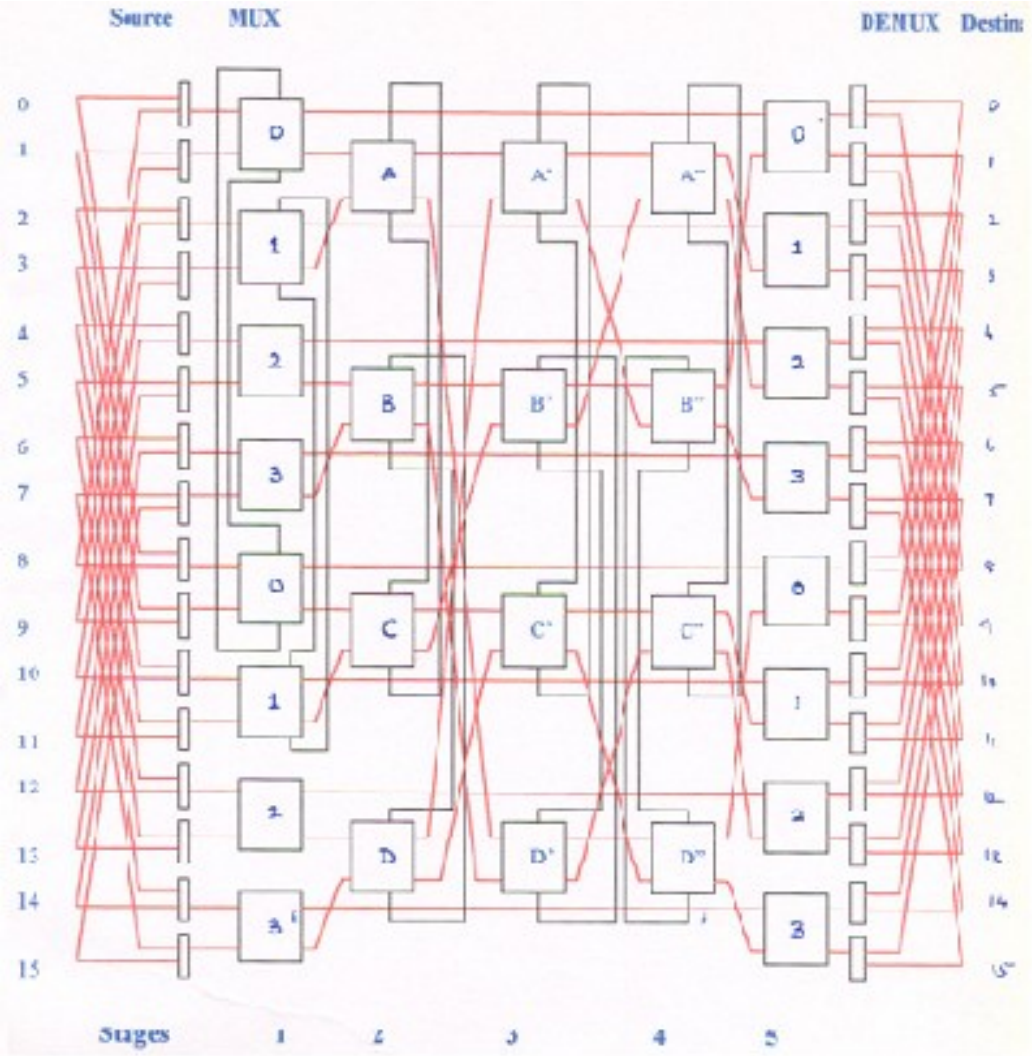


Figure 2.15: Modified Four tree network for N=16

2.10 Quad Tree Network

Quad tree (QT) network of size $2^n \times 2^n$ is constructed with the help of two identical groups G^s_{n-1} , each consisting of MDOT network of size $2^{n-1} \times 2^{n-1}$, which are arranged one above the other[3,18]. The two groups are formed based on the most significant bit (MSB) of the source/destination terminals. Thus the half of the source/destination terminals with a MSB 0 falls into G^0 group and the others having MSB as 1 falls into G^1.

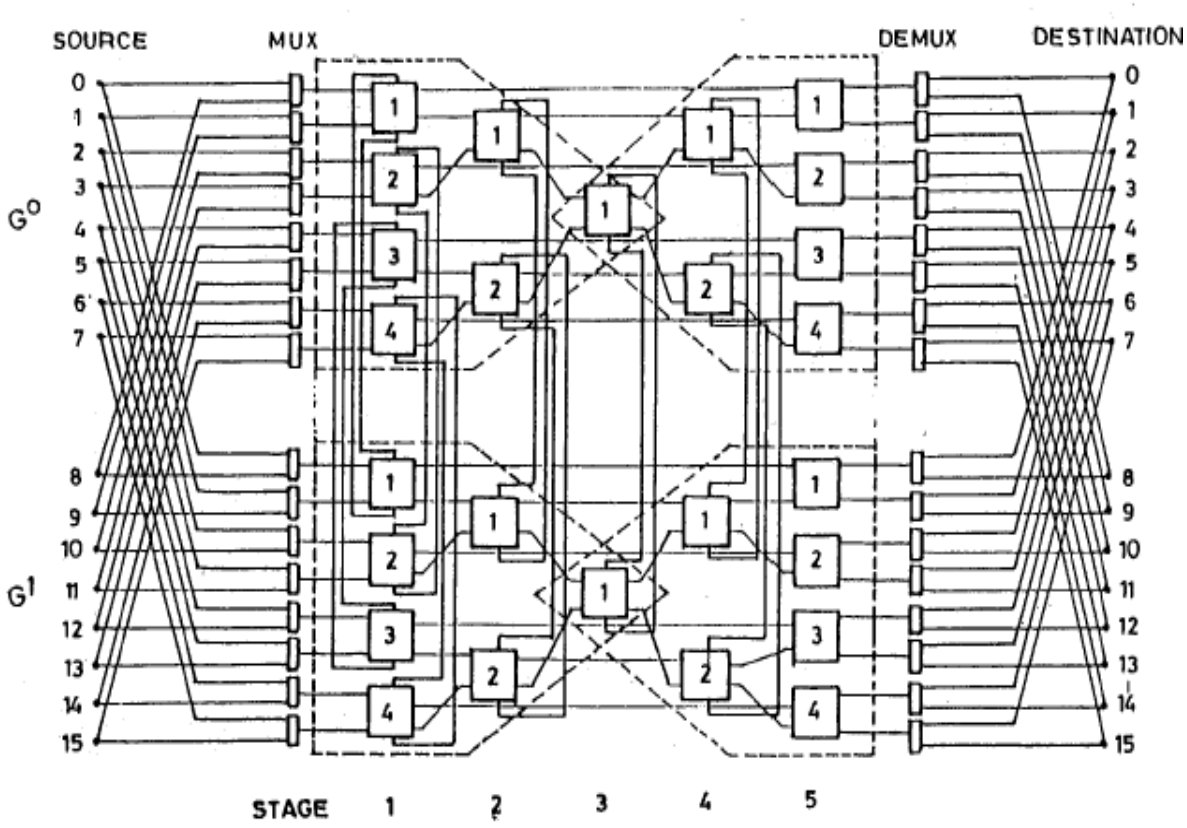


Figure 2.16: Quad network for N=16

CHAPTER 3
Research Question

3.1 Gaps between existing and targeted work

Multi-stage interconnection networks play an important role in the parallel computing systems. A multi-stage interconnection network consists of more than one stage of interconnection elements and links. Reliability and efficiency in terms of the speed of operations and the cost are the major considerations in the design of multi-stage interconnection networks. A lot of work has already been done in the design and analysis of regular multi-stage interconnection networks. Since irregular networks, in general, are less costly and inherently multi path in nature compared to regular multi-stage interconnection networks. So, analysis of irregular multi-stage interconnection network is important.

A study of different irregular multi-stage interconnection networks is made here and some modifications in the design of existing irregular MINs has been proposed. Also, performance analysis of the proposed networks is done under different parameters. Results of the analysis are presented and conclusions are drawn on the basis of result, which concludes that the proposed irregular network performs better than the proposed and existing regular networks.

3.2 Problem Statement

Objective of the research was to design new MINs which perform better than the existing MINs, in terms of reliability [6], bandwidth [10], permutation passibility [7] and cost-effectiveness [3]. Main emphasis was to design a new irregular network, since lots of work has been already done on regular networks. Irregular networks have an edge over regular network in terms of performance.

Objective of the research work can be summarized as follows:

To design fault-tolerant networks that can achieve the general goals i.e. high reliability, good performance even in the presence of faults, and low cost.

To explain the construction and routing of the proposed networks.

To calculate the reliability of fault-tolerant Irregular network in terms of Mean Time to failure.

To analyze the proposed networks in terms of permutation passibility, with respect to ABN.

To calculate the bandwidth of the proposed networks and few more parameters dependant on bandwidth namely

- Probability of Acceptance
- Throughput
- Processor Utilization

To calculate the cost of proposed networks and based on that, measure the cost-effectiveness of the networks.

CHAPTER 4

Construction and Routing Scheme

This chapter explains the construction and routing scheme for three networks namely, ABN (existing), MABN and IABN (proposed). Section 1 explains the construction of ABN and section 2 and 3 illustrates redundancy graph and routing scheme respectively, with the help of figures. In section 2, construction, redundancy graph and routing of proposed regular network MABN is described. Section 3 deals with the construction and routing of IABN along with its redundancy graph.

4.1 ABN (Augmented Baseline Network)

An Augmented Baseline Network is a baseline network with two less stages, additional intra-stage auxiliary links, multiplexers, demultiplexers, and slightly more complex switches.

4.1.1 Construction of ABN

To construct an ABN of size N i.e. N sources and N destinations, two identical groups of N/2 sources and N/2 destinations need to be formed first. Each group consists of a multiple path modified baseline network of size N/2 [4]. The modified baseline network is a baseline network with one less stage and feature links among switches belonging to the same stage and forming several loops of switches. The switches in the last stage are of size 2 x 2 and the remaining switches in stages 1 through n-3($n=\log_2 N$) are of size 3 x 3. In each stage, the switches can be grouped into conjugate subsets, where a conjugate subset is composed of all switches in a particular stage that lead to the same subset of destinations.

The modified baseline network achieves the multiple path property by permitting two switches in the same conjugate subset that are not a conjugate pair to communicate through auxiliary links [4]. The switches which communicate through the use of auxiliary links are called a conjugate loop. The conjugate loops are formed in such a way that the two switches which form a loop have their respective conjugate switches in a different loop. These pair of loops is called conjugate loops.

Each source is linked to both the groups via multiplexers. There is one 4 x 1 MUX for each input link of a switch in stage 1 and one 1 x 2 DEMUX for each output link of a switch in stage n-2. Each group consisting of a modified baseline network of size N/2 plus its associated MUXs and DEMUXs is called a subnetwork. Thus an ABN consists of two identical sub-networks which are denoted by G^i. For example, in Figure 4.1 switches A, B, C, D belonging to stage 1 of a subnetwork (G^i) form a conjugate subset, switches A and B form a conjugate pair, and switches A and C form a conjugate loop.

A source selects a particular subnetwork (G^i) based upon the most significant bit of the destination. As there are two paths between a source-destination pair, so each source is connected to two switches (primary and secondary) in a subnetwork.
The sources are connected to the switches of stage 1 as follows:
Let the source S and destination D be represented in binary code as:

$S=s_0, s_1,\ldots,s_{n-2},s_{n-1}$

$D=d_0, d_1,\ldots,d_{n-2},d_{n-1}$

(i) Source S is connected to the ($s_1,\ldots,s_{n-2}$) primary switch in both the sub-networks through the multiplexers.

(ii) Source S is also connected to the $[\{(s_1,\ldots,s_{n-2})+1\} \bmod N/4]$ secondary switch in both the sub-networks through the multiplexers.

Thus an ABN of size N consists of N number of 4 x 1 MUXs, N number of 1 x 2 DEMUXs, and n-2 stages of N/2 switches each (the last stage has 2 x 2 switches and the remaining stages have 3 x 3 switches). Observe that this construction procedure has two benefits.

The network can tolerate the failure of any switch in the network.

It provides a topology which lends itself to on-line repair and maintainability, as a loop can be removed from any stage of the ABN without disrupting the operation of the network.

Since the sub-networks are identical, so the VLSI implementation of the network becomes simple. The construction procedure given in this section can be applied to a broad class of unique path networks which are topologically equivalent. An ABN of size 16 is illustrated in Figure 4.1.

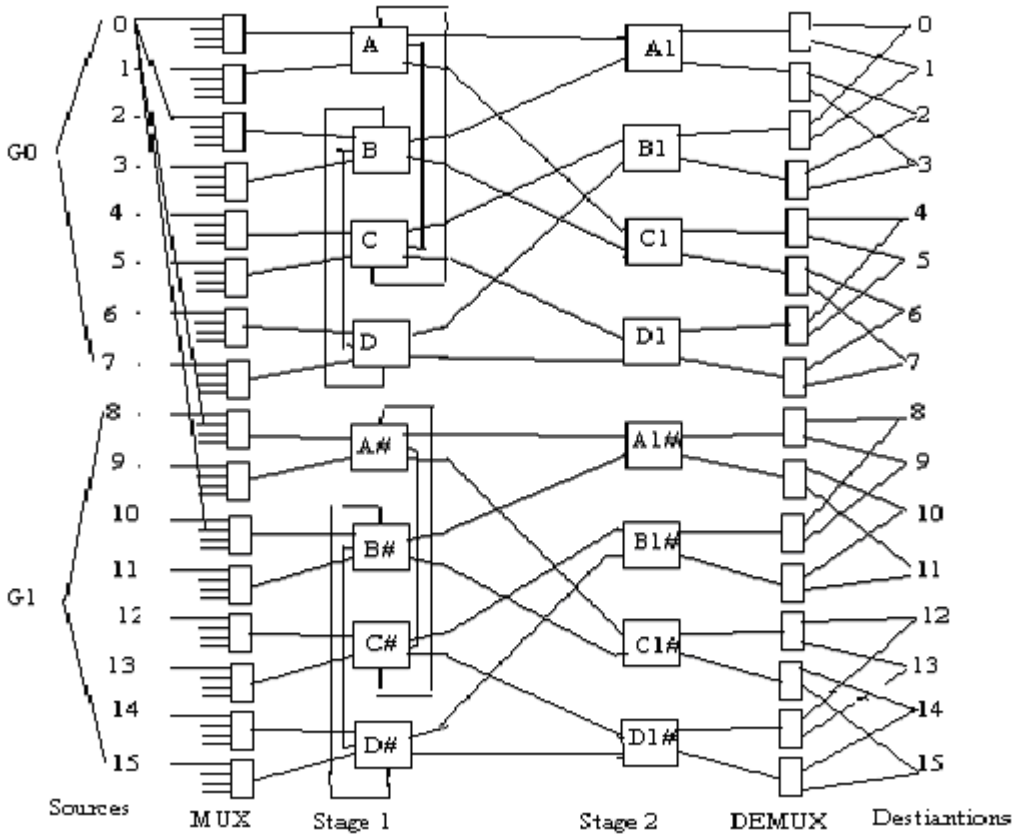


Figure 4.1: An ABN of size 16 X 16.

4.1.2 Redundancy Graph of ABN

A redundancy graph offers a convenient way to study the properties of a multi-path MIN, such as the number of faults tolerated or the type of rerouting possible [6]. A redundancy graph depicts all the available paths between a source and a destination in a MIN. It consists of two distinguished nodes-the source S and the destination D-and the rest of the nodes correspond to the switches that lie along the paths between S and D. Consider the redundancy graph of ABN as shown in Figure 4.2.

Each source is connected to two switches in one particular subnetwork; these switches are further connected to two more switches (in conjugate loop) via auxiliary links. If a switch becomes faulty, then the loop containing the faulty switch can be removed from the network and a replacement loop inserted. It is necessary to have a procedure for gracefully terminating the connections using the non-faulty switches in the loop before removing the loop. However, the overall operation of the network need not be disrupted, and the network can continue to operate at a reduced level of performance while it is being repaired.

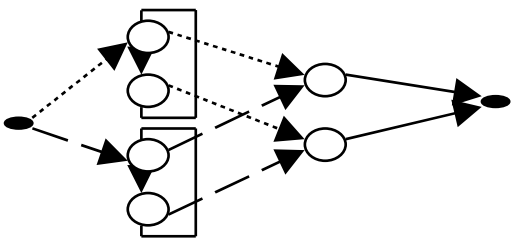


Figure 4.2: The redundancy graph of ABN.

4.1.3 Routing scheme for ABN

In this section, the routing scheme of ABNs in the case that each source-destination pair tries to utilize only one path at a time is given. The non-backtracking scheme given below is easy to implement and performs quite well. This scheme assumes that sources and switches have the ability to detect faults in the switches to which they are connected. Several techniques of detecting faults have been reported. The ABNs are self-routing [4]. A request from any source S to a given destination D is routed through the ABN as:

1) The source S selects one of the sub-network G^i based on the most significant bit of the destination D ($i=d_0$).
2) There are two parts, i.e. Primary and Secondary, between each source-destination pair. Each source attempts entry into the ABN via its primary path. If the primary path is faulty (i.e. either MUX or primary switch or both are faulty), then the request is routed to secondary path. If the secondary path is also faulty then the ABN is failed.
3) After the MUX, the routing of the request in the intermediate stages of the subnetwork depends upon (n-2) tag bits. The routing tag of the ABN is the destination address with its most significant bit being trimmed (i.e. routing tag = $d_1,\ldots d_{n-2},d_{n-1}$). For each switch in stage i ($i< n-2$), use tag bit d_i and route the request through the usual output link, if it is busy or if the successor switch (in the next stage) is faulty, route the request via the auxiliary output links to the other switch in the loop with the same tag bit d_i. If the auxiliary link is also unusable because it is busy or because of a fault then drop the request. A faulty DEMUX at the output of the ABN is regarded as a failure of its associated switch in stage n-2. This strategy essentially enables a switch to detect a failure of its successor switch and re-routes the request whenever possible.

4) For a request at a switch in stage n-2, use bit d_{n-2} of the routing tag and route the request accordingly to one of the output links. If the required output link is busy, drop the request.
5) For routing a request through a DEMUX, use bit d_{n-1} of the routing tag.

Since there are two choices to route at each step, except in stage n-2 (where it is assumed that a fault in a DEMUX at the output of a switch is a fault in that switch and the destinations are fault free), it is clear that the routing procedure delivers a request from a source to any required destination in the presence of single switch failure.

Multiple paths between S=0000 and D=0100 of an ABN are highlighted in Figure 4.3. For connections between S=0000 and D=0100, switch A and switch B of stage 1 belonging to subnetwork G^0 act as the primary and the secondary switches respectively. The paths connecting, source and the primary switch is named as the primary path and the source and the secondary switch is named as the secondary path.

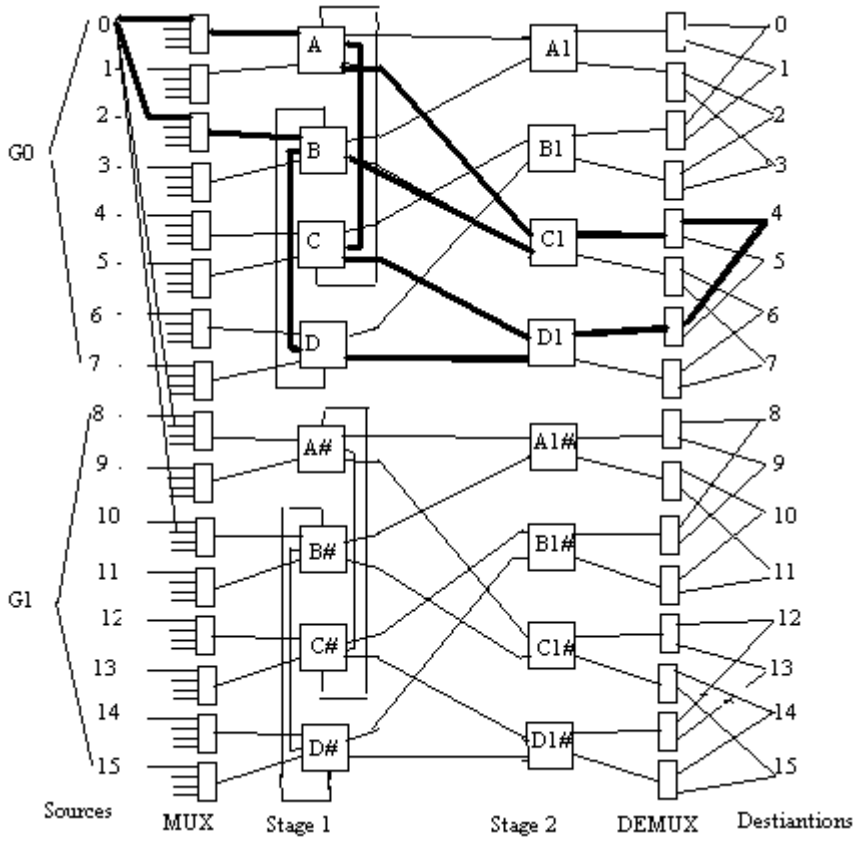


Figure 4.3: Routing in ABN

Primary path: 0->MUX (0) ->A->C1->DEMUX (4) ->4

0->MUX (0) ->A->C->D1->DEMUX (6) ->4

Secondary path: 0->MUX (2) ->B->C1->DEMUX (4) ->4

0->MUX (2) ->B->D->D1->DEMUX (6) ->4

This routing algorithm can be easily incorporated in the hardware design of a switching element. The design of the switches can be uniform for all stages. At first, the switches in the final stage appear to be different in terms of the number of links and also because they are required to report faults in the demultiplexers at their outputs to the switches in the previous stage. However, designing all the switches in such a way that they report themselves to be faulty whenever their auxiliary link and at least one other output link lead to faulty switches, and then permanently typing the auxiliary output links of the switches used in the final stage to the faulty state, solves the uniformity problem.

Notice that the routing algorithm is independent of the number of switches in a loop. Thus, once the switching element is designed, loops of any number of switches can be formed to meet the performance and reliability requirements and the physical design restrictions.

The probability of success of the routing algorithm, called the effective terminal reliability, or ETR, is the same for all source/destination pairs if faults occur uniformly among the switches, since ABNs are symmetric networks [2].

CHAPTER 5
Performance Analysis

Many performance parameters are applicable for MINs. Some of the important performance parameters are briefed here, which are analyzed in this thesis:

Permutation passibility means how many input requests occurring simultaneously at the input are able to pass through a given network, means how many of them will successfully mature i.e. will reach their respective destinations [7].

Fault tolerance is a criterion that must be met for the network which has tolerated a given fault or faults. A network is called single fault tolerant if it can tolerate or function in case of a single fault [9]. In general, if any set of i-faults can be tolerated by a network, then network is called i-fault tolerant [3].

Bandwidth is defined as the number of PE requests honored per unit of time. It can also be defined as the mean number of active memory modules in a transfer cycle of the IN or the expected number of destination receiving requests in any given cycle [10].

Throughput is the maximum number of traffic accepted by the network per unit time [2]. The average number of packets delivered from source to destination by network in unit time is called Throughput. Measured as packets per node per cycle [6]. It can also be defined as the average number of cells delivered by the network per unit time.

Probability of Acceptance (P_a) is often used parameter in synchronous analysis and is defined as the ratio of expected bandwidth to the expected number of requests generated per cycle [21].

Processor Utilization (PU) is the expected percentage of time, a processor is active doing internal computation without accessing the global memory.

Reliability means, in the presence of multiple CPU's if one goes down, the others may be able to take over its work [6]. It is one of the major design issues, for any network.

Cost Effectiveness: cost of a switch is proportional to the number of gates involved, which is roughly proportional to the number of 'cross points' within a switch [3]. For example, a 4 x 4 switch has 16 units of hardware cost whereas a 2 x 2 switch has 4 units. For the multiplexers and demultiplexers, we roughly assume that each of K x 1 multiplexers or 1 x K demultiplexers has K units of cost. Cost-effectiveness is given by MTTF divided by cost.

5.1 Permutation Passibility

In unique path multi-stage interconnection networks there is a unique path between a source and a destination. While multi path multi-stage interconnection networks provide more than one path from a source to a destination. In multi path multi-stage interconnection networks, the request is first tried through the most favorable path, generally, the minimum length path. The request is routed through an alternative path, if the most favorable path is not available due to one of the following reasons:

Some switch(s) may be faulty

Some other request is occupying that path at that instant of time.

If there is no alternative path available, the request is dropped. The desirable character of any network is that the network should be such that it allows maximum requests through it with minimum path length. A minimum path length means a request will:

take less time to reach destination

have less probability of intermediate link failures

In this chapter, permutation passibility of three networks is calculated. This will show that at a particular moment of time, if a number of requests simultaneously occur at source, how many of them will successfully mature i.e. will reach their destination.

The following multi-stage interconnections networks are analyzed in this chapter on the basis of two parameters, i.e. number of requests reaching the destination and average path length.

Augmented Baseline Network(ABN)

Modified Augmented Baseline Network (MABN)

Irregular Augmented Baseline Network(IABN)

Different combinations of source destination pairs are selected and their path length is calculated for these networks, only if a request matures successfully. Unsuccessful requests are indicated by CLASH. The average path length is defined as the ratio of sum of the path lengths used in the successful requests to the total number of requests. FAULT indicates faulty component encountered while establishing path.

5.1 Permutation passibility in absence of faults

5.1.1 CASE I: Total number of requests: 10, Number of faults: 0

(2,3)	→	2	MUX(2)	B	A1	DEMUX(1)	3	
(3,11)	→	3	MUX(11)	B#	A1#	DEMUX(9)	11	
(12,14)	→	12	MUX(12)	C#	D1#	DEMUX(15)	14	
(6,4)	→	6	MUX(6)	D	D1	DEMUX(6)	4	
(9,3)	→	9	MUX(1) MUX(1)	A A	A1 C	DEMUX(1) B1	CLASH DEMUX(3)	3
(10,0)	→	10	MUX(2) MUX(4)	CLASH C	B1	DEMUX(2)	0	
(7,4)	→	7	MUX(7) MUX(7)	D D	D1 B	CLASH C1	DEMUX(4)	4
(5,2)	→	5	MUX(5) MUX(5)	C C	B1 A	CLASH A1	DEMUX(1)	2
(0,7)	→	0	MUX(0)	A	C1	DEMUX(5)	7	
(8,7)	→	8	MUX(0) MUX(2)	CLASH CLASH				

Number of requests matured successfully: 9

Average path length: (2+2+2+2+3+2+3+3+2)/9=2.33

5.1.2 CASE II: Total number of requests: 12, Number of faults: 0

1) ABN

(1,6)	→	1	MUX(1)	A	C1	DEMUX(5)	6	
(3,4)	→	3	MUX(3)	B	C1	DEMUX(6)	4	
(2,10)	→	2	MUX(10)	B#	A1#	DEMUX(9)	10	
(4,8)	→	4	MUX(12)	C#	B1#	DEMUX(10)	8	
(5,12)	→	5	MUX(13)	C#	D1#	DEMUX(14)	12	
(6,7)	→	6	MUX(6)	D	D1	DEMUX(7)	7	
(8,1)	→	8	MUX(0)	A	A1	DEMUX(0)	1	
(9,5)	→	9	MUX(1) MUX(3)	CLASH CLASH				
(7,13)	→	7	MUX(15) MUX(15)	D# D#	D1# B#	DEMUX(14) C1#	CLASH DEMUX(12)	13
(10,6)	→	10	MUX(2) MUX(2)	B B	C1 D	CLASH D1	DEMUX(7)	6
(12,14)	→	12	MUX(14)	D#	D1#	DEMUX(15)	14	
(15,3)	→	15	MUX(7)	D	B1	DEMUX(3)	3	

Number of requests matured successfully: 11

Average path length: (2+2+2+2+2+2+2+3+3+2+2)/11=2.1818

5.1.3 CASE III: Total number of requests: 14, Number of faults: 0

(5,0)	→	5	MUX(5)	C	B1	DEMUX(2)	0	
(8,12)	→	8	MUX(8)	A#	C1#	DEMUX(12)	12	
(3,9)	→	3	MUX(11)	B#	A1#	DEMUX(8)	9	

(4,10)	→	4	MUX(12)	C#	B1#	DEMUX(11)	10	
(0,0)	→	0	MUX(0)	A	A1	DEMUX(0)	0	
(12,6)	→	1 2	MUX(4)	C	D1	DEMUX(7)	6	
(6,4)	→	6	MUX(6)	D	D1	DEMUX(6)	4	
(7,8)	→	7	MUX(15)	D#	B1#	DEMUX(10)	8	
(15,15)	→	1 5	MUX(15) MUX(9) MUX(9)	CLASH A# A#	C1# C#	CLASH D1#	DEMUX(15)	15
(1,2)	→	1	MUX(1) MUX(1)	A A	A1 C	CLASH B1	DEMUX(3)	2
(11,13)	→	11	MUX(11) MUX(13)	CLASH C#	D1#	DEMUX(14)	13	
(2,3)	→	2	MUX(2)	B	A1	DEMUX(1)	3	
(9,7)		9	MUX(1) MUX(3)	CLASH B	C1	DEMUX(5)	7	
(13,11)		1 3	MUX(13) MUX(15)	CLASH CLASH				

Number of requests matured successfully: 13

Average path length: (2+2+2+2+2+2+2+2+3+3+2+2+2)/13=2.154

5.2 Permutation passibility in presence of faults

This section gives the permutation passibility of three networks ABN, MABN and IABN in presence of faults.

5.2.1 CASE I: Total number of requests: 10, Number of faults: 3

1) ABN

(2,3)	→	2	MUX(2)	B	A1	DEMUX(1)	3	
(3,11)	→	3	MUX(11)	B#	A1#	DEMUX(9)	11	
(12,14)	→	12	MUX(12)	C#	D1#	DEMUX(15)	14	
(6,4)	→	6	MUX(6) MUX(6)	D D	D1 B	FAULT C1	DEMUX(4)	4
(9,3)	→	9	MUX(1) MUX(1) MUX(3)	A A C	A1 C FAULT	DEMUX(1) FAULT	CLASH	
(10,0)	→	10	MUX(2) MUX(4)	CLASH C	FAULT			
(7,4)	→	7	MUX(7) MUX(1) MUX(1)	D A A	D1 A1 C	FAULT DEMUX(1) FAULT	CLASH	
(5,2)	→	5	MUX(5) MUX(7)	C D	FAULT B1	DEMUX(3)	2	
(0,7)	→	0	MUX(0)	A	C1	DEMUX(5)	7	
(8,7)	→	8	MUX(0) MUX(2)	CLASH CLASH				

Number of requests matured successfully: 6

Average path length: (2+2+2+3+2+2)/6=2.16

5.2.2 CASE II: Total number of requests: 12, Number of faults: 3

1) ABN

(1,6)	→	1	M(1)	A	C1	FAULT		

			M(1)	A	C	D1	D(7)	6
(3,4)	→	3	M(3) M(5)	B C	FAULT D1	D(6)	4	
(2,10)	→	2	M(10)	B#	A1#	D(9)	10	
(4,8)	→	4	M(12)	C#	B1#	D(10)	8	
(5,12)	→	5	M(13) M(13)	C# C#	D1# A#	FAULT C1#	D(12)	12
(6,7)	→	6	M(6)	D	D1	D(7)	7	
(8,1)	→	8	M(0)	A	A1	D(0)	1	
(9,5)	→	9	M(1) M(3) M(9)	CLASH B A#	FAULT C1#	D(13)	5	
(7,13)	→	7	M(15) M(15) M(9)	D# D# CLASH	D1# B#	FAULT C1#	D(12)	CLASH
(10,6)	→	1 0	M(4) M(4) M(6)	C C CLASH	D1 A	CLASH C1	FAULT	
(12,14)	→	1 2	M(12) M(14) M(14)	CLASH D# D#	D1# B#	FAULT C1#	D(13)	14
(15,3)	→	1 5	M(7)	D	B1	D(3)	3	

Number of requests matured successfully: 10

Average path length: (3+2+2+2+3+2+2+2+3+2)/10=2.3

5.2.3 CASE III: Total number of requests: 14, Number of faults: 3

1) ABN

(5,0)	→	5	MUX(5)	C	B1	DEMUX(2)	0	
(8,12)	→	8	MUX(8)	A#	C1#	DEMUX(12)	12	
(3,9)	→	3	MUX(11)	B#	A1#	DEMUX(8)	9	

(4,10)	→	4	MUX(12)	C#	B1#	DEMUX(11)	10	
(0,0)	→	0	MUX(0)	A	A1	DEMUX(0)	0	
(12,6)	→	1 2	MUX(4)	C	D1	DEMUX(7)	6	
(6,4)	→	6	MUX(6)	D	D1	DEMUX(0)	4	
(7,8)	→	7	MUX(15)	D#	D1#	FAULT		
(15,15)	→	1 5	MUX(15) MUX(9) MUX(9)	CLASH A# A#	C1# C#	CLASH D1#	FAULT	
(1,2)	→	1	MUX(1) MUX(1)	A A	A1 C	CLASH B1	DEMUX(3)	2
(11,13)	→	11	MUX(11) MUX(13)	CLASH C1	FAULT			
(2,3)	→	2	MUX(2) MUX(4)	B CLASH	FAULT			
(9,7)	→	9	MUX(1) MUX(3)	CLASH B	FAULT			
(13,11)	→	1 3	MUX(13) MUX(15)	C1 CLASH	FAULT			

Number of requests matured successfully: 8

Average path length: (2+2+2+2+2+2+2+3)/8 = 2.125

5.2 Reliability analysis

This section deals with the reliability analysis of proposed networks MABN and IABN in respect to the existing network ABN. Block diagrams and reliability equations of these three networks are given in this section.

5.2.1 Introduction to MTTF

Mean Time to Failure (MTTF) is a well known criterion to measure reliability of fault-tolerant networks having full access [4]. Under this criterion, a network is faulty if there is any source-destination pair that cannot be connected because of faulty components in the network. MTTF of the network is defined as the expected time elapsed before some source is disconnected from some destination.

To achieve fault tolerance, we exploit the fact that there are subsets of switches in each stage which lie on paths leading to the destinations. All the switches in a given stage which lead to the same subset of destinations comprise a conjugate subset of partitioned into several conjugate subsets. The partial routing tag required to set up a connection to a reachable destination from either switch of a conjugate subset is also the same. In each conjugate subset of switches, there are several pairs of switches called conjugate pairs of switches [17]. The switches in such a pair are connected to the same switches in the next stage. Conjugate subsets and conjugate pairs of switches play a fundamental role. If a switch is not able to process a request for connection because of a faulty switch in the next stage or because of a busy link, it can route that request via its auxiliary output link to the next switch in the loop. The next switch can then make a connection to a different (non-faulty) switch in the following stage. In fact, using an auxiliary link whenever a fault is encountered allows any source to be connected to any destination while tolerating any single faulty switch in any stage other than the initial stage and the final stage.

Notice that the loops formed in all stages except the final stage are such that for every loop there exists another loop which is connected to the same set of switches in the next stage. Such pairs of loops are called conjugate loops. In the final stage, since a loop consists of only one switch, there are in effect conjugate pairs of switches connected to the same subset of destinations.

A major advantage of implementing the loops as modules is that such implementation makes MINs easier to maintain and repair. As we will show later, the fault tolerance of ABN and two new proposed networks is not limited to single switch

faults but extends to faults affecting all the switches in a given loop. In other words, the removal of a loop as a whole from a network need not disrupt the continued operation of the network. Thus, when a switch fault is identified, the loop containing the faulty switch can be removed from the network and a replacement loop inserted without interrupting the operation of the network. If more than one loop can be accommodated on a single printed-circuit board or a VLSI chip, then no two loops on a board or chip must form a conjugate pair of loops. It is economically attractive to have all the loops are of the same size so that only one type of replacement board is needed.

The reliability and performance improvement obtained from a multi-path network depend upon how effectively the alternate paths available are used by the routing algorithm. One can use a backtracking routing algorithm that exhaustively searches for an available fault-free path. However, implementation of backtracking is relatively expensive in terms of hardware and backtracking can, in some situations; take an inordinately long time to set up connections. The non-backtracking algorithm given below is easy to implement and performs quite well. This algorithm assumes that sources and switches have the ability to detect faults in the switches to which they are connected.

Several techniques for detecting faults in multi-stage networks have been reported.5 Such techniques usually require off-line application of test inputs or make the test results available only to the external resources attached to the network. If adaptive routing is to be implemented, with the switches making the routing decisions, then fault information is required by the individual switches at the time they process a request. In such situations, off-line diagnosis methods are not adequate. It is necessary to design switching elements capable of detecting faults in adjoining switching elements as they occur and concurrently with normal operation.

We evaluate the MTTF of the given networks using a simple series-parallel reliability model. The two loops in a conjugate pair are in parallel and all the conjugate pairs of loops are in series. From the redundancy graph, we can note that in each stage there are two "gateway" switches that receive connections from the previous stage. These two form, in fact, a conjugate pair of switches. Also, switches in every conjugate pair in the network act as gateways to the stage they belong to, for some source-destination pair. Thus, if both the switches in a conjugate pair fail, then clearly some source is connected from some destination. If we assume that the

network becomes faulty only when a conjugate pair of switches is present in a fault pattern (an optimistic assumption), then we get an upper bound for its MTTF. This case also has a series-parallel model, with the switches in a conjugate pair in parallel and all the conjugate pairs in series. The optimistic and pessimistic estimates of the MTTF of ABN, MABN and IABN are discussed below. It is assumed that the switches have a Poisson failure rate of 10^{-6} per hour.

There are two types of fault models adapted to the reliability analysis of networks; 'switch-fault model' and 'link-fault model'. In the former, a switch is considered to be totally unusable if it becomes faulty. But in the later, a switch can be partially operational even if it contains some faulty links. In this paper, the switch-fault model is used for the analysis of networks. We assume that any of the switching components, i.e. crossbar switches [21], multiplexers or demultiplexers, in networks can fail. We also assume that faults are independent of each other.

A network is single-fault tolerant if it can function as specified by its fault-tolerance criterion despite any single fault conforming to its fault model. More generally, if any set of i faults can be tolerated, then a network is i-fault tolerant. A network that can tolerate some instances of i faults is robust although not i-fault tolerant. Under the criterion of full access, the reliability of a network can be measured in terms of Mean Time to Failure (MTTF). Before we are involved in the analysis of MTTF, the number of faults that networks can tolerate is worth being mentioned. A network is said to be k-fault tolerant, if it still provide a connection for any source-destination pair in the presence of any instance of up to k faults in the network. Since ABNs provides two disjoint paths for each source-destination pair, so ABNs are single switch fault-tolerant. But one can find combination of two switches, which when simultaneously faulty, can disconnect a source from a destination. For instance, if both the switches to which a source or a destination is connected become faulty, then that source or destination is disconnected from the rest of the network. However, if such critical combinations of switches are not present in a fault-pattern, several multiple faults can be tolerated.

In this section the reliability of the ABN, MABN and IABN in terms of Mean time to Failure (MTTF) is analyzed. To make the analysis traceable, we need to have some assumptions. We use the assumptions similar to ones that have been made previously in the other studies of the fault tolerant networks. The assumptions used in the analysis on the failure rates of the components are given below [20]:

Switch failures occur independently in a network with a failure rate of for 2x2 crossbar switches (a reasonable estimate for is about 10^{-6}per hour)

Failure of the multiplexers and demultiplexers also occur independently with failure rates of $_m$ and $_d$ receptively, which can be different from . In general, more complicated components lead to higher failure rates. Assuming that the hardware complexity of a component is directly proportional to the gate counts of it, one can derive a failure rate of the component.

Assuming that the hardware complexity of a component is directly proportional to the gate counts of it, one can derive a failure rate of the component. Based on gate counts of crossbar switches, the number of gates in a 2x2 crossbar switch is approximately equal to that in a 4 x 1 MUX or a 1 x 4 DEMUX. Thus to simplify the analysis, we can assume that for a 3x3 switching element $_3$ = 9/4 (i.e. 2.25) and that for a mx1 MUX $_m$ = m /4 for $_d$ (= $_m$) for a 1xm DEMUX (i.e. $_m$ = $_d$ = m /4).

Irregular MINs are inherently multi-path and the MTTF needs to be calculated at all existing path-lengths separately based upon the series and parallel models of reliability. The adaptive routing scheme described in section 3 considers a 2 x 2 switch in the last stage and its associated demultiplexers as a series system, so we consider these three elements as a single component (SE_{2d}), and based on gate count, a failure rate of $_{2d}$=2 can be assigned to this group of elements. Also let $_3$ be the failure rate for the 3 x 3 switch (SE_3), then based on gate count, $_3$ = 9/4 (i.e. 2.25).

The multiple paths in an irregular MIN have varying path lengths which are chosen adaptively by the routing algorithm, which in turn depends on the faults present, the shortest being just 2 for a favorite memory module. The presence of faults forces a longer path to be taken where no damaged SEs are present. The one-to-one connection between input and output are also called 'permutations' [7]. The delay encountered in the journey to complete a permutation decides the latency [2] of the network. The reliability of the network degrades exponentially as a function of time and the availability of the network again depends crucially on the degradation of each, SE encountered on the route taken to the destination.

5.2.2 Reliability analysis of ABN

In this section reliability equation of existing network ABN are derived based on the reliability block diagram for both lower bound and upper bound in terms of MTTF (Mean Time To Failure).

5.2.2.1 Upper bound (optimistic): To obtain an upper bound for the ABN, we observe that each source is connected to two multiplexers in each sub-network, and each switch has a conjugate [18]. So if we assume that the ABN is operational as long as one of the two multiplexers attached to a source (in a particular sub-network) is operational and as long as a conjugate pair (loop or switch) is not faulty, then we will permit as many as one half of the components to fail and the ABN may still be operational. This permits a simple reliability block diagram of the optimistic (upper) bound as shown in Figure5.1.

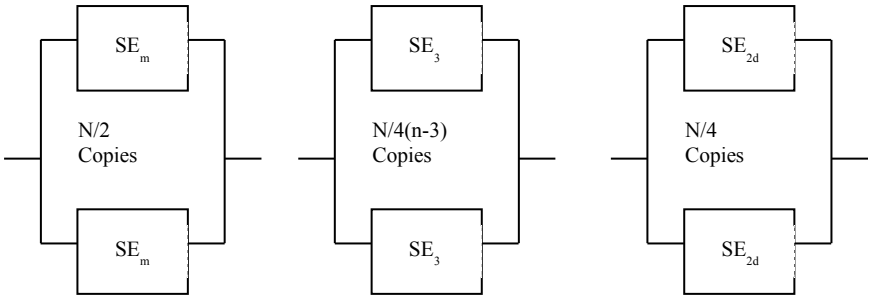


Figure 5.1: Reliability block diagram of ABN for MTTF upper bound.

The expression for the upper bound of the ABN reliability is:

$R_{ABN\text{-}ub}(t)$= f1 * f2 * f3

$$f1 = \left[1 - \left(1 - e^{-\lambda_m t}\right)^2\right]^{\left(N/2\right)}$$

$$f2 = \left[1 - \left(1 - e^{-\lambda_3 t}\right)^2\right]^{N/4(n-3)}$$

$$f3 = \left[1 - \left(1 - e^{-\lambda_{2d} t}\right)^2\right]^{\left(N/4\right)}$$

Where,

$_m$= , $_3$=2.25 , $_{2d}$=2

$$MTTF_{ABN\text{-}ub} = \int_0^\infty R_{ABN\text{-}ub}(t).dt$$

5.2.2.1 Lower bound (pessimistic): At the input side of the ABN, the routing scheme does not consider the multiplexers to be an integral part of a 3 x 3 switch. For example, as long as at least one of the two multiplexers attached to a particular switch is operational, the switch can still be used for routing. Hence, if we group two multiplexers with each switch in the input side and consider them a series system (SE_{3m}), then we will have a conservative estimate of the reliability of these components. Their aggregate failure rate will be $_{3m} = 4.25$. Finally these aggregated components and the switches in the intermediate stages can be arranged in pairs of conjugate loops. To obtain the pessimistic (lower) bound on the reliability of ABN, we assume that the network is failed whenever more than one conjugate loop has a faulty element or more than one conjugate switch in the last stage fails [18]. The reliability block diagram is shown in Figure5.2.

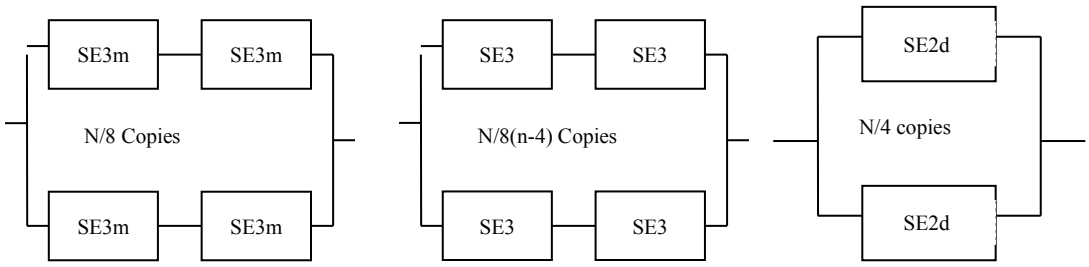


Figure5.2: Reliability block diagram of ABN for MTTF lower bound.

$R_{ABN\text{-}lb}(t)$= f1 * f2 * f3

$$f1 = \left[1 - \left(1 - e^{-2\lambda_m t}\right)^2\right]^{\left(N/8\right)}$$

$$f2 = \left[1 - \left(1 - e^{-2\lambda_3 t}\right)^2\right]^{N/8(n-4)}$$

$$f3 = \left[1 - \left(1 - e^{-\lambda_{2d} t}\right)^2\right]^{\left(N/4\right)}$$

Where,

$_{3m}$=4.25 , $_{3}$=2.25 , $_{2d}$=2

$$MTTF_{ABN\text{-}lb} = \int_0^\infty R_{ABN\text{-}lb}(t).dt$$

5.3 Bandwidth analysis

In this section bandwidth of three networks ABN, MABN and IABN is calculated in terms of probability equations. Further three performance meters based on bandwidth namely, Probability of Acceptance, Throughput and Processor Utilization are described.

5.3.1 Bandwidth

Expected number of memory modules remaining busy in a cycle or the number of memory requests accepted per cycle. It is the most common performance parameter used in analyzing a synchronous IN [24]. It is defined as the number of PE requests honored per unit of time. It is defined as the mean number of active memory modules in a transfer cycle of the IN. The term ”active” means a process is successfully performing memory operation (either read or write) in that memory module. BW also takes into account the memory access conflicts caused by the random nature of the process requests. Bandwidth (BW) is also defined as the expected number of destination receiving requests in any given cycle [10]. Thus it is the total number of requests matured. A high bandwidth is often desired at reasonably low network cost. This parameter specifies to what extent a network is efficient.

Here are the assumptions on the basis of which the probabilistic relations have been carried out [20]:

1) The IN operates in a synchronous mode, i.e., the requests issued by the processors begin and end simultaneously.

2) The requests are random and the request generated by a processor is independent of the request generated by another processor.

3) Requests which are not accepted are blocked or rejected.

4) The requests generated in a cycle are independent of the requests generated in the previous cycle.

5) “p_0” is the probability with which a processor generates a request. Thus, p_0 is the rate of request of a processor per cycle.

6) The probability with which processor P_i addresses memory M_i is zero i.e. there is no favorite memory.

7) Networks are of same size i.e. N X N.

For calculating probabilistic equations, let us suppose a MIN of size a^n x b^n with a^n sources and b^n destinations. The expected number of requests that passes per unit of time is given by $b-b(1-p/b)^a$. Dividing it by number of output lines we get rate of requests at any of the output lines [20]. So output rate, which is function of input line, is given by

$1 – (1- p/b)^a$. Output rate of final stage will determine the bandwidth of a MIN.

So Bandwidth is

$BW = b^n p_n$ where $p_0=p$.

Probability Equations for ABN

$p[1]=1-(1-p[0]/3)^3$

$p[2]=1-(1-p[1]/2)^2$

To have an idea that how these equations have been derived, Consider Figure of IABN. There are 3 stages. First and last stages have N/2 switches whereas middle stage has N/4 switches. As discussed above probability to reach i^{th} stage is given by $1 – (1- p_{i-1}/b)^a$. In the first stage there are 3 x 3 switches, therefore a=b=3. In second stage, the number of switches is half of the first stage; probability to reach second stage would be half. In last stage, switches are of 2X2 and the input lines are coming form first and second stages. The number of switches in third stage is double of the second stage and for requests coming from first stage the probability will be same.

5.3.2 Probability of acceptance

Ratio of the expected BW to the expected number of requests generated per cycle. This often used parameter in synchronous analysis is defined as the ratio of expected bandwidth to the expected number of requests generated per cycle [10]. In other words, it is probability that a request generated by a source should be successful in reaching the destination [20]. Thus, it is the ratio of requests matured to the total requests generated.

Probability of acceptance is the probability that, in a random-access environment, a request submitted by a source is accepted by a destination without getting blocked by other requests or connections in the network. This probability is usually evaluated by assuming that all the sources simultaneously generate their

requests for connection with a probability p and aim them at randomly chosen destinations, at the beginning of a cycle. These requests propagate through the network one stage at a time. When two or more requests arrive at a switch requiring the same output link, the requests that are serviced are chosen at random and the others are blocked and dropped. In the case of ABN and its variants proposed in this thesis i.e. MABN and IABN, a maximum of three requests can arrive at a switch. Up to two requests with the same routing tag bit can be serviced, by connecting one to the required output link to the next stage and the other to the auxiliary output link leading to the next switch in the loop. Destinations accept up to two requests per cycle, since they have two connections to the network. The probability of acceptance is defined as the ratio of the expected number of successful requests to the expected number of requests submitted by the sources.

Probability of Acceptance (P_a) = $BW / (a^n . p)$

5.3.3 Throughput

Average number of cells delivered by the network per unit time. Performance of the multiprocessor system depends largely on the latency and throughput [2] of the interconnection network of the processor and memory modules. Latency of the network is the time taken by a packet of information to traverse from an input port to an input port and is measured in cycles [2]; whereas throughput is the maximum traffic accepted by the network and is measured as packets per node per cycle. For non-uniform network traffic, an irregular network gives larger throughput than any regular network because of its smaller path length for favorable memory module. These benefits increase with the increase in the size of the network.

Throughput (TP) = $BW / (a^n . T)$
Where, T=0.285

5.3.4 Processor utilization

It is the expected percentage of time a processor is active doing internal computation without accessing the global memory [20].

Processor Utilization (PU) = $BW / (a^n .p. T)$

Where, T=0.285

5.4 Cost-effectiveness

We can observe that ABNs and its two variants as proposed in this thesis can provide higher or at least equal reliability compared to some other fault-tolerant networks. However, if such high reliability comes at the expense of high cost, it may have little value in practice. This section concerns the cost-effectiveness of ABNs.

To estimate the cost of a network, one common method is to calculate the switch complexity with an assumption that the cost of a switch is proportional to the number of gates involved, which is roughly proportional to the number of cross-points within a switch [3]. For example a 2 x 2 switch has four units of hardware cost, whereas a 3 x 3 switch has nine units. For the multiplexers and demultiplexers, we roughly assume that each of m x 1 multiplexers or 1 x m demultiplexers has m units of cost. Thus an ABN has the cost of $N/2(3\log_2 N+13)$ [4].

Realizing different sizes of switches increases the cost and complexity of an interconnection network in a VLSI environment. But this problem can be solved in ABN, by designing uniform switches (i.e. 3 x 3 switches) for all stages. At first, the switches in the final stage appear to be different in terms of the number of links and also because they are required to report faults in the demultiplexers at their outputs to the switches in the previous stage. However, designing all the switches in such a way that they report themselves to be faulty whenever their auxiliary link and at least one other output link leads to faulty switches, and then permanently assuming the auxiliary output links of the switches used in the final stage to the faulty stage, solves the uniformity problem.

Table 5.1: Cost Functions

MIN	**Cost**
ABN	$N/2(3\log_2 N+13)$
MABN	$N/2(5\log_2 N+9)$
IABN	$N/2(5\log_2 N+9)+9\log_2 N$

Now a simple measure of the cost-effectiveness for reliability can be given by comparing MTTF and the cost of the network. Let the cost-effectiveness, η of a network for reliability be the ratio of MTTF to its cost [4, 20].

Chapter 6
Experimental Results

In this chapter results of performance analysis are given section wise according to the main four performance measures that are permutation passibility, reliability, bandwidth and cost.

6.1 Results of Permutation Passibility

Permutation passibility is the measure which tells how many number of requests appearing at the source side has got successfully matured i.e. have reached the respective destinations successfully. Further both the cases have been considered when there is no faulty switch in the network and when there are faulty switches present in the network.

6.1.1 ABN

Results of permutation passibility analysis done in section 5.1 for existing regular network ABN are summarized as follows:

Total number of request appearing at source side =36
Total requests matured when no switch is failed =33
Total requests matured when switch is failed =24
Total path length when no switch is failed =73
Total path length when switch is failed =53
Average path length when no switch is failed =73/33 = 2.21
Average path length when switch is failed =53/24 = 2.208

6.1.2 MABN

Results of permutation passibility analysis done in section 5.1 for proposed regular network MABN are summarized as follows:

Total number of request appearing at source side =36
Total requests matured when no switch is failed =35
Total requests matured when switch is failed =33
Total path length when no switch is failed =80
Total path length when switch is failed =77
Average path length when no switch is failed =80/35 = 2.29
Average path length when switch is failed =77/33 = 2.33

6.1.3 IABN

Results of permutation passibility analysis done in section 5.1 for proposed irregular network IABN are summarized as follows:

Total number of request appearing at source side =36

Total requests matured when no switch is failed =36

Total requests matured when switch is failed =34

Total path length when no switch is failed =90

Total path length when switch is failed =95

Average path length when no switch is failed =90/36 = 2.5

Average path length when switch is failed =95/34 = 2.79

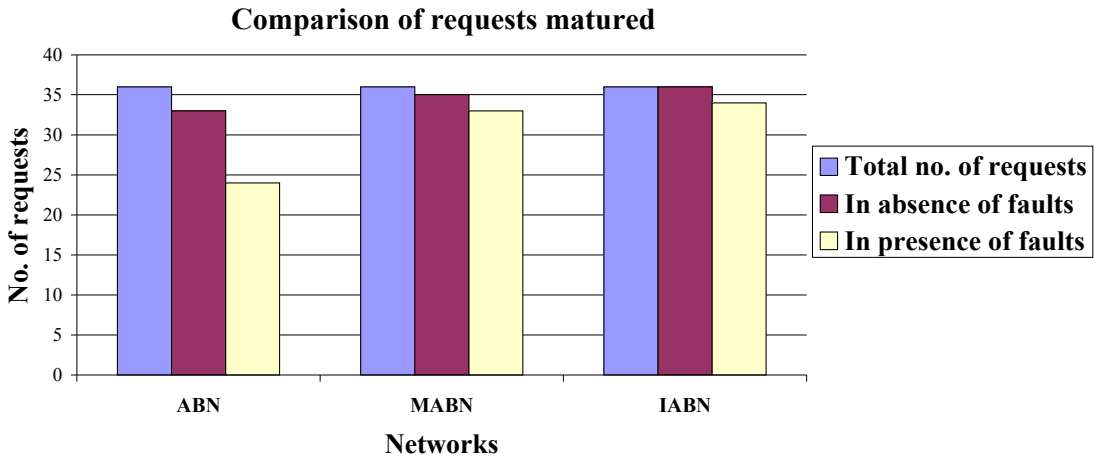


Figure 6.1: Comparison on the basis of number of requests matured.

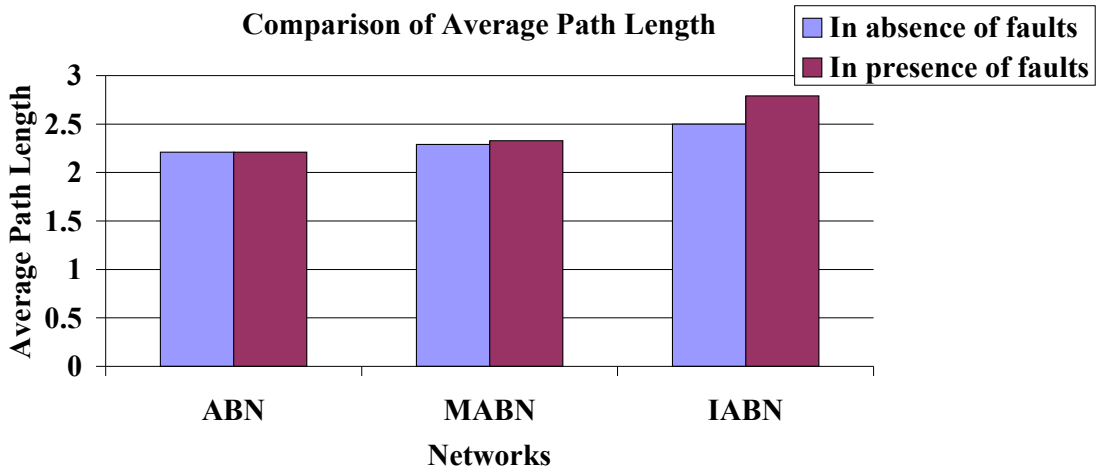


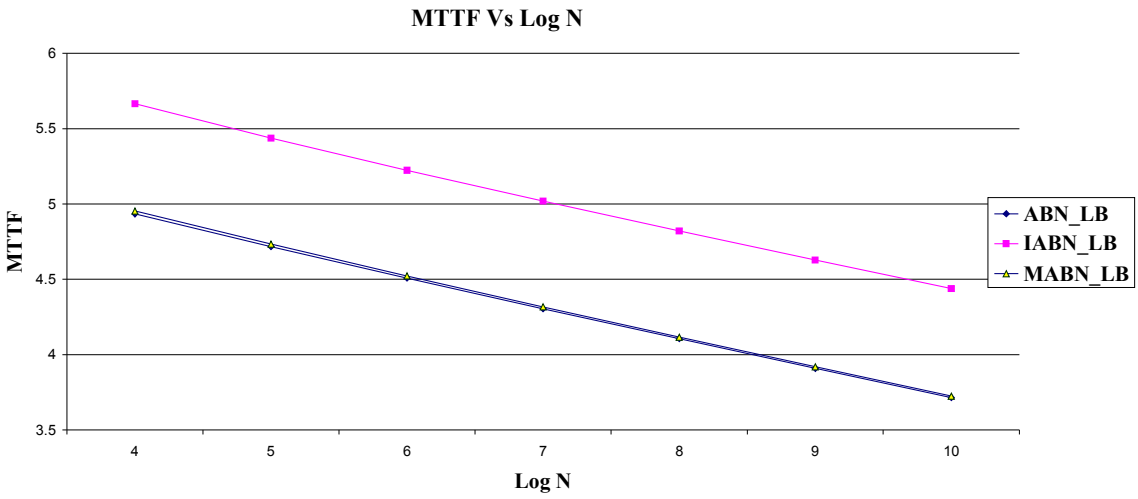
Figure 6.2: Comparison on the basis of average path length.

From the graphs shown above, it can be concluded that average path length of both the proposed networks (MABN and IABN) is greater than ABN. But there is

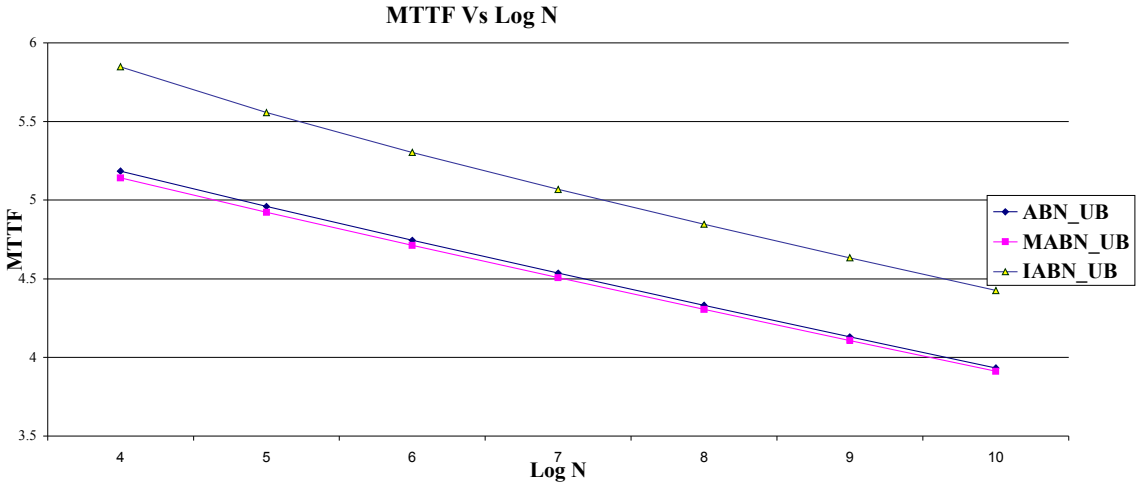
significant improvement in number of requests successfully maturing at the destination side especially in case of IABN, in both the cases i.e. in presence and absence of faults. Main consideration in permutation passibility is how many requests get matured in presence of faults, the proposed irregular network IABN gives best performance in this respect.

6.2 Results of reliability analysis

In this section values of MTTF i.e. Mean Time To Failure are given for three networks (ABN, MABN, IABN) for both lower and upper bounds. These MTTF values are obtained by putting values into the reliability equations derived in section 5.2. Both ABN and MABN has three reliability equations, whereas IABN has four equations due to addition of one extra stage in the middle of the network which has thus increased the reliability of the network.



(a)



(b)

Figure 6.3 Comparison of MTTF Vs Log N (a) Lower Bound (b) Upper Bound

It can be concluded from the graph shown above that both the proposed networks have more reliability then ABN. Increase in reliability values of MABN than ABN is very less, whereas for proposed irregular network IABN, increase is quite significant which is due to the addition of one stage in middle of the network along with auxiliary links which lets the two sub-network to communicate among themselves. It can be observed that the difference of values of IABN with respect to both ABN and MABN remains constant irrespective of the increase in the size of the network.

6.3 Results of bandwidth analysis

In this section values of Bandwidth are given for three networks (ABN, MABN, and IABN). These values are obtained by putting values into the probability equations derived in section 5.3. Further, three more performance parameters namely probability of acceptance, throughput and processor utilization, are calculated based on bandwidth values.

6.3.1 Bandwidth

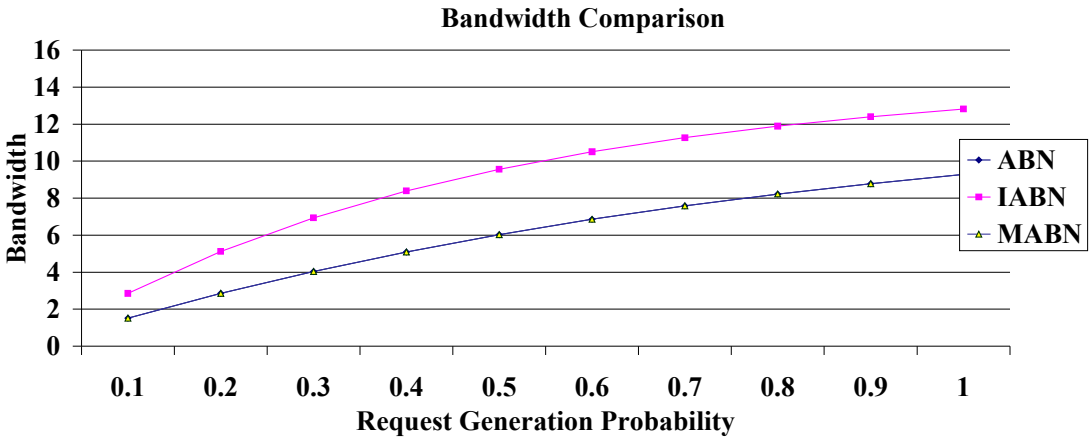


Figure 6.4: Comparative Bandwidth of ABN, MABN, IABN

As can be seen from the graph above, the bandwidth of ABN and MABN is same, since they have same probability equations. Bandwidth of IABN (Irregular Augmented Baseline Network) is almost double for low request generation probabilities (0.1, 0.2) and for higher generation probabilities also, the bandwidth values of IABN is much higher than that of ABN and MABN. Increase in bandwidth of IABN is due to the addition of one stage in the middle of network, which accepts input from both the first stage and middle stage according to routing tag.

6.3.2 Probability of Acceptance

Probability of Acceptance (P_a) = BW / (a^n . p)

BW = Bandwidth for (p=0.8)

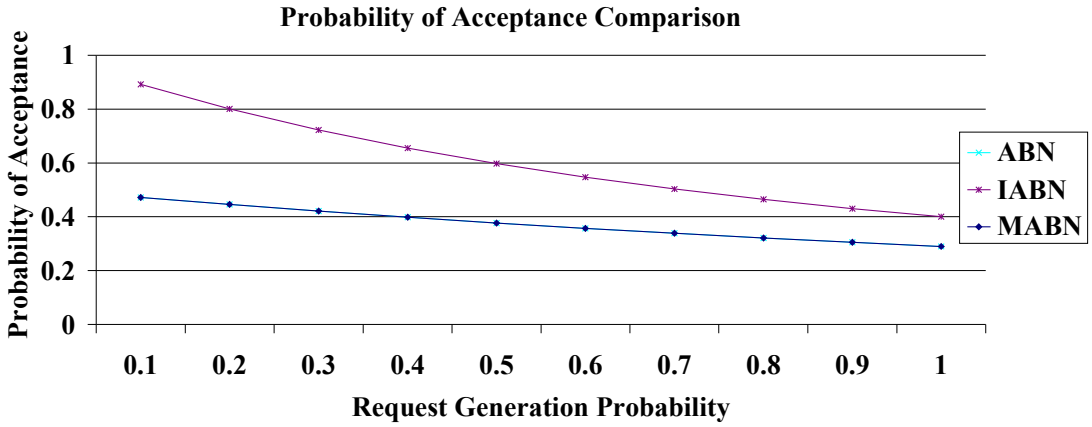


Figure 6.5: Comparative Probability of acceptance of ABN, MABN, IABN

As can be seen from the graph above, the Probability of Acceptance of ABN and MABN is same, since value of bandwidth is same for both the networks. Whereas in case if IABN, for low values of request generation probability (0.1 - 0.4), the value of probability of acceptance is almost double than ABN and MABN. But with increase in the value of request generation probability, the difference between the two values i.e. of ABN/MABN and IABN decreases.

6.3.3 Throughput

Throughput (TP) = BW / (a^n . T)

T=0.285, BW = Bandwidth for (p=0.8)

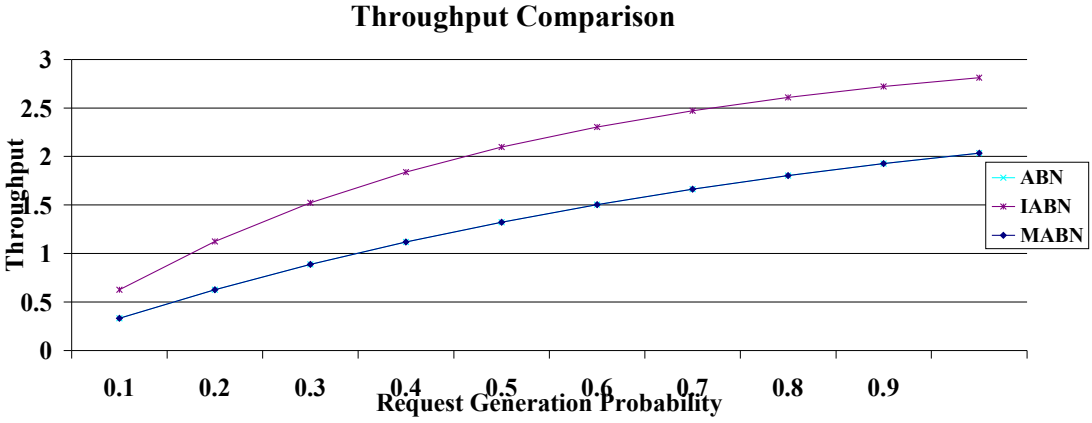


Figure 6.6: Comparative Throughput of ABN, MABN, IABN

Values of throughput is same for ABN and MABN, since for p=0.8 both has same bandwidth values. But throughput values for IABN is almost double than ABN/IABN. With the increase in the value of probability the increase in throughput values also increase. For larger values of probability, the gap between IABN and ABN/MABN is more as compared for smaller values of probability. For ease of understanding, graphical representation of the throughput values with respect to probability is shown.

6.3.4 Processor Utilization

Processor Utilization (PU) = BW / (a^n.p. T)

T=0.285, BW = Bandwidth for (p=0.8)

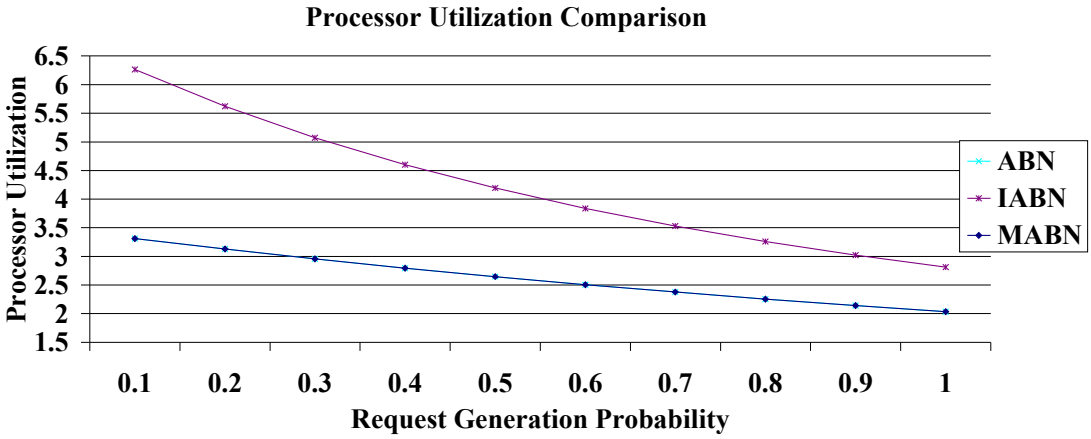


Figure 6.7: Comparative Processor Utilization of ABN, MABN, IABN

As can be seen from the graph above, the processor utilization of ABN and MABN is same, since value of bandwidth is same for both the networks. Whereas in case if IABN, for low values of request generation probability (0.1 - 0.4), the value of processor utilization is almost double than ABN and MABN. But with increase in the value of request generation probability, the difference between the two values i.e. of ABN/MABN and IABN decreases.

6.4 Results of cost analysis

In this section value of cost is given for three networks (ABN, MABN, and IABN) with respect to size of the network. These values are obtained by putting values into the cost functions derived in section 5.4 given in table 5.1.

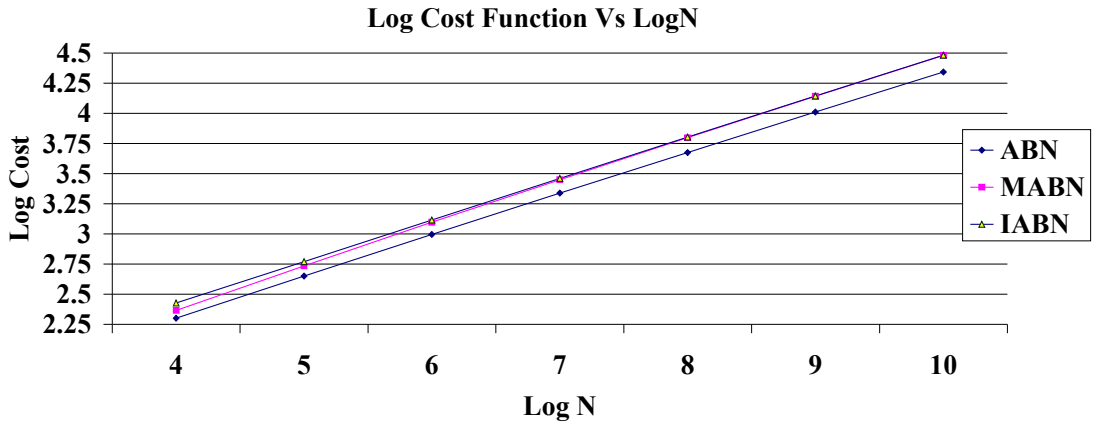


Figure 6.8: Comparative Log Cost Vs Log N of ABN, MABN, IABN

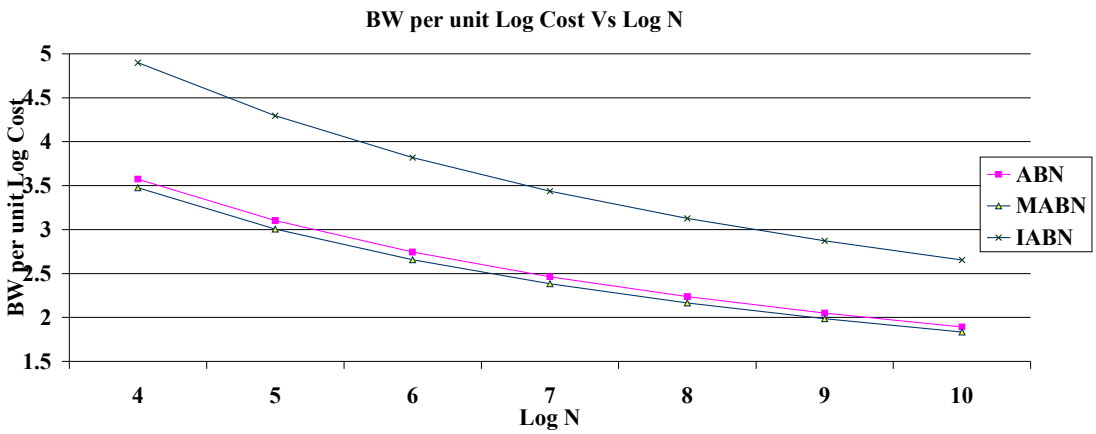
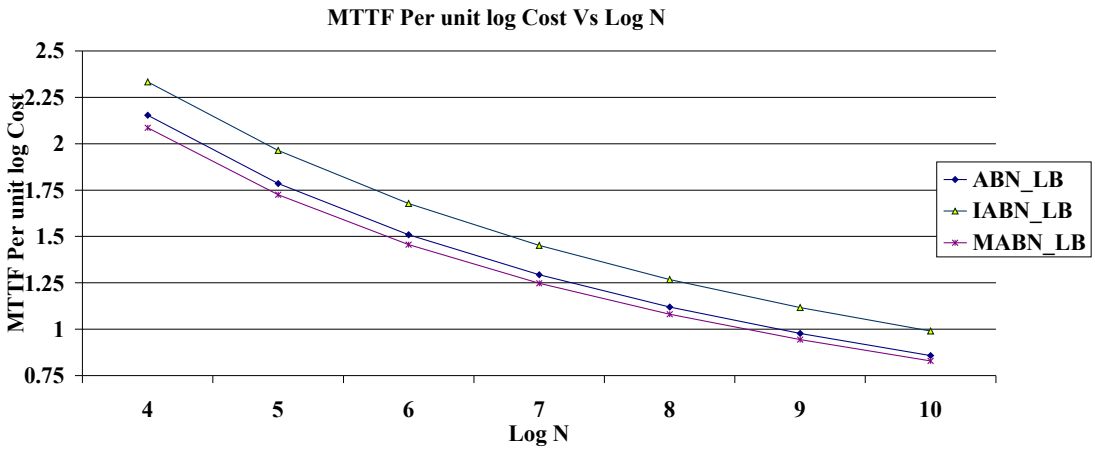
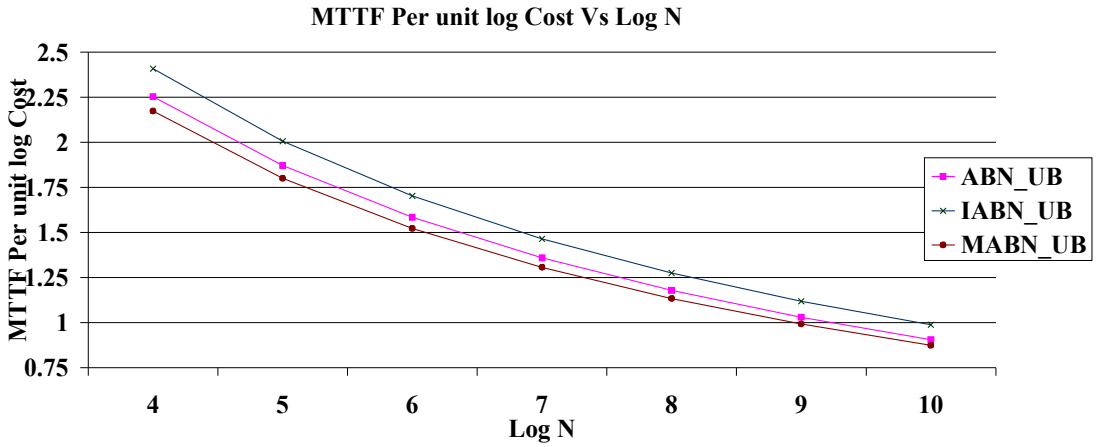


Figure 6.9: Comparative BW per unit Log Cost Vs Log N of ABN, MABN, IABN

To highlight the cost-effectiveness of ABNs, MABNs and IABNs (both upper and lower bounds) are evaluated and compared, and the improvement in results is shown. From the results we can observe that IABNs are more cost effective than most of the other fault-tolerant networks.



(a)



(b)

Figure 6.10 Comparison of MTTF Per unit log Cost Vs Log N
(a) Lower Bound (b) Upper Bound

(b)

CHAPTER 7

Conclusions and Future Scope

7.1 Conclusions

In this thesis regular and irregular networks are analyzed in terms of permutation passibility, reliability, bandwidth , probability of acceptance, throughput, processor utilization and cost. A detail of different multi-stage interconnection networks is provided.

Conclusions drawn from the work done in this thesis are summarized as:

Permutation passibility of proposed irregular network IABN is better than existing regular network ABN and proposed regular network MABN, in terms of lesser number of clashes. Whereas in terms of path length, the regular networks (ABN and MABN) have an edge over irregular network IABN. This is due to the addition of one more stage in the middle of the IABN, which thus increases the path length, but substantially reduces the number of clashes.

Reliability is analyzed in terms of MTTF (Mean Time To Failure) or ability of an network to perform even in the presence of faults, although at degraded performance. Reliability of IABN is quite high than ABN and MABN. Values of MTTF for ABN and MABN are comparable. Increased reliability in case of IABN is due to the presence of one extra stage in the middle of the network which connects two sub-networks to each other via auxiliary links.

Bandwidth has also improved for the proposed irregular network, IABN. Whereas, bandwidth of proposed MABN and ABN is comparable.

In addition to bandwidth, three more performance criteria's dependant on bandwidth namely Throughput, Processor Utilization and Probability of Acceptance has also substantially increased for IABN, as compared to ABN and MABN.

Costs of the two proposed networks (MABN and IABN) are calculated and cost-effectiveness, which is given by MTTF divided by cost, is analyzed.

The analysis of these three networks (ABN, MABN, and IABN), establishes that IABN is better in performance than the existing ABN network and proposed MABN. This is due to the reason that ABN and MABN are regular networks, whereas IABN is an irregular network.

7.2 Summary of contributions

The contributions made by the work presented in this thesis are summarized as:

Survey of existing regular and irregular multi-stage interconnection networks.

Two new networks MABN and IABN are proposed, which is designed by modifying the existing ABN network. MABN network has same number of stages and switches, whereas IABN has one stage more as compared to the ABN network.

Existing ABN and two proposed networks, MABN and IABN are analyzed for following performance measures:

Permutation Passibility.

Bandwidth (Throughput, Probability of acceptance and Processor Utilization).

Reliability (Upper Bound and Lower Bound).

Cost.

7.3 Future Scope

The field of irregular networks can be further explored in the light of the following suggestions:

A comparative analysis of the irregular networks with regular multi-stage interconnection networks can be carried out with respect to the other parameters.

The designing of more irregular networks having better reliability and permutation passibility can be explored.

The study can be extended to optical MINs.

REFRENCES

[1] Adams George B., Agrawal Dharma P. and Siegel Howard Jay, "A Survey and Comparison of Fault-Tolerant Interconnection Networks", IEEE Transactions on Computers, June 1987, pp. 14-27.

[2] Adams George B. and Siegel Howard Jay, "The Extra Stage Cube: A Fault-Tolerant Interconnection Network for Super systems", IEEE Transactions on Computers, vol. c-31, no. 5, May 1982, pp. 443-454.

[3] Bansal P.K, Singh Kuldeep and Joshi R.C., "Quad Tree: A Cost-Effective Fault-Tolerant Multistage Interconnection network**",** Proceeding of International Conference IEEE INFOCOM, 1992, pp. 6D.1.1-6D.1.7.

[4] Bansal P.K, Singh Kuldeep and Joshi R.C, " On Fault tolerant Multistage Interconnection Network", Conference on Computer Electrical Engineering, vol. 20, no.4, 1994, pp. 335-345.

[5] Bansal P.K, Singh Kuldeep and Joshi R.C, "Routing and path length algorithm for a cost-effective four-tree multistage interconnection network" International Journal of Electronics, vol. 73,no.1,1992, pp. 107-115

[6] Bhogavilli Suresh K. and Abu-Amara Hosame, "Design and Analysis of High Performance Multistage Interconnection Networks", IEEE Transactions on Computers, vol. 46, no. 1, January 1997, pp. 110 -117.

[7] Bhuyan Laxmi N., Yang Qing and Aggarwal P. Dharma, "Performance of Multiprocessor Interconnection Networks", Proceeding of IEEE, February 1989, pp. 25-37.

[8] Blaket James T. and Trivedi Kishor S., "Reliabilities of Two Fault-Tolerant Interconnection Networks", Proceeding of IEEE, 1988, pp. 300-305.

[9] Cam Hasan, "Rearrangeability of (2n − 1)-Stage Shuffle-Exchange Networks", Society for Industrial and Applied Mathematics, vol. 32, no. 3, 2003, pp. 557–585.

[10]Charles Chenggong Charles and Bruck Jehoshua, "Tolerating Multiple Faults in Multistage Interconnection Networks with Minimal Extra Stages", IEEE Transactions on Computers, vol. 49, no. 9, September 2000, pp. 998-1004.

[11]Chi Hsin-Chou and Wu Wen-Jen, "Routing Tree Construction for Interconnection Networks with Irregular Topologies", Proceedings of the

Eleventh Euromicro Conference on Parallel, Distributed and Network-Based Processing (Euro-PDP'03).

[12]Chuan-Lin Wu and Tse-Yun Feng, "The Reverse-Exchange Interconnection Network", IEEE Transactions on Computers, vol. c-29, no. 9, September 1980, pp. 801-811.

[13]Fan Chenggong Charles and Bruck Jehoshua , "Tolerating Multiple Faults in Multistage Interconnection Networks with Minimal Extra Stages", IEEE Transactions on Computers, vol. 49, no. 9, September 2000, pp. 998-1004.

[14]Kumar V.P. and Reddy S.M., "Augumented Shuffle Exchange Multistage Interconnection Network", Proceeding of International Conference IEEE, June 1987, pp.30-40.

[15]Malhotra Deepti and Aggarwal Rinkle," Performance Analysis of Fault Tolerant Irregular MINs", Proceedings of National Conference on Challenges and opportunities in Information Technologies (COIT-2007), RIMT-IET, March 2007, pp. 81-87.

[16]Mittal R., Cherian D. and Mohan P.J., "Routing and Performance of the double tree (DOT) network" Proceeding of International Conference on Computer Digital Technology, vol. 142,no. 2, March 1995, pp. 93-97.

[17]Nitin, "On Analytic Bounds of Regular and Irregular Fault-tolerant Multi-stage Interconnection Networks", Proceedings of International Conference, 2006.

[18]Kruskal Clyde P. and Snir Marc, "The Performance of Multistage Interconnection Networks for Multiprocessors", IEEE Transactions on Computers, vol. c-32, no. 12, December 1983, pp. 1091-1098.

[19]Li Shuo-Yen Robert and Tan Xuesong Jonathan, "Preservation of Conditionally Nonblocking Switches Under Two-Stage Interconnection", IEEE Transactions on Communications, vol. 55, no. 5, May 2007, pp. 973-980.

[20]Sadawarti Harsh and Bansal P.K., " Fault Tolerant Irregular Augmented Shuffle Network", Proceeding of the 2007 WSEAS International Conference on Computer Engineering and Applications, Australia, January 17-19,2007. pp. 7-12.

[21]Sengupta J. and Bansal P.K, "Performance of Regular and Irregular Dynamic MINs", Proceeding of International Conference IEEE TENCON, 1999, pp. 427-430.

[22]Sengupta J. and Bansal P.K, "Fault-Tolerant Routing in Irregular MINs", Proceeding of International Conference IEEE TENCON, 1998, pp. 638-641.

[23]Sengupta J., Bansal P.K and Gupta Ajay, "Permutation and Reliability measures of Regular and Irregular MINs", International Conference IEEE, 2000, pp. I-531-I-536.

[24]Sergio D'Angelo, Alderighi Monica, Fabio Casini, , Salvi Davide and Giacomo R. Sechi "A Fault-Tolerant FPGA-based Multi-Stage Interconnection Network for Space Applications", Proceedings of the First IEEE International Workshop on Electronic Design, Test and Applications (DELTA.02) IEEE ,2002.

[25]Sharma Sandeep and Bansal P.K., "A New Fault Tolerant Multistage Interconnection Network", Proceeding of International Conference IEEE TENCON, 2002, pp. 347-350.

[26]Sharma Sonia and Aggarwal Himanshu, "Performance Analysis Of Modified Fourtree Network", Proceedings of National Conference on Challenges & Opportunities in Information Technology (COIT-2007) RIMT-IET, March 2007, pp, 98-101.

[27]Ted H. Szymanski and V. Carl Hamacher, "On the Permutation Capability of Multistage Interconnection Networks", IEEE Transactions on Computers, vol. c-36, no. 7, July 1987, pp. 810-822.

[28]Wu Chuan-Lin and Feng Tse-Yun, "The Universality of the Shuffle-Exchange Network" IEEE Transactions on Computers, vol. c-30, no. 5, May 1981, pp. 324-332.

[29]Wu Chuan-Lin and Feng Tse-Yun, "On a Class of Multistage Interconnection Networks", IEEE Transactions on Computers, vol. c-29, no. 8, August 1980, pp. 694-702.

APPENDIX

Pseudo-code for permutation passibility of IABN

Global Parameters Passed:

(i) An array to store the network connections.

(ii) Two variables for storing the number of source destination pairs and the total number of faults.

(iii) An array for storing source and destination values.

(iv) An array for storing the faulty nodes.

(v) An array that stores the binary value of the destination.

(vi) An array for storing the nodes, traversed in the path from source to the destination.

(vii) An array that keeps track of all visited nodes.

Function Name: **main()**

Called By: **Operating System**.

Calling :

(i) enter_faults()

(ii) find_path()

(iii) display()

Parameters Passed: No.

Purpose: This function does all the necessary initializations.

Function Body:

(i) Initialize the graph matrix with value 1, if there is a path from a given source to the given destination. Otherwise, initialize with infinite value (indicating that no direct path exists between the corresponding source-destination pair).

(ii) Inputs the number of source-destination pairs from the user.

(iii) Inputs the value of source-destination pairs.

(iv) Inputs the number of faulty nodes.

(v) enter_faults() function is called for getting the faulty nodes.

(vi) find_path() function is called to find out the path between each source-destination pair.

(vii) Finally, display function is called to display the output in an understandable format.

Function Name: **enter_faults ()**

Called By: main function.

Calling: nothing.

Parameters Passed: Number of faulty nodes.

Purpose: To enter the value of faulty nodes from user.

Function Body:

(i) Get the value of faulty nodes, and store it in an array.

(ii) Return to main () function.

Function Name: **find_path()**

Called By: main function

Calling:

(i) dec_bin()

(ii) find_mux ()

(iii) find_mux1()

(iv) find_mux2()

(v) mux_to_stage1()

(vi) stage_1()

(vii) stage_2()

(viii) stage_3()

Parameters Passed: No

Purpose: To find the possible path for all source-destination pairs.

Function Body:

(i) Firstly converts the destination value in binary format by calling the function dec_bin().

(ii) Then calls the find_mux() function, that will return the primary MUX to be used for establishing the path between source-destination pair.

(iii) If the primary MUX found by the find_mux() function is busy or faulty, then call find_mux1() function which will return secondary MUX of the primary sub-network.

(iv) If the MUX found by the find_mux1() function is also busy or faulty, then find_mux2() function is called to find out the next alternative i.e primary MUX in secondary sub-network .

(v) Then call mux_to_stage1() function that will find out the switch after the MUX.

(vi) Then stage_1(), stage_2() and stage_3() are called for finding the next switches in the path, respectively for stage 1,2 and 3.

(vii) Return to the main() function.

Function Name: **dec_bin()**

Called By: find_path()

Calling: Nothing

Parameters Passed: Destination value.

Purpose: For decimal to binary conversion of the destination.

Function Body:

(i) Apply logic for the decimal to binary conversion.

(ii) Return the binary value to the function find_path().

Function Name: **find_mux()**

Called By: find_path()

Calling: chek_clash()

Parameters Passed: Value of source and destination.

Purpose: To find out primary MUX for establishing the path from source to destination.

Function Body:

(i) Find the appropriate MUX.

(ii) Then call the function chek_clash() for checking whether the MUX is already in use or not.

(iii) Return the MUX value to the calling function i.e. find_path(), if there's no clash else return -1.

Function Name: **find_mux1()**

Called By: find_path()

Calling: chek_clash()

Parameters Passed: Value of source.

Purpose: To find out the secondary MUX of primary sub-network.

Function Body:

(i) Find out the MUX.

(ii) Then call the function chek_clash() for checking whether the MUX is already in use or not.

(iii) Return the MUX value to the calling function i.e. find_path(), if there's no clash else return -1..

Function Name: **find_mux2()**

Called By: find_path()

Calling: chek_clash()

Parameters Passed: Value of source.

Purpose: To find out the next alternate MUX, i.e. primary MUX in secondary sub-network.

Function Body:

(i) Find out the MUX.

(ii) Then call the function chek_clash() for checking whether the MUX is already in use or not.

(iii) Return the MUX value to the calling function i.e. find_path(), if there's no clash else return -1..

Function Name: **mux_to_stage1()**

Called By: find_path()

Calling: chek_clash()

Parameters Passed: The MUX value that is used as new source.

Purpose: Find out the next switch in the path.

Function Body:

(i) Find out next switch connected with MUX.

(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.

(iii) Return the switch value to the function find_path().

Function Name: **stage_1()**

Called By: find_path()

Calling: chek_clash()

Parameters Passed: The next switch value.

Purpose: Find out the next switch in the path.

Function Body:

(i) Find out next switch.

(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.

(iii) Return the switch value to the function find_path().

Function Name: **stage_2()**

Called By: find_path()

Calling: chek_clash()

Parameters Passed: The next switch value.

Purpose: Find out the next switch in the path.

Function Body:

(i) Find out next switch.

(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.

(iii) Return the switch value to the function find_path().

Function Name: **stage_3()**

Called By: find_path()

Calling: chek_clash()

Parameters Passed: The next switch value.

Purpose: Find out the next switch in the path.

Function Body:

(i) Find out next switch.

(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.

(iii) Return the switch value to the function find_path().

Function Name: **chek_clash()**

Called By:

(i) find_mux()

(ii) find_mux1()

(iii) find_mux2()

(iv) mux_to_stage1()

(v) stage_1()

(vi) stage_2()

(vii) stage_3()

Calling: Nothing.

Parameters Passed: The switch or MUX value sent by the calling functions.

Purpose: Find out the next switch in the path is free or used by another request or faulty.

Function Body:

(i) Find out the switch or MUX is used by another request or free to use.

(ii) If free than it set visit is equal to 1.

(iii) If not free or faulty then update the value and find out the alternate path if available.

(iv) Return to the calling function.

Function Name: **display()**

Called By: main function ()

Calling: fprint ()

Parameters Passed: No.

Purpose: To display the output in the desired format and get the path length of each source, destination pair.

Function Body:

(i) Check the clash value if it is –1 then that request is not successfully matured and after displaying the path to that switch, print clash.

(ii) Else call the function fprint() and print the evaluated path.

Function Name: **fprint()**

Called By: display()

Calling: Nothing.

Parameters Passed: Value of the next node in the path.

Purpose: To display the switch number according to manipulated value.

Function Body:

(i) Display the values.

(ii) Return to the function display().

With some changes according to the networks same procedure is used for the MABN and ABN.

Pseudo-code for Calculating Bandwidth, Probability of Acceptance, Throughput and Processor Utilization.

Function Name: **main()**

Called By: Operating System

Calling: Nothing.

Parameters Passed: No.

Purpose: This function calculates the bandwidth, probability of acceptance, throughput and processor utilization.

Function Body:

(i) Evaluate the bandwidth of ABN network by applying the bandwidth equations.

(ii) Then calculate probability of acceptance, throughput and processor utilization using the calculated value of bandwidth.

(iii) Display the values of all these parameters.

(iv) Repeat the steps (i) to (iii) for the calculation of these parameters for MABN and IABN networks by using the bandwidth equations of these networks.

Pseudo-code for cost Evaluation

Function Name: main()

Called By: Operating System

Calling:

(i) ABN()

(ii) MABN()

(iii) IABN()

Parameters Passed: No.

Purpose: This function gets the choice and calls the appropriate function.

Function Body:

(i) Get the choice of network whose cost has to calculate.

(ii) According to the choice call the appropriate function either ABN() or MABN() or IABN() or all three.

Function Name: **ABN()**

Called By: main()

Calling: Nothing.

Parameters Passed: No.

Purpose: This function calculate the cost of ABN network

Function Body:

(i) Evaluate the cost of the network according to the formula.

(ii) Finally, display the evaluated cost.

Same procedure is used for calculating the cost of MABN and IABN.

Pseudo-code for Reliability calculation

Function Name: **main()**

Called By: Operating System

Calling:

(i) upper()

(ii) lower()

Parameters Passed: No.

Purpose: This function gets the choice and calls the appropriate function.

Function Body:

(i) Get the choice for calculating upper or lower bound or both.

(ii) Call the appropriate function.

Function Name: **upper()**

Called By: main()

Calling: Nothing

Parameters Passed: No.

Purpose: This function calculates the upper bound of IABN network.

Function Body:

(i) Evaluate the upper bound values according to the reliability equations.

(ii) Apply trapezoidal rule for calculating the upper bound MTTF.

(iii) Finally, display the evaluated MTTF.

Function Name: **lower()**

Called By: main()

Calling: Nothing.

Parameters Passed: No.

Purpose: This function calculates the lower bound of IABN network.

Function Body:

(i) Evaluate the lower bound values according to the reliability equations.

(ii) Apply trapezoidal rule for calculating the lower bound MTTF.

(iii) Finally, display the evaluated MTTF.

Same procedure is used for calculating the reliability of MABN and ABN.

LIST OF PUBLICATIONS

[1] Karamjit Kaur and Rinkle Aggarwal, "Designing Isomorphic Networks from the

Baseline", pp. 105-108 at National Conference on Advancements in Computer Engineering (ACE-08), held on April3-4 2008, in Fatehgarh Sahib (Punjab), India **[Published]**.