Deanship of Graduate Studies

Al-Quds University

An Aggregate Scalable Scheme for Expanding the Crossbar Switch Network; Design and Performance Analysis

Ahlam Qurei’

M.Sc. Thesis

Jerusalem- Palestine

2004

"An Aggregate Scalable Scheme for Expanding the Crossbar Switch Network; Design and Performance analysis"

By

Ahlam 'Muhammad Darweesh' Qurei'

B.Sc.**:** Computer Engineering, Palestine Polytechnic Institute, Hebron, Palestine

Supervisor: Dr. Abdulkarim Ayyad

A Thesis submitted in partial fulfillment of requirements for the degree of Master of Computer Science, Department of Computer Science, The M.Sc Program

Al-Quds University

July, 2004

The Master Degree Program / Department of Computer Science

Faculty of Science and Technology

Deanship of Graduate Studies

"An Aggregate Scalable Scheme for Expanding the Crossbar Switch Network: Design and Performance Analysis"

By

Student Name: Ahlam 'Muhammad Darweesh' Qurei'

Registration No.: 20020181

Supervisor: Dr. Abdulkarim AYYAD

Master thesis submitted and accepted, Date: 5th September, 2004

The Names and signatures for the examining committee members are as follows:

1- Dr. Abdulkarim AYYAD	Head of committee	Signature ……………..
2- Dr. Rashid JAYOUSI	Internal Examiner	Signature ……………..
3- Dr. Luai MALHIS	External Examiner	Signature ……………..

Al-Quds University

2004

Declaration:

I Certify that this thesis submitted for the degree of Master is the result of my own research, except where otherwise acknowledged, and that this thesis (or any part of the same) has not been submitted for higher degree to any other university or institution.

Signed …………………………………….

Ahlam ‘Muhammad Darweesh’ Qurei’

Date: ……………………………………..

Acknowledgement

I would like to extend my sincere thanks to the following people:

- My parents and my family for their relentless emotional and financial support.
- My supervisor for his guidance and valuable discussions.
- The Director of the Master degree program for his support and encouragement.
- The academic staff of the computer science department who attended the midterm presentations for their valuable comments.
- The internal and the external examiners for their rich discussion of the thesis and their valuable comments.
- All the friends and collogues who cared about the course of my study.

Abstract

New computer network topology, called Penta-S, is simulated. This network is built of cross bar switch modules. Each module connects 32 computer nodes. Each node has two ports, one connects the node to the crossbar switch module and the other connects the node to a correspondent client node in another module through a shuffle link. The performance of this network is simulated under various network sizes, packet lengths and loads. The results are compared with those obtained from Macramé project for Clos multistage interconnection network and 2D-Grid network. The throughput of Penta-S falls between the throughput of Clos and the throughput of 2D-Grid networks. The maximum throughput of Penta-S was obtained at packet length of 128 bytes. Also the throughput grows linearly with the network size. On the opposite of Clos and 2D-Grid networks, the per-node throughput of Penta-S improves as the network size grows. The per-packet latency proved to be better than that of Clos network for large packet lengths and high loads. Also the packet latency proved to be nearly constant against various loads. The cost-efficiency of Penta-S proved to be better than those of 2D-Grid and Clos networks for large number of nodes (>200 nodes in the case of 2D-Grid and >350 nodes in the case of Clos).On the opposite of other networks, the cost-efficiency of Penta-S grows as its size grows. So this topology suits large networks and high traffic loads.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The early computer networks like Ethernet, ring, star, and token ring have suffered from the data collision and / or the physical connection bottleneck [Hwa 93, Tan 96, Vir 97]. In recent years the demand for high speed switching has risen in conjunction with the increase in bandwidth available in all types of computer networks [Sven]. Local area networks (LANs) in many corporations have made the transition from 10 Mbps Ethernet to 100 Mbps Ethernet. This has resulted in a need for high bandwidth network switches with a large number of ports [Khan 01].

Communication network technology has evolved to offer the multimedia services including sound and video together with data transmission at the same time. This led to creation of ISDN (Integrated Service Digital Network) and growing up to BISDN (Broadband ISDN) networks. ATM (Asynchronous Transfer Mode) was introduced in mid 80's and has been selected by ITU-T (International Telecommunication Union-Telecommunication) as the most appropriate technology for BISDN [Khan 01].

Higher bandwidth, non-blocking network topologies such as Multistage Interconnection Network (MIN), Matrix switch [NI 01], KVM (Keyboard-Video-Mouse) switch [Andr 01], and crossbar switch had to be adopted. These topologies provide multiple paths from the inputs to the outputs of the network.

The basic function of a switch is to forward packets from any input port to any given output port. A generic switch architecture therefore consists of a number of port interfaces linked through same form of interconnect [Sven 03].

Crossbar based switching fabrics have been successfully used in multiprocessor system for a long time, but only recently have they been applied to LAN switches [Hwa 93, Wilk 96]. It has been proved that the crossbar topology has the least bottleneck problem among other topologies [Hwa 93, Wilk 96]. In the Banyan network (discussed in the next chapter), there is a contention which may occur in the case of requesting the same intermediate link by two sources in their attempt to access different destinations. This could be considered as an additional contention since the only expected contention in crossbar topology for a link exists only when more than one source try to access the same destination. The performance of a crossbar based interconnect therefore only depend on the collision rate in the crossbars, and that rate will be proportional to the link speed for any given link utilization [Khan 01].

In order to reduce the contention and the latency of the crossbar networks, buffers were introduced on the inputs and the outputs of the intermediate switches of multistage interconnection networks [Atiq 95, Zhou 96].

Building a large crossbar of discrete small crossbar units to accommodate large number of computers is complicated and costly process, especially if the control of the network is centralized. With the advent of VLSI it becomes feasible to build (32 X 32) crossbar router on a chip. STC-104 router is an example [Thomp 97].

1.1 Network Expansion and Scalability

For increasing demand on large network with high performance the network designers proposed and built large network topologies by expanding the existed topologies. **Expansion** means increasing the system by adding more modules or nodes to the network in order to obtain higher performance. **Scalability** is a term used to refer to a design that allows the system to be increased in size and in doing, so its performance is improved [Sten 88, Hwa 93]. **Modularity** refers to expanding the network by adding more building modules to the system without the need for redesigning the module or parts of the system.

As mentioned above, building large crossbars of discrete small crossbar units to accommodate large numbers of computers is complicated and costly process. The numbers of switches must be squared in order to double the number of computers served by a full crossbar complicated and costly process for large number of computers (~$10^{3)}$ computers. In 1953, Clos introduced a new topology for expanding the crossbar networks. This topology costs less and relatively simple. Small crossbar modules are arranged in a number of columns (Stages), and the outputs of each column are connected through a shuffle link to inputs of the next column. The outputs of the computer nodes are connected to the inputs of the first column and their inputs to the outputs of the last column [Yank 99]. The General name for this topology class is "Multistage Interconnection Network" (MIN). The size of network is a function of ($N/n \log_n N$) switches of (nxn) size. Expanding the MIN is less costly and complicated than the full crossbar. However, more rows and columns of switching modules have to be added.

Another expansion topology which is less scalable, but cheap and modular is the 2D-Grid network [Hass 98]. In this topology, the crossbar modules are arranged in a matrix. Part of the inputs and the outputs of the crossbar accommodate the computer nodes, and the other part is used to provide links with the four neighboring modules.

Another feature that must be accommodated is the **re-configurability**, it means the ability to face fault problems or to meet certain needs like making the network local to a group of computers, this feature must be achieved just by re-programming the nodes so that they avoid the faulty part of the network.

Motivated by the lower cost, scalability, and modularity measures, Ayyad from Al-Quds University, proposed, and mathematically analyzed a new expansion network topology for distributed shared memory multiprocessor systems called Penta-

S [Ayya 04] The results of his work showed that this expansion topology meets the criteria of scalability, modularity, simplicity, low cost (cost efficiency) and re-configurability. With the same motivations in mind, **the author proposed to apply the Penta-S expansion topology to computer networks**. As the time and resources available does not allow building such a network, the author proposed to simulate this network to see its feasibility for computer networks. The simulation includes the building block (crossbar switch module) and the expansion topology. **This represents the main goal of the work conducted in this project.**

1.2 The Penta-S Scheme

The building block of a computer network using this scheme is a full (n x n) crossbar switch module. No input or output buffers are used in this scheme. Each module can accommodate n nodes of a computer. An n-shuffle connection is used to connect up to (n+1) of theses modules. For example, if 32 x 32 size crossbar modules are used as building block, then up to 1056 [(32 x 32) + 32] nodes can be connected using this scheme. Each nodes in the network is provided with two ports; one to connect it to its crossbar module called crossbar port, and the other to connect it to a node of another module in the scheme via the shuffle (shuffle port). Detailed description of this network is presented in chapter 3.

1.3 Goals and Objectives

The **main goals** of this work are:

1. To check the feasibility of Penta-S topology as a computer network and to define its position among other computer network topologies.

2. To find out what features of the modern crossbar switches that suit this topology (Features of a modern crossbar switch called STC104 are discussed in chapter 4).

The **main objectives** of this work are

1. Investigate the performance of STC104 switch and how its performance is related to its features. Also to investigate which of its features (originally designed to be a building block in packet switching networks) enhances or degrades the performance of Penta-S topology.
2. To build a network simulator which is capable of testing the performance in the above two point, modifiable for future work.
3. To test the performance parameters (throughput, packet latency and cost efficiency) of the Penta-S topology as a computer network under various sizes of network, packet lengths, and loads.
4. To compare the results obtained from point 2 with other well known topologies.

1.4 Contributions

The author believes that this project has the following contributions

1. Testing the performance and the feasibility of Penta-S topology as a computer network as compared with other well established and known computer network topologies.
2. A lay out of the computer node interface to Penta-S is accomplished.
3. As a result of investigating the STC104 features against Penta-S needs, a non-buffered switch with wormhole routing capability that suits the Penta-S expansion topology is proposed and its design is laid out.

4. A simulator for future studies has been designed and built. The results obtained from this simulator compared with Macramé project results. The simulator design is presented in chapter 5, and a UML model of the simulator program is presented in Appendix-F

1.5 Methodology

1. Literature survey: an in-depth study of the computer networks, their parameters, features and concepts was done before conducting the literature survey. The literature survey concentrated on the efforts paid to improve the network performance through modifications on the topologies of the network. The aim of the literature search was to find a practical, explicitly presented project using the largest crossbar switch module as a building block in well known and tested topologies like MIN and 2D-Grid. This is to facilitate the comparison of Penta-S with well known and widely used computer networks. Macramé project [Haas 98] was the best to suit this purpose.
2. A detailed study on STC104 and Macramé project was conducted. This resulted in a proposed simpler switch by the author.
3. The simulator was built taking into consideration that it is for new topology. Two types of building blocks were considered in the simulation process, namely, the no-buffered STC104, and the switch proposed in this project.
4. The simulator was run under various values of network parameters (network size, packet length, applied load, and switching delay) to test the throughput and the latency in order to see their effect on the network performance.
5. Results were collected and compared with results from well established networks like Clos and 2D-grid of Macramé project.

6. Conclusion on the whole work was drawn and recommendations for future work were stated.

1.6 Thesis Outline

Following this chapter, **Chapter 2** presents the main concepts of the switching computer networks followed by a literature search and a review of the switching networks.

Chapter 3 presents the Penta-S and reports on the networks that were used as test-bed in Macramé project STC104.

Chapter 4 The STC104 packet routing switch is studied and its suitability to Penta-S topology is discussed. The STC104 without its "input and output buffer" is simulated as a single module to see the buffer effect on its performance. A Non-Buffered-Wormhole-Routing (NBWR) switch is proposed as a building block of Penta-S.

Chapter 5 discusses the design of the simulator; building a prototype of this topology is costly and time consuming. A simulator was designed to simulate this network depending on its topology and the features of the switch used as a building block.

Chapter 6 presents the results of the simulation. To see where this topology stands, the results obtained from the simulator were compared with those obtained from Macramé project. In Macramé' project two tested networks were built using STC104 as a building block. The first network used Clos Multistage Interconnection Network topology to accommodate 512 nodes. The second used the 2D-Grid and its Tours version to accommodate 1024 nodes. Also the comparison included the throughput and latency versus various packet sizes, loads, and number of active nodes. Cost-Efficiency also was compared for the three topologies.

Chapter 7 includes the conclusion of the whole work and the suggested future work.

Chapter 2

Switching Networks

2.1 Introduction

In this chapter, the main concepts of switching networks are introduced, and a literature survey of research in this field is conducted. The literature survey concentrated on the developments in switching network topologies and performance. The term ‘Topology’ means the physical interconnection structure of the network elements, namely, the terminal nodes, the switches and links. Performance, basically, means the amount of data the network can transport in time unit.

The basic building block of switching network is the crossbar switch. The smallest switch size can connect two inputs to two outputs. It is referred to as 2X2 switch. Before the age of VLSI (Very Large Scale Integration) technology, this size was the main building block of switching networks (larger crossbar switches and multistage interconnection networks). With the advent of semiconductor technology, larger crossbar switch sizes started to be manufactured and used as building blocks in larger networks. As the crossbar has the best performance among other topologies, the larger the switch size, the better the network performance will be. The largest switch known so far is the STC104 from SGS-Thomson Microelectronics with size 32X32. As this size is used in Penta-S topology, while doing the literature search, it was kept in mind that we need a work in which large switching networks were built and tested rather than simulated. The details of such work must be clearly stated. This is to facilitate the comparison between Penta-S and other topologies in order to define the exact position of this topology among other switching network topologies. Two projects that use STC104 as a building block were found. The First used STC104 switch to connect T9000 transputers in order to test the idea of indirectly connecting a number of transputers [Heel 96]. The second project [Haas 98] also came from Liverpool University. In this project a 512 node Clos and a 1024 node 2D-Grid networks were built and tested. The two networks represent the two main classes of the switching networks, the direct network class is represented by the 2D-Grid, and the indirect network class is represented by Clos multistage interconnection network. So these two networks of the project were adopted for comparison with Penta-S network. The concepts defined in this project and the test terms and the testing approach were used in simulating the Penta-S networks. In The following section the basic concepts of topology and performance are explained and in the section after, a brief of the literature survey is given with concluding comments.

2.2 Basic Concepts of Switching Networks

2.2.1 Switch Strategy

Switching strategy determines how the data in a message traverses its route through the network. There are basically two switching strategies, circuit switching and packet switching. In circuit switching the path from source to the destination must be established and reserved first, before the message is transferred over the circuit. Subsequently the connection is then removed. The alternative is packet switching in which the end nodes send messages by breaking them up into a sequence of packets, which are individually routed from the source to the destination. Each packet contains route information examined by switching elements to forward it correctly to its destination; packet switching typically allows better utilization of network resources. A combination of the two strategies will be used in this thesis. The packet will be used as a data transfer unit to be sent through the network in two stages. The packet includes routing information which leads its path in the two stages. However in each stage it behaves like circuit switching, i.e., it reserves the path before transmitting the packet body. The packet resides in an intermediate buffer before continuing the second stage. The whole process looks like a packet wormhole routing process (the concept of wormhole routing is explained in chapter 4). The switching strategy draws the major lines of the network topology.

2.2.2 The Topology

The network topology decides the physical structure of the way the nodes, the links and the switches are organized and connected. The basic building block of these networks is the crossbar switch. The crossbar is a non-blocking switch. Non-blocking means that any permutation of inputs and outputs, can be connected without interfering with each other. However, not more than one input port can be connected to the same output at the same time. Usually the number of output ports is equal to the number of input ports. Figure 2-1 shows a schematic of the crossbar connection.

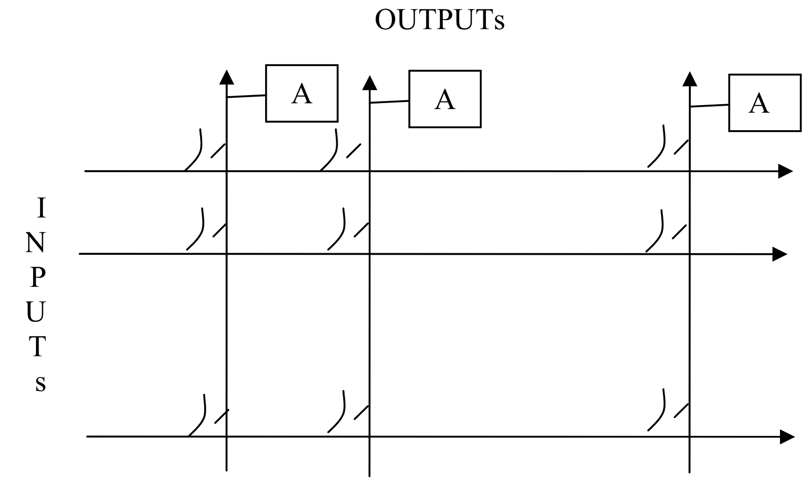


Figure 2-1: The crossbar switch

In figure 2-1 any input can be connected to any output by closing the switch on the cross-point between the input and the output. If more than one input is trying to connect itself to the same output, the arbiter (A) of the output selects one input and rejects the others. So, not more than one input can be connected to the same output. Expanding the crossbar switch to accommodate more inputs and outputs proved to be costly and complicated [HWA 93, Wilk 96]. Doubling the number of inputs and outputs implies that the cost grows four times. In other words the cost and complexity of the crossbar grows as a function of N^2, where N is the number of inputs which is usually equals the number of outputs. So other less cost topologies are needed.

2.2.2.1 Expansion Topologies

The expansion topologies of the crossbar switch falls into two main classes [Haas 98]:

1. The Direct Networks: In these networks the nodes are directly connected to the switches of the network. The node means the computer and its interface to the network. Each switch and its nodes represent a module in the network. Modules are arranged in a matrix fashion and enough links are available between each module and its four neighbors. Example is the 2D-Grid and Torus networks.
2. The Indirect Networks: In this network, the nodes are connected to the switches laying on the network edges. Switches lay in the middle of the network connect the edge switches. Examples of this class are the matrix switch [NI 01], and the multistage interconnection networks (MINS).

a. The Matrix Switch

The main module of matrix switch is an array of crossbar switches of the 4X4 size. This switch can be expanded by placing them in an array, connecting the internal edges, (the output of one to the input of the next) and connecting the nodes to two of the four external edges of the array [NI 01]. This expansion method is as costly as the crossbar switch expansion especially for networks with large number of nodes.

b. The 2D-Grid Network

In this network the nodes are directly connected to their switch modules. The modules are arranged in a matrix fashion. Enough links are provided between each module and its four neighbors. However, not all links of the switch module are used for connecting the nodes. Part of the switch module links are used to provide connections with the neighboring modules. Figure 2-2 shows how the 2D-Grid is implemented in Macramé project [Haas 98].

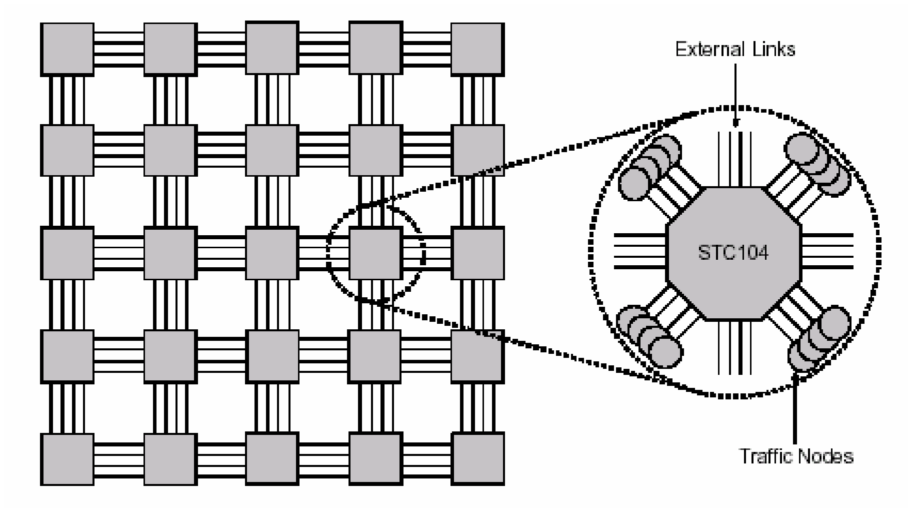


Figure 2-2: The 2D-Grid expansion topology

From the above figure, it is clear that half the 32 links of the STC104 module are used to accommodate 16 nodes, and the other half is used to provide 16 connection with the four neighboring modules, four links for each.

c. The Multistage Interconnection Networks (MIN)

In 1953, Charles Clos published a technical paper describing a method to reduce the cross-points count through the multistage switching technique [Yang 99, HWA 93, Wilk 96]. Since that time the age of the MIN started. Different connection fabrics and different switch sizes were proposed and implemented since then.

In this topology the switching modules are arranged in columns. The outputs of each column are connected to the inputs of the next in a shuffle fashion to provide a non-blocking connection between the inputs of the first column and the outputs of the last column. Figure 2-3 shows a 2-stage MIN built of 4X4 switch module, which connect 16 inputs to 16 outputs.

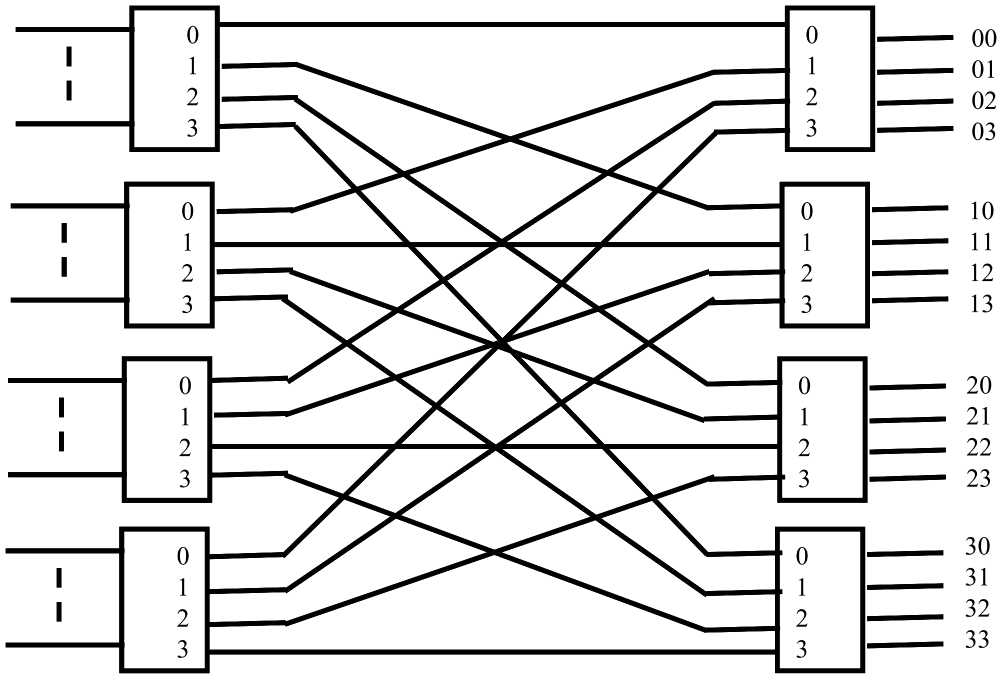


Figure 2-3: A 2-stage MIN

The cost of the network in this expansion topology is expressed in terms of the number of switch modules (K) by the following equation:

$$K = (N/n) \text{ Log}_{n} N$$

Where N is the number of nodes (equals the number of inputs = the number of outputs) in the network and n refers to nxn switch module size. (N/n) equals the number of switch modules in each column whereas (Log_{n} N) represents the number of columns (stages). In the network shown in figure 2-3, the number of switches in each column = 16/4 = 4, and the number of columns = Log_{4} 16 = 2. So, the total number of switches = 4 X 2 = 8 switch modules. To double the number of nodes in this network, we need 3 columns each contains 8 switches, i.e., 24 switches are needed. If an equivalent crossbar is to double the number of nodes it needs $(32/4)^2$= 64 switches of the 4X4 size. This shows the difference in the expansion cost between the full crossbar and the MIN networks.

Varieties of MINs can be found in the literature. The most two used MIN network topologies are shown in the following:

Clos Network is a MIN that allows the usage of various sizes of switches, with general nxm switch. This topology provides the network designers with flexibility. Figure 2-4 show a Clos network with nxm switch size and r stages.

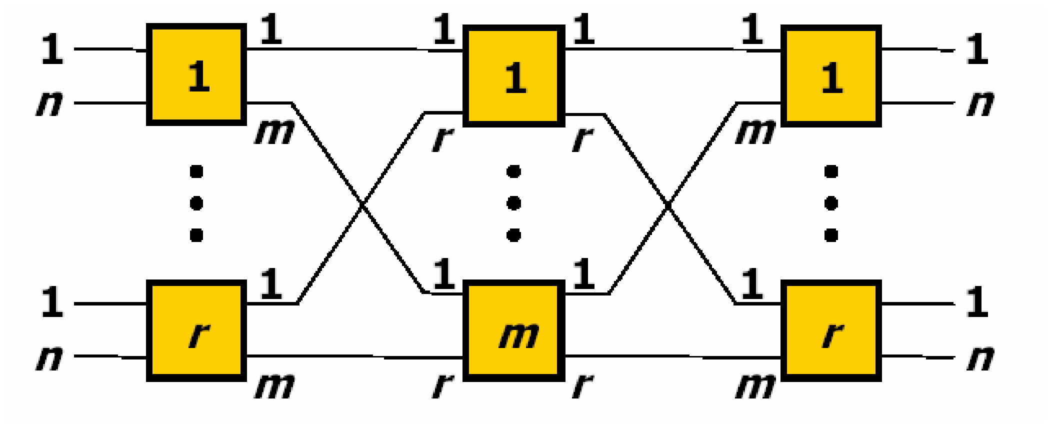


Figure 2-4: Clos network u(n,m,r) network

Banyan Network is an expansion topology which uses smaller n-shuffled MIN networks like in figure 2-3 as building block for larger MIN. However, uniform switch size is used in all network stages with each switch having the same inputs and outputs. An extra stage is needed which uses an L-shuffle where L is the number of inputs of the building block. Figure 2-5-a shows Banyan network that connects a number of 2-shuffle MINs. This type of network is adopted for ATM networks. From its architecture it is clear that a failure of a block would not affect the rest of the network. This means that this topology is a fault tolerant one. Figure 2-5-b shows a 4-shuffle MIN which can be used as a building block for a larger Banyan network.

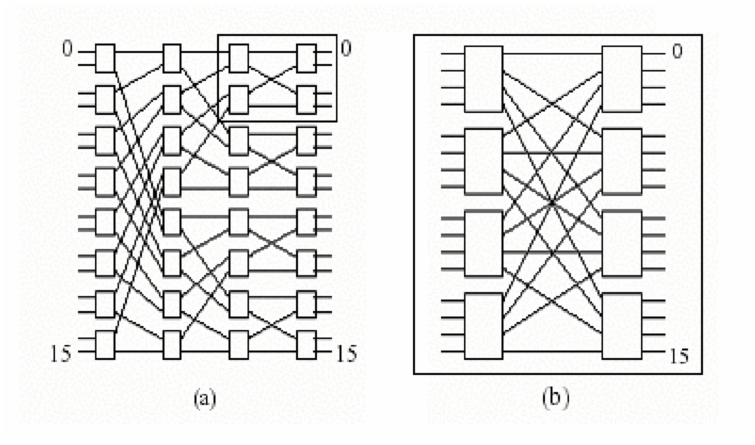


Figure 2-5: Banyan networks [Shi 01]

In MINs, when more than one packet arrives at the switch inputs destined for the same output, then some **arbitration** and **queuing** mechanisms has to be implemented in the switch. There are two possible architectures:

1. Input Buffers. In this configuration the buffers are located at the input ports of the switch. Arbitration logic is needed to determine which of the packets held in different input buffers destined to the same output will be transferred through the interconnection matrix. The arbitration logic can be very simple, e.g. First In First Serve (FIFS). Input buffered switches are the easiest to implement, as the buffers and the switching fabric only need to run at the speed of the ports. In the proposed scheme the input buffer is added inside each PC(node).

2. Output Buffer. With this architecture, the buffers are located at the output ports of the switch element. The assumption is that more than one packet from the input ports can cross the internal interconnection matrix and be stored in the same output port buffer. This solution requires the use of a very fast internal cross-connect. In order to allow a non-blocking switch, the interconnection network and the output buffer have to be capable of handling N packets simultaneously, where N is number of switch ports.

2.2.3 Network performance

In this section the measures of network performance are defined. The performance of a switching network can be characterized by throughput and latency. These definitions and concept was agreed with those used by Macramé project after consulting the Webopedia [Webo].

2.2.3.1 Throughput

The per-node throughput is the number of data bytes received or transmitted by an end-node in a given period of time. The header and end-of-packet characters are not counted. The total network throughput is the aggregate transmit or receive rate of all nodes.

Note: At some point the network saturates and an increase in network load does not yield any additional increase in the delivered throughput. The maximum achieved throughput is also called saturation throughput.

2.2.3.2 Latency

The network latency is defined as the delay between the head of the packet entering the network at the source to the packet being completely received at the destination. The latency is the sum of the three components:

- Switching latency;
- Transmission latency;
- Queuing latency;

The switching latency is proportional to the number of switches a packet has to traverse: In proposed scheme each packet can be traversing a maximum of 2 switches.

These cases occurred when the destination address is global, means the destination node belongs to another switch module

The transmission latency is the time to transmit a full packet, including the overheads of the packet (packet header and end-of-packet). The transmission latency is proportional to the packet length; it is also a function of the link speed and the packet overheads.

The queuing latency is the additional delay a packet experiences, because it is blocked and must be queued waiting for the selected resource to become available. It is a function of the number of switches a packet has to traverse, of the network load and the traffic pattern. As the network load increases, more and more packets get blocked due to contention within the network and the queuing delay increases. A network can provide low latency communication when the requested bandwidth is well below that which can be delivered. However, at some point the network saturates and additional load causes the latency to increase sharply without yielding additional delivered bandwidth.

2.2.4 Traffic Patterns

The performance of a switching network will also be a function of the traffic pattern used. The following traffic parameters influence the network latency and throughput.

- Load;
- Packet length;
- Packet destination;

The applied network load is defined as a percentage of the theoretical maximum throughput of the network. The traffic patterns used for the measurements in this project is the **Uniform random traffic**. This traffic pattern provides all-to-all communication between the end-nodes. Each node sends fixed length packets to random destinations. The destinations are chosen from a uniform distribution. The special case of a node sending to itself is excluded. The time between sending packets is taken from a Poisson distribution.(This distribution is illustrated in chapter simulation).

2.2.5 Network Routing

The routing algorithm determines which routes the packets follow through the network. In this thesis are presented the following routing algorithms.

1. Wormhole Routing or cut-though routing: At the beginning each switch examines the header, decides where to send the message, and then starts forwarding it immediately.
2. Store and forward: Each switch waits for the full packet to arrive in the switch before it is sent on to the next switch.

In Penta-S first will send the header of packet, this header used to establish the connection between the sender and receiver, after that the line is reserved and the packet sends between them.

2.3 Literature Review

The majority of the papers studied in this research, concentrate on the topology, routing algorithms, buffering and blocking problems. Most of the papers encountered in this research use the simulation as a tool of analysis. The only practical project which was interesting to compare with Penta-S topology was the Macramé project

[Haas 98], which was studied and analyzed in details. This is because this project uses large crossbar modules for building large networks. The networks which were built in Macramé project represent the two main classes of switching networks, namely, the direct class represented by the 2D-Grid network and the indirect class represented by Clos network. The thesis of this project included fine details on building the project, its parameters, and the conditions under which they ran their experiments. This enabled us to use the same terms and conditions in our simulation in order to have a fair comparison with the networks presented in Macramé project.

Through the literature search, we found that most research papers are interested in improving the performance of the existing topologies. This includes the bandwidth of the network and finding the best non-blocking conditions of the network. Most of the papers addressed MIN topologies like Banyan and Clos network. Only one research paper has addressed the alternation of the n-cube topology [Mud 01]. In this paper, rather than placing the nodes on the corners of the cubes, the authors suggested putting them on the middle of the cube edges. This increases the number of links between each node and its neighboring nodes. Rather than being connected to other two nodes in the cube it is connected other four nodes in the dual of the cube (suggested alternation). This increases the bandwidth of the network.

Parameter like the link speed, the size of the switch (used as a building block of the network), the size of the input and output buffers (in the buffered MIN), the offered load and the routing algorithms are of the prime importance to the performance of the network and the blocking problem. Some papers like [Tut 02] investigated the effect of buffer size and the switch size on Banyan network in the case of multicast traffic. A model is built to find the optimal network parameters for a given traffic pattern. Other study [Shi 01] concentrated on finding a trade-off between link capacity and the size of switching element on one hand and the blocking probability on the other hand. The study concluded that, in contrast to what is generally believed, larger switching elements are often, but not always the best design choice.

Two studies [Yang 99, Collier 94] investigated the non-blocking condition of Clos network Yang investigated the non-blocking conditions of Clos network under several routing strategies in terms of the number of middle switches. Collier has put a criterion for Clos non-blocking conditions which distinguishes between the channel grouping and link-speed up as methods of increasing the bandwidth available to calls.

The hot-spot and the back-pressure in MIN is investigated in [Sapo 03]. This work shows that the buffers near the inputs of the network must be larger than in the middle and final stages of the network.

One can conclude that the current efforts are concentrating so far on improving the performance of the current network topologies. Nothing much is done about creating new topologies. This justifies the efforts paid in this research to evaluate the newly proposed Penta-S network topology.

Chapter 3

The Penta-S Scheme and Macramé Networks

3.1 Introduction

In this chapter, the Penta-S scheme is presented. The networks which were tested in Macramé project (Clos and 2D-Grid networks) are introduced. This is because the performance results and cost of Penta-S scheme will be compared with the results obtained from these networks. These networks were selected for comparison with Penta-S scheme because, to the best of the author's knowledge, these are the only networks with large number of nodes (1024 nodes) that were built and tested for research purposes.

3.2 The Penta-S Scheme

3.2.1 The Topology

The building block in this scheme is a full (n x n) crossbar module. The module used here are non-buffered STC104 or a proposed non-buffered wormhole routing switch (NBWR) with a simpler and faster control circuitry. The architecture and functions of these switches are presented in chapter 4. Each switch can connect n computers, where n has a maximum value of 32. Each computer in the network is provided with two ports, one to connect it to the crossbar and the other to connect it to the rest of the modules (blocks) of the scheme via the shuffle link of the scheme. Each port of the node has an input and an output. Figure 3-1 shows a block diagram of the scheme.

The computer, its buffers, bypasses and it interface to the switch and to the shuffle is called a node. Figure 3-2 shows a schematic of the node.

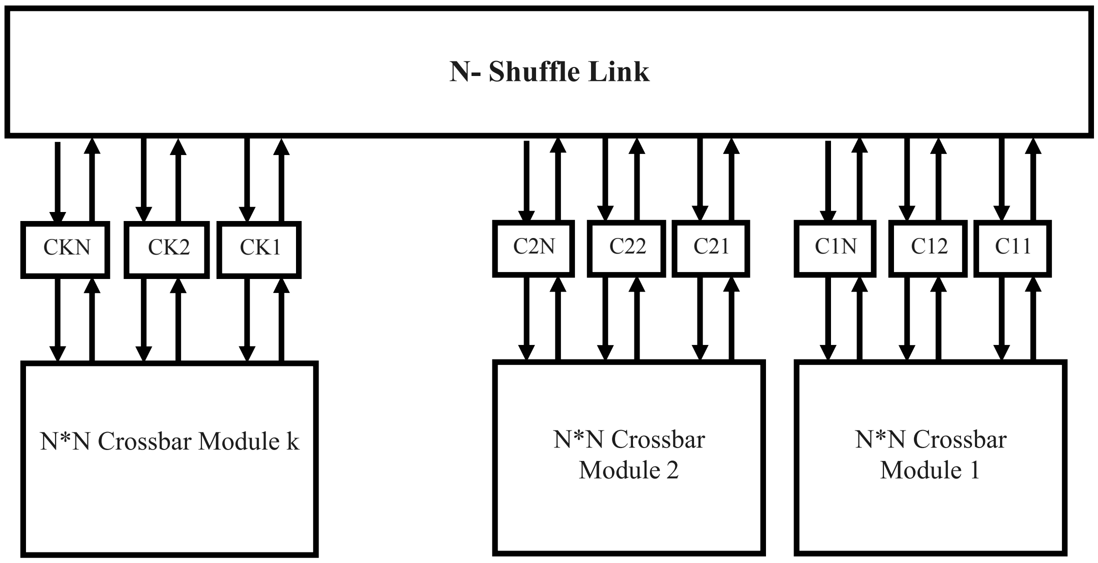


Figure 3-1: A block diagram of the Penta-S scheme

The port that connects the computer to the crossbar (called the crossbar port) has a bypass control unit to pass the information to the output of the Shuffle port if the information belongs to a computer in another crossbar module. The Shuffle port connects the computer to other computers in other crossbar modules via an n-Shuffle connection. The n-shuffle connection can connect up to (n+1) crossbar modules, where each module accommodates n computers. For example, if 32*32 crossbar modules are used as building blocks, then up to 1056 [(32 x 32) + 32] computers can be connected using this scheme. This topology forms a **S**ingle **S**tage, **S**ingle **S**huffle, and **S**calable system. ***The initials of these features are five 'S'es. So the scheme was given the name "Penta-S'.*** The n-Shuffle provides a direct link between each computer in the crossbar module and its correspondent in another module. Figure 3-3 shows a 4-shuffle which is capable of connecting 20 sources to 20 destinations.

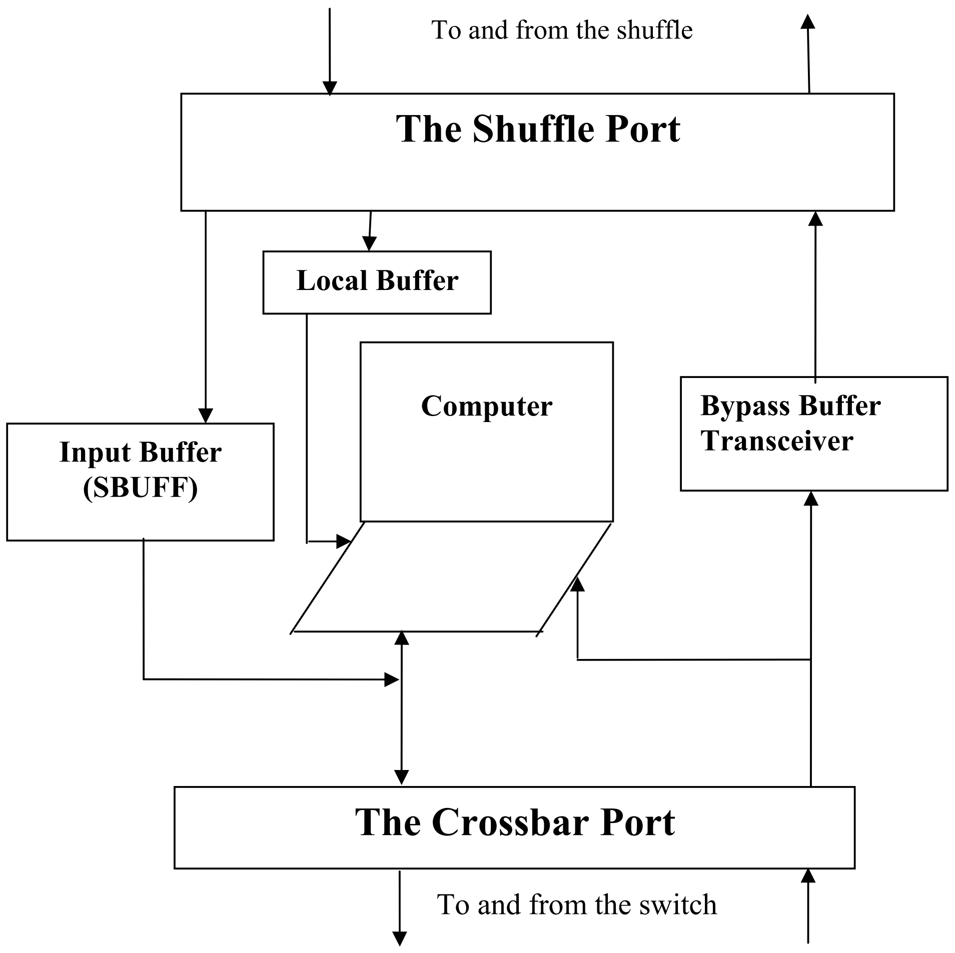


Figure 3-2: The schematic of the network node

Each link of the shuffle connects the output of the shuffle port of a local client node to the input of the shuffle port of its corresponding client in another module (block) of the scheme. For example, take the module ‘i’ and ‘j’ of the scheme. ‘Node ij’ of module ‘i’ is responsible for sending the packet produced by any node in module ‘i’ and its destination is a node in module ‘j’. Also, it is responsible for receiving any packet coming from module ’j’ sent to a destination node in module ‘i’. The corresponding client in module ‘j’ is the ‘node ji’.

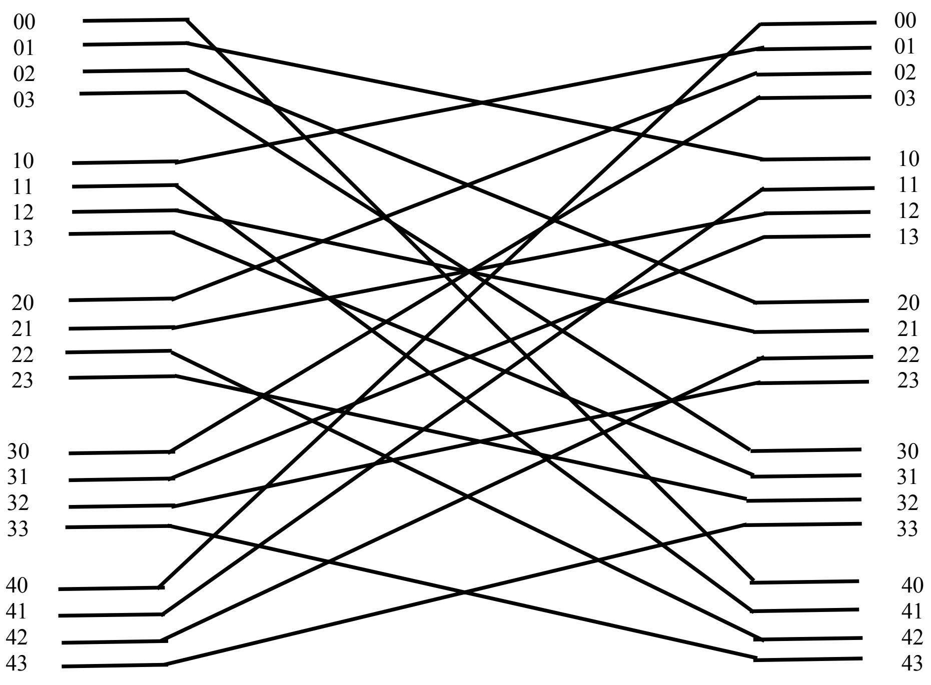


Figure 3-3: A 4-shuffle link connecting 20 sources to 20 destinations

3.2.2 The Communication Process

In this system, the entire requests are presented internally, i.e. within the crossbar module, regardless of whether the destination belongs to the same module as the source or belongs to another module. In case the destination belongs to another module, the bypass mechanism allows the data to go directly through the shuffle link to a client node in that module. The data is stored in a temporary buffer in that node in order to be sent to the final destination in the coming cycles. For example, assume in figure 3-1 that the output of node C12 shuffle port is connected via the shuffle to the input of the node C21 that represents the client destination of module 1 in module 2. Assume now that node C11 want to send data to node C2n. So the destination node belongs to module 2. The communication process takes place as follows: node C11 sends the data to node C12 (the client of module 2 in module 1). Node C12 (from the header) recognizes that the packet is sent to a node in module 2 and not to node C12 itself. So using the bypass mechanism, node C12 directs the packet to the output of its shuffle port which is connected to the input of the shuffle port of node C21 in module 2. Thus the data goes directly input of node C21 shuffle port and stored in its shuffle buffer (SBUFF). In the coming cycles, node C21 sends the data locally via module 2 to node C2n. Regarding the priority, the node gives higher priority to the requests coming from other modules in proportion of 32 to 1 for their own requests. This priority is measured for different values starting from 2, 4, 8… 32 and according to the results; the highest throughput was obtained at priority 31 to 1 for incoming request from other modules.

3.3 The Penta-S Performance

The performance measures defined in chapter 2 are adopted for measuring the performance of this scheme. The simulator which the author designed and presented in chapter 5 takes into consideration the definitions of bandwidth, throughput, delay, latency, and load, in addition to the detailed behavior of the scheme and the switches used as building blocks in it. Deriving a mathematical model for this scheme based on the model presented in [Ayya 04] proved to be complicated for the packets and queues. So a special care was given to the details in the simulator in order to best represent the true behavior of the scheme.

3.4 Macramé project

3.4.1 The Traffic and Timing Nodes

The aim of Macramé project test bed is providing the ability to study different topologies for large number of nodes. The building block used in this project is the STC104 32-way packet switch. Details of this switch are presented in chapter 4.

Traffic Nodes: The traffic node is a DS-link data source that generates programmable network traffic patterns. It drives 100 MBaud link and can simultaneously send and receive data at the full link bandwidth. In other words it emulates a computer connected to the network. Each traffic node is connected to a pair of links of the STC104. Each four traffic nodes are controlled by a Transputer 225. Figure 48 shows a schematic of a single STC104 switch as connected to traffic nodes.

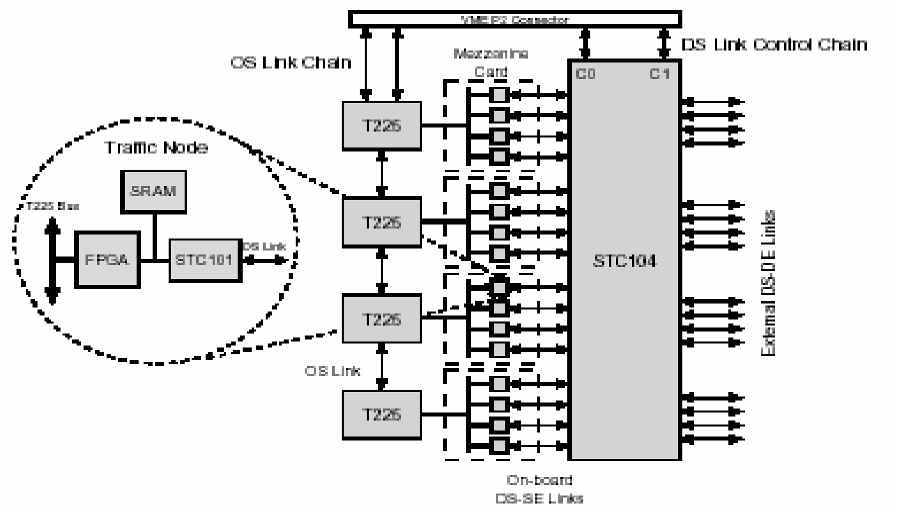


Figure 3-4: Traffic nodes connected to a single STC104 switch

Timing Nodes: Two timing nodes are provided for each switch; timing node consists of T800 transputer with 4 Mbytes of DRAM and STC101 parallel DS link adapter to measure network latency between specific terminals. This is done by injecting time stamped trace packets into network by one timing node. These trace packets traverse the network and are received and analyzed by the other timing node.

The timing node packet is of the following format

1. Routing header (1 to 2 bytes)
2. Source identifier(16 bits)
3. Transmit time stamp (24 bits)
4. packet length (16 bits)
5. Payload (packet body programmable)
6. End of packet character.

3.4.2 System Integration

The network topology can be configured as Clos, 2D-grid or tours network. A 1024-node 2D-Grid and a 512-node Clos networks were built. In 2D-grid they only used 16 out of the 32 ways of the switch to emulate 16-way STC104. 4 links are available between the switch and each of its 4 neighbors i.e. 16 links are used to connect 16 nodes to the switch. Figure 3-5 shows how 400-node 2D-Grid network is built of 25 switches of the STC104 type.

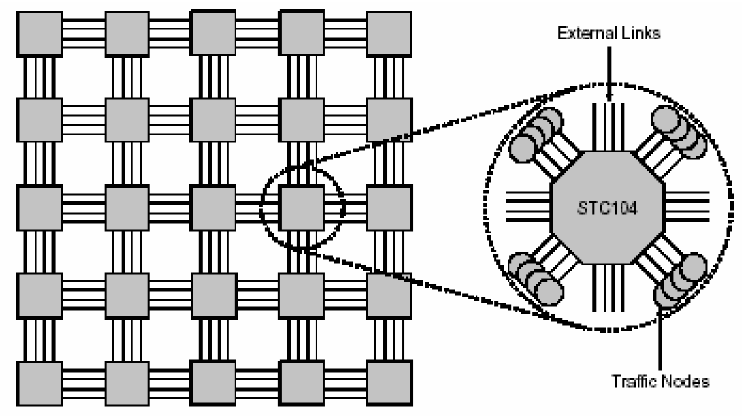


Figure 3-5: 400-node 2D Grid network [Haas 98]

Due to limited number of the switches available for the project and to the complexity expected in building 1024-node Clos network, the team of Macramé project decided to build a 512-node only Clos network [Hass 98]. Figure 3-6 shows a block diagram of the 512-node Clos network. A total of forty eight STC104 switched are used in constructing this network.

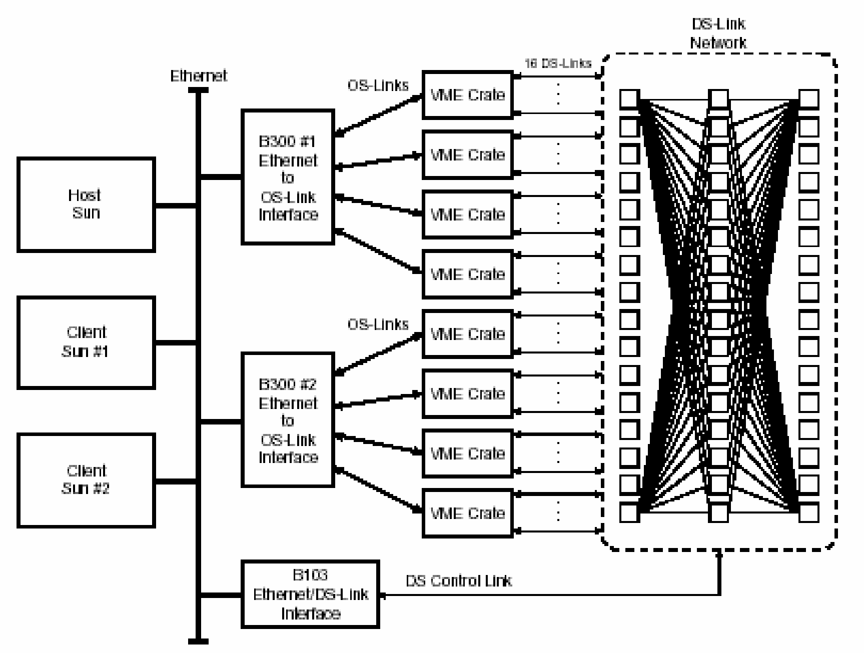

DS-Link Network
16 DS-Links
Ethernet
OS-Links
VME Crate
B300 #1 Ethernet to OS-Link Interface
Host Sun
Client Sun #1
Client Sun #2
B300 #2 Ethernet to OS-Link Interface
B103 Ethernet/DS-Link Interface
DS Control Link

Figure 3-6: Macramé Project 512-node Clos network [Haas 98]

The two networks were tested for various packet lengths, number of active nodes, and loads. The delay parameters and behavior feature of the STC104 were considered. Uniform Random traffic was used as a traffic pattern. The inter-arrival time between successive packets was generated according to Poisson distribution. Note that the packets produced by the timing nodes are considered as part of the traffic.

As will be shown in chapters 5 and 6, apart from the topology, the same concepts and parameters used in Macramé will be used in simulating the Penta-S topology.

Chapter 4

The STC104 and the NBWR Switches

4.1 Introduction

As mentioned in chapter 2, the larger the switch used as a building block in the network, the better the performance will be, provided enough links are available among the blocks. This is due to the fact that the crossbar switch allows the best non-blocking networking. Large switches can be built of smaller switches but it will be bulky and complicated. The larger the number of links integrated on one switch, the simpler building large networks is. In 1994, Thompson managed to integrate 32x32 serial link crossbar switches on a chip [Thomp 94]. This switch was named as STC104 switch. It is capable of connecting 32 nodes (PCs). The STC104 architecture obeys the IEEE 1355 standard, see appendix-A. Later, using this switch as a building block, researchers managed to build large networks which accommodate large number of nodes (~ 1024 computers). In Macramé project, two experimental networks with two different topologies were built using STC104 as a building block, Clos multistage interconnection network and 2D-Grid (with Torus version) [Haas 98]. In order to understand the behavior of these networks, the architecture of the STC104 must be studied.

4.2 The STC104 Switch

This section reports on the STC104 Asynchronous Packet Switch (APS). The main reference of the following information is the STC104 data sheet produced by SGC-Thomson Microelectronics [SGS-TH 96].

The STC104 is a complete, low latency, packet routing switch on a single chip. It connects 32 high bandwidth serial communication links to each other via 32 by 32 way non-blocking crossbar switch, enabling packets to be routed from any of its input links to any output link. The links operate concurrently and the transfer of a packet between one pair of links does not affect the data rate or latency for another packet passing between a second pair of links. Each link can operate at up to 100 Mbits/s, providing a bidirectional bandwidth of 19 Mbytes/s. Figure 4-1 shows a block diagram of the STC104 switch.

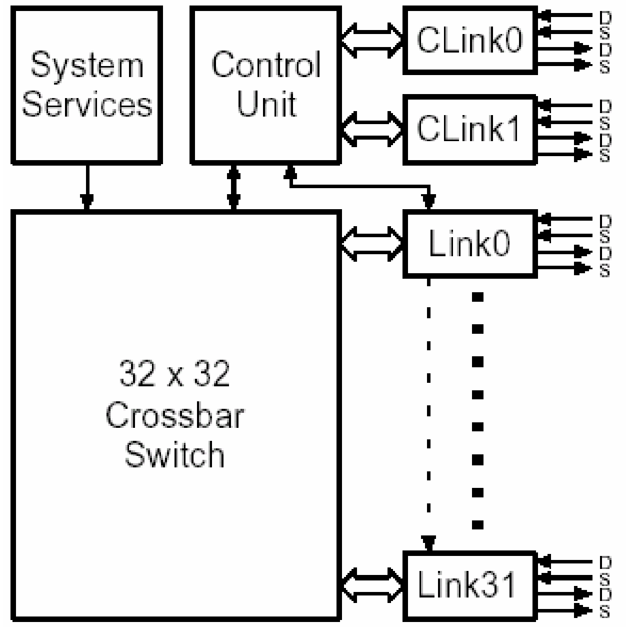


Figure 4-1: STC104 block diagram

The STC104 allows communication between devices, such as microprocessors, that are not directly connected. A single STC104 can be used to connect up to 32 microprocessors. The STC104 can also be connected to other STC104s to make larger and more complex switching networks, linking any number of microprocessors, link adaptors, and any other devices that use the link protocol. In the absence of any contention for a link output, the packet latency will be less than 1μ second through each STC104. Latency here means the time between the first bit of the packet being received on one link and being re–transmitted on another.

The STC104 is controlled and programmed via a control link. The STC104 has two separate control links, one for receiving commands and one to provide daisy chaining. The control links enable networks of STC104s to be controlled and monitored for errors. The control links can be connected into a daisy chain or tree, with a controlling processor at the root.

Data in an STC104 communication system is transmitted in packets. To enable packets to be routed, each packet has a header at the front which contains routing information. The STC104 uses the header of each incoming packet to determine the link to be used to output the packet. Anything after the header is treated as the packet body until the packet terminator is received. This enables the STC104 to transmit packets of arbitrary length. Figure 4-2 shows the packet structure.

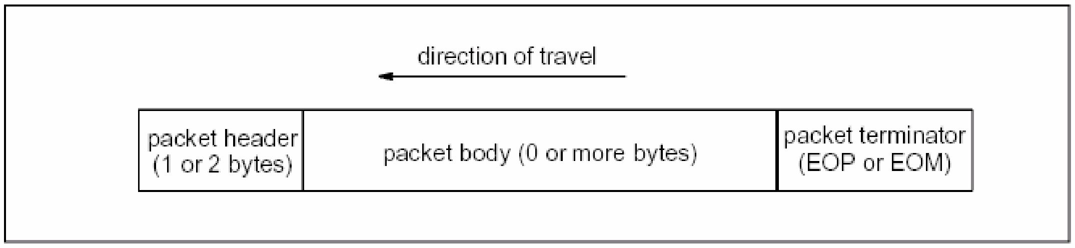

direction of travel
packet header (1 or 2 bytes)
packet body (0 or more bytes)
packet terminator (EOP or EOM)

Figure 4-2: Packet structure [Haas 98, SGS-TH 96]

In most packet switching networks complete packets are stored internally, decoded, and then routed to the destination node. This causes relatively long delays due to high latency at each node. To overcome this limitation, the STC104 uses wormhole routing, in which the routing decision is taken as soon as the routing information, which is contained in the packet header, has been input. Therefore the packet header can be received, and the routing decision taken, before the whole packet has been transmitted by the source. A packet may be passing through several nodes at any one time, thereby pipelining the transmission of the packet. Wormhole routing is invisible as far as the senders and receivers of packets are concerned, its only effect is to minimize the latency in message transmission.

In most packet-switching networks, Store-and-Forward routing is used, where each routing switch inputs the whole of a packet, decodes the routing information, and then forwards the packet to the next node. This is undesirable because it requires storage for packets in each routing switch and it causes long delays between the output of a packet and its reception.

4.2.1 Wormhole routing

The STC104 uses wormhole routing (figure 4-3) in which the routing decision is taken as soon as the header of the packet has been input. If the output link is free, the header is output and the rest of the packet is sent directly from input to output without being stored. If the output link is not free the packet is buffered.

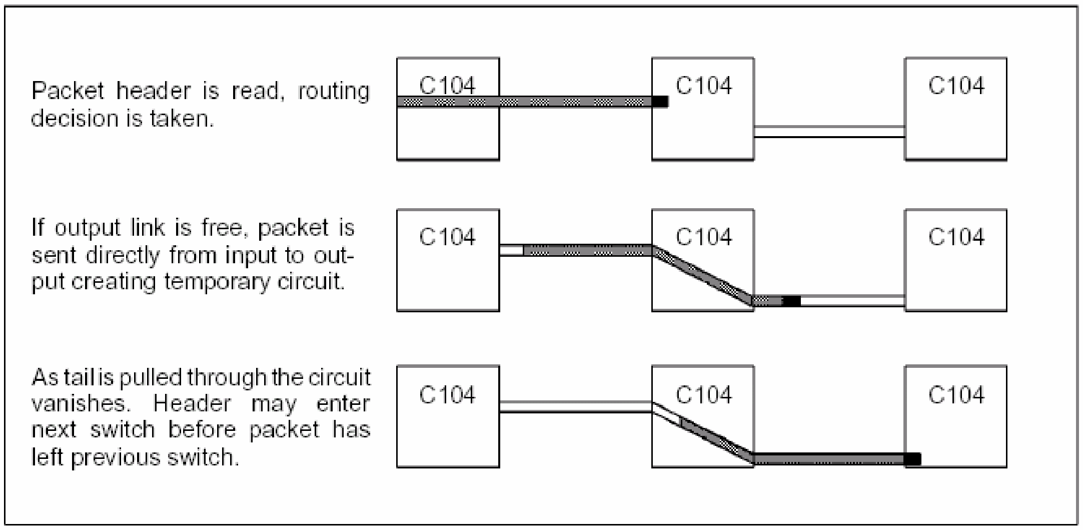

Packet header is read, routing decision is taken.
C104
C104
C104
If output link is free, packet is sent directly from input to output creating temporary circuit.
C104
C104
C104
As tail is pulled through the circuit vanishes. Header may enter next switch before packet has left previous switch.
C104
C104
C104

Figure 4-3: Wormhole routing [SGS-TH 96]

The packet header, in passing through a network of STC104s, creates a temporary circuit through which the data flows. As the end of the packet is pulled through, the circuit vanishes. The wormhole analogy is based on the comparison with a worm crawling through sandy soil, which creates a hole that closes again behind its tail.

4.2.2 Interval labeling

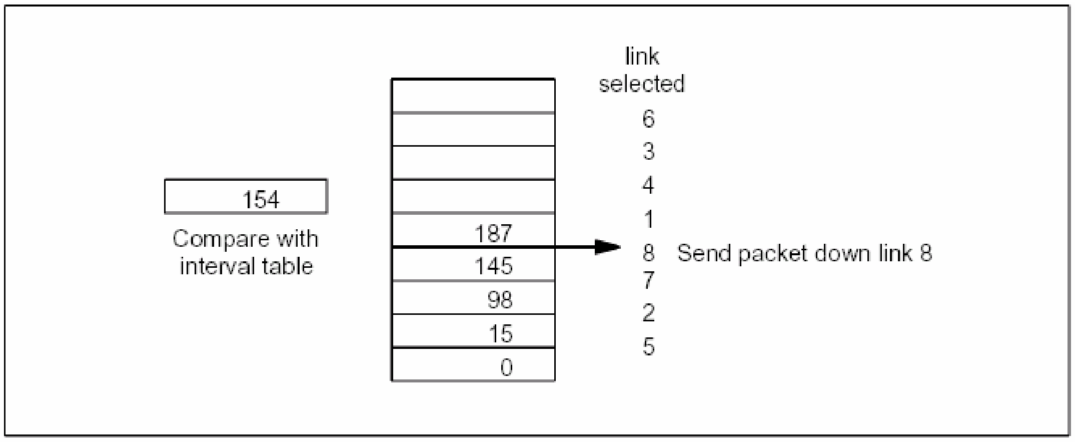


Figure 4-4: Interval labeling [SGS-TH 96]

Wormhole routing requires an efficient routing strategy to decide which link a packet should be output from. The STC104 uses a routing scheme called interval labeling, whereby each output link of an STC104 is assigned a range, or interval, of labels. This interval contains the number of all the terminal nodes (i.e. microprocessors, gateway to another network, peripheral chip, etc) which are accessible via that link. As a packet arrives at anSTC104 the selection of the outgoing link is made by comparing the header label with the set of intervals. This is illustrated in figure 4-4. The intervals are contiguous and non-overlapping and assigned so that each header label can only belong to one of the intervals. The output link associated with the interval in which the header label lies in the one selected. In the example the incoming header contains the value 154, which lies between 145 and 186, so the packet is output along link 8.

4.2.3 Buffering

To exploit the full bandwidth of the internal pathways on the STC104 there is buffering on each path through the device. The buffering is fully handshaken FIFO buffering with minimal latency.

Input Buffers. In this configuration the buffers are located at the input ports of the switch. Arbitration logic is needed to determine which of the packets held in different input buffers destined to the same output will be transferred through the interconnection matrix. The arbitration logic can be very simple, e.g. round robin. Input buffered switches are the easiest to implement, as the buffers and the switching fabric only need to run at the speed of the ports.

Output Buffers. With this architecture, the buffers are located at the output ports of the switch element. The assumption is that more than one packet from the input ports can cross the internal interconnection matrix and be stored in the same output buffer. This solution requires the use of a very fast internal cross-connect. In order to allow a non-blocking switch, the interconnection network and the output buffer have to be capable of handling N packets simultaneously, where N is the number of switch ports.

4.2.4 Adaptive Grouping

Consecutive links may be programmed to be 'grouped', so if a packet is routed to an output link which is busy it will automatically be routed along any other link in the group which is available. In this way performance can be optimized by allowing packets to be routed to any one of several outputs, depending on which link in the group is the first to become available. Grouping also provides fault tolerance.

The STC104 can be programmed so that the output link selected by the router is independent of the input link on which the packet arrives. Alternatively the STC104 can be programmed so that some link inputs are mapped to a specific set of link outputs. This can be used to enable independent networks to be implemented with the same STC104 with complete security. The STC104 can be programmed so that certain header ranges are marked as invalid in which case packets whose headers fall within this range are discarded. This can be used to enforce security in multi-user networks.

4.2.5 Hot-Spot Problem and Universal Routing

To eliminate network hot spots, the STC104 can optionally implement a two phase routing algorithm. This involves every packet being first sent to a randomly chosen intermediate destination; from the intermediate destination it is forwarded to its final destination. This algorithm, referred to as Universal Routing, is designed to maximize capacity and minimize delay under conditions of heavy load.

Usually packets are routed through the STC104 unchanged. However a flag can be set in the specified output link, in which case the header of the packet is discarded. Each link output of the STC104 can be programmed to delete the header of a packet, revealing a second header to route the remainder of the packet to the next destination device. This assists in the modular and hierarchical composition of routing networks and simplifies the labeling of networks. (For further details see appendix-B)

4.3 The STC104 Switch and Penta-S Expansion Scheme

After reviewing the STC104 switch and Penta-S expansion scheme, one can deduce that most of the STC104 features are not of great benefit if STC104 is used as a building block in Penta-S scheme. Even some of its features, like grouping and universal routing, degrade the Penta-S performance.

1. Input and Output Buffers: The STC104 has 70 characters of buffering on each of the 32 paths, of which 45 characters are on the input side and 25 characters are on the output side of the crossbar [Haas 98]. Data can be transferred from the input buffer to the output buffer at a speed three times the link speed. Transferring the data at this speed leaves the link free to transfer the other packet.

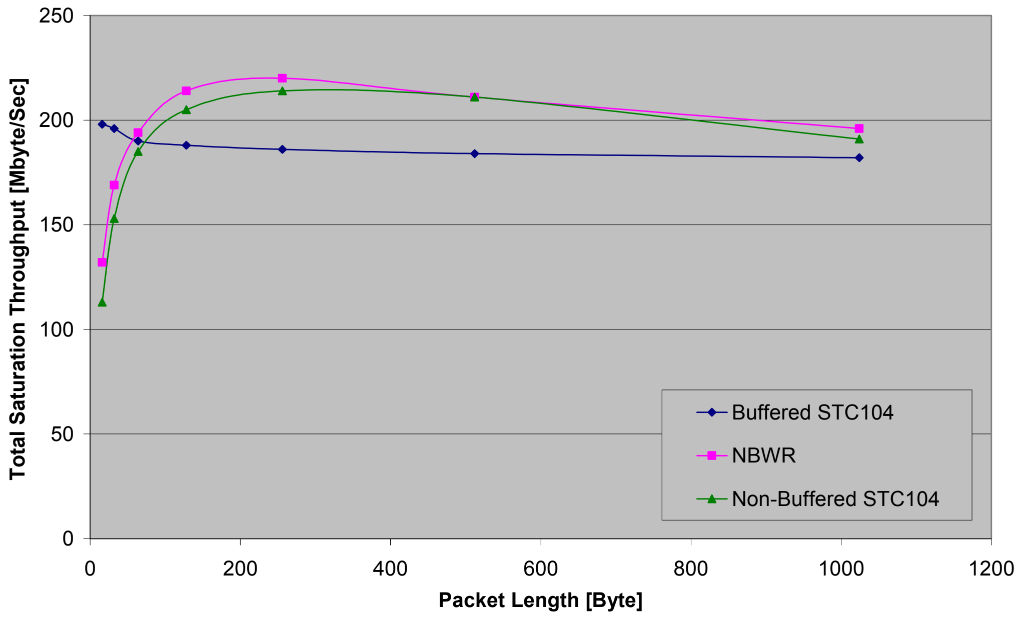


Figure 4-5: The total saturation throughput for the buffered, the non-buffered STC104 and NBWR switches under random traffic

So it is the size output buffer which makes the difference. However, because of the small buffer size, the advantage of the faster core speed is most significant for short packets [Haas 98]. Figure 4-5 shows that the best performance of the buffered STC104 is obtained at packet length of 25 bytes. This packet length is equal to the output buffer size. For packets smaller than 25 bytes this feature is underused and hence the performance is less. For packets longer than 25 bytes, the performance is degraded, because this feature starts causing congestion due to holding the link for longer time and not benefiting much from the buffer

The simulation results of non-buffered STC104 give a curve parallel to the systematic traffic curve. It shows also that for packet longer than or equal 64 byte, the non-buffered STC104 has a better performance than the buffered STC104 (STC104 can be programmed to be buffered or non-buffered). So for long packets, the non-buffered switch performs better than the buffered one.

2. Adaptive Grouping: The adaptive grouping technique provided in the STC104 suits the multistage interconnection networks. If the intermediate link is busy, another link of a pre-specified group can be used to lead to the next stage then to the final destination. Because of the one stage topology, using this technique in Penta-S leads to different destination locally or to different client (different block) globally. Therefore, the final destination won't be reached. So, this technique does not suit the Penta-S scheme.

3. Universal Routing: Using this technique in Penta-S implies sending the global packet to a different block from the destination block and asking it to send the packet to its final destination block. The packet will wait in the shuffle buffer of the intermediate block pending sending it to its final destination block. This increases the latency and leads to overloading the intermediate block. Rather than waiting in the intermediate block shuffle buffer, it is better to keep the packet waiting its turn in the source block arbitrator.

4. Interval labeling: this process is meant to replace the decoder in the traditional crossbar switches. It is a part of packet processor functions provided for each link of the STC104 switch. This technique needs a fully associative memory (CAM) in order to compare the packet interval with all stored intervals in the CAM simultaneously. In fact this needs a delay time equivalent to that of the decoder or more. Still the packet processor, and based on the destination link, resulting from this stage, places a request to the destination link arbiter. Each destination link of the switch is provided with an arbitrator.

5. Wormhole Routing: In this technique, the packet header is sent from one stage to the next, opening the links. When the link is open the packet body can follow if there is enough buffer to accommodate it, otherwise, the source node waits until the whole path from source to destination is open then it can be transferred in one burst. In the proposed non-buffered switch discussed in the next section, the wormhole routing is done by sending the header to a Serial-In-Parallel-Out (SIPO) shift register which keeps the destination address for the decoder. The decoder keeps presenting the request to the arbitrator until it is granted the link. The grant signal opens the link to the destination and signals the source node to start transmitting the packet body. On the arrival of the last bit of the packet, the link is closed and the arbitrator grants the link to another request according to a pre-defined priority. For global packets there is a large buffer associated with the client node which keeps the whole packet (including the final destination address) until it is presented to the switch of the destination block where it is processed and transferred to its final destination node in the same way as in the source block switch. Large FIFOs (16K X 9 bits) are available in the market since 1992 [IDT-DB 92]. For all above reasons we propose a simpler, faster, wormhole routing, and non-buffered switch.

4.4 The Proposed Switch

The proposed switch is a non-buffered crossbar switch with wormhole routing capability. A suggested name for this switch can be the **N**on-**B**uffered **W**ormhole **R**outing (NBWR) switch and so it will be referred to in this thesis.

As a wormhole routing switch, it allows the node to send the control information first and waits until the path is open then starts transmitting the packet. So, this switch is a collision free non-blocking switch. The transmit lines multiplex the control and data information.

4.4.1 The NBWR Switch Architecture

The main components of this switch are crossbar switch and a control circuit (see figure 4-6). The crossbar switch is just a matrix of switches, which connects horizontal and vertical lines in the network. These switches are switched on or off by the arbiters which are part of the control circuit.

The node presents the request for transmitting a packet to the control circuit via the transmit line, and waits until it receives a grant signal (via the transmit line as well) before it starts transmitting the packet body to its destination. Receiving the grant (acknowledge) signal implies that the control circuit has established a data path between the requesting (source) node and the destination node. In figure 4-6 the horizontal lines of the switch are connected to the outputs of the nodes (transmit lines) and the vertical lines are connected to inputs of the nodes (receive lines). Note that the transmit lines are connected also to the control circuit of the switch. This is to allow the source node to send the header to the control circuit. Note also that there is a simple connecting switch on each cross-point of the switch except the diagonal cross points, which connect between the input and the output of the same node. Of course no node needs to send a message to it self.

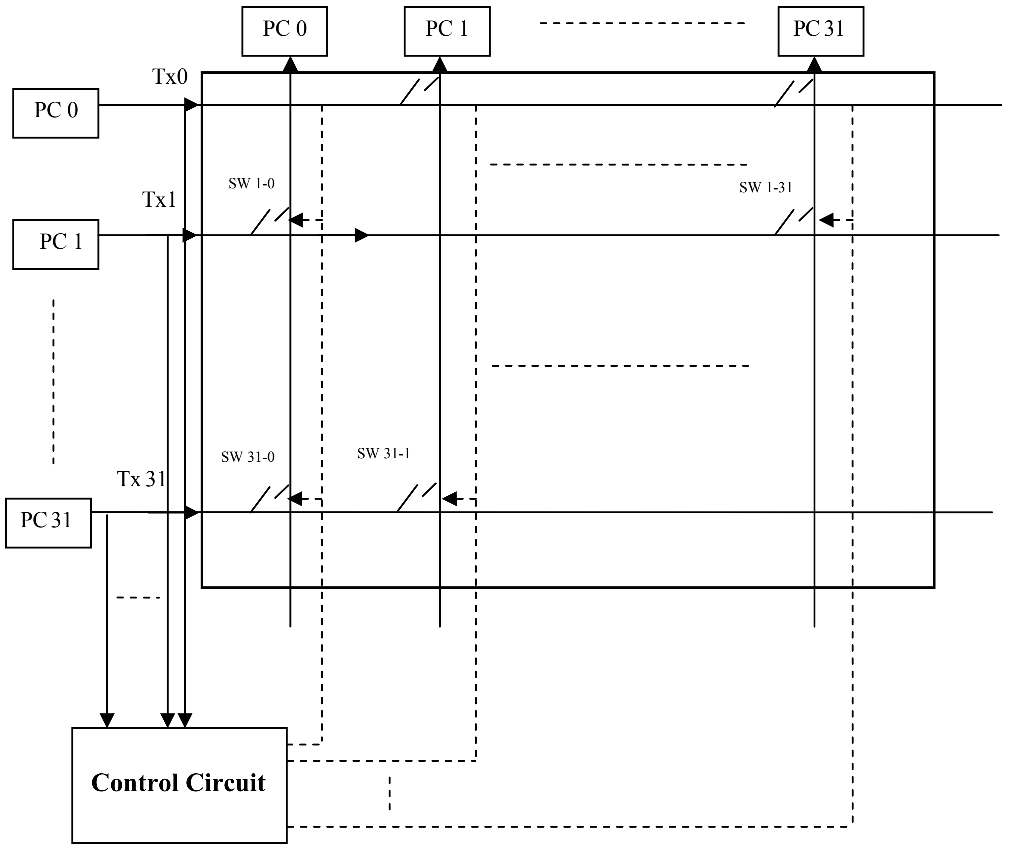


Figure 4-6: A block diagram of the NBWR switch

The requests are processed in the control circuit block. The suitable paths are established and acknowledgement signals are sent to the source node in order to start transmitting their packets.

4.4.2 The Control Circuit

The main components of the control circuit are the Link Control Circuit (LCC), The Serial In Parallel Out (SIPO) shift register, the decoders, and the arbiters. Figure 4-7 shows the details of the control circuit.

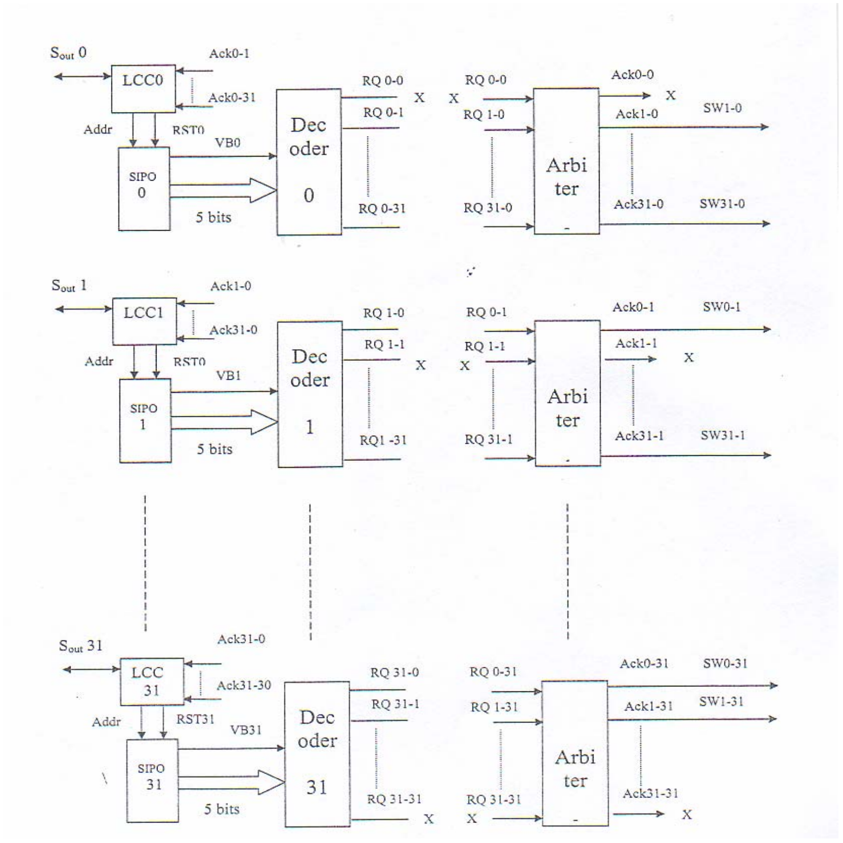


Figure 4-7: The NBWR control circuit

4.4.2.1 The LCC

This circuit recognizes the start of transmission and shifts the destination address (the header) bits into the SIPO shift register at the data link transmit rate. After the header bits are shifted into the SIPO, the transmit line is relinquished by the source node and the node starts waiting for the acknowledge signal. Upon receiving the acknowledge signal from the destination arbiter, the LCC sends a single pulse to the requesting node (via the transmit line). This pulse notifies the awaiting node to start transmitting the packet body.

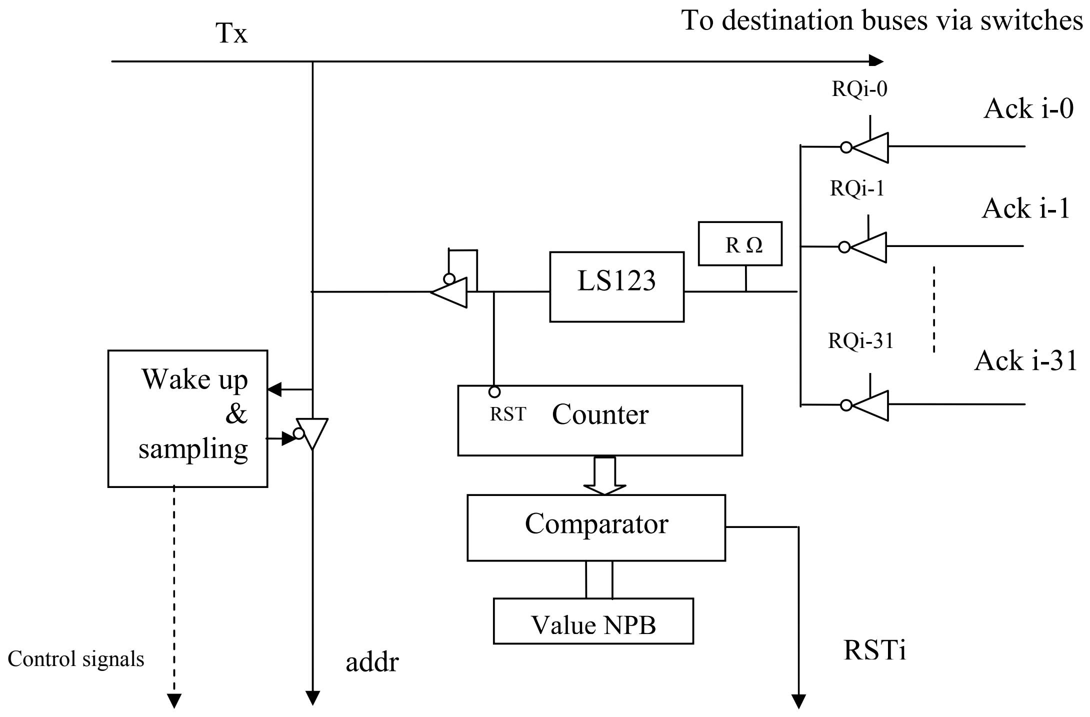


Figure 4-8: The LCC circuit

The SIPO keeps the destination address for the decoder which decodes the address and presents the request signal to the relevant destination arbiter. Note that there is an LCC, a SIPO and a decoder for each source line and an arbiter for each destination line.

4.4.2.2 The Arbiter

The arbiter circuit functions only when its destination is free. As far as its destination is busy receiving a packet, it won't issue an acknowledge signal to any source node. The arbiter control circuit provides the arbiter with a signal which indicates the status of the destination link (Busy/Free). The arbiter checks the status of all the switches leading to the destination link. If the destination link is not busy, it sends an acknowledgement message to the node to start sending its packet data. When the end bit arrives at its destination the link will be released and the arbiter makes another sample. The arbiter can be programmed to acknowledge the requests according to fixed or round robin priority. For further details on the arbiter circuit (see Appendix-C).

4.4.3 The NBWR Switch Delays

The packet is expected to face the following delays in the NBWR switch:

1. The SIPO delay: this delay is equivalent to the number of clock cycle consumed in shifting the header into the SIPO. For example, if the header is two byte, the delay is equivalent to 16 clock cycles.
2. The Arbiter delay: This is equal the delay between enabling the decoder by the destination free signal and outputting the acknowledge signal to the selected source node. This is equal to 560 ns, i.e., 56 clock cycles of a 100 MHZ clock (see appendix-C)
3. Pending Acknowledge Delay (PAD): This is the time the request waits on the arbiter before being acknowledged. This depends on the number of the request that have higher priority.

Chapter 5

The Simulation Program

5.1 Introduction

Often in hardware design, it is desirable to attempt to accurately simulate the anticipated behavior of the hardware before it is actually cast in silicon or used in larger construction. By simulating the behavior, this can demonstrate how successful (or unsuccessful) your design will be when it is fabricated or used in larger construction. The simulation can be subjected to a variety of conditions to demonstrate how it will behave in complex systems. It is also easier to implement design changes to a simulation than it is to actually change the design, extract it, test it, re-build it, and then test a prototype.

For computer networks, different data link layer and media access protocols can be simulated to see how they perform under various loading conditions. Also, cheaper, easier and cost effective designs can be tested.

For all these reasons, the work in this thesis tries to simulate the anticipated behavior of the new scheme for an expansion network topology called Penta-S. A non-buffered STC104 and the proposed NBWR switches are used as building blocks in this topology. The proposed switch uses the same line for sending data and control information, whereas STC104 uses separate lines for data and control information. The proposed switch has the ability to distinguish whether it is data or control information.

The simulation program will produce statistics on the following network performance terms:

- Network bandwidth and link bandwidth
- Network throughput and Per node throughput
- Average delay associated with a packet transmission
- Latency.

As in [Haas 98] these statistics can be mapped against various channel loads, packet lengths and number of nodes in the network to obtain a large amount of information about the behavior of a routing algorithm before the algorithm is implemented in an actual network.

5.2 The Simulation Model

During the literature search we tried to find a discrete event simulator in order to simulate the Penta-S model. Sophisticated discrete event simulators which can accommodate large models with large number of asynchronous parallel/sequential events, such as Penta-S model, are available. However, part of the assignments of this project was to use the software engineering skills and knowledge, the author gained during his study, to build an application specific discrete event simulator for simulating the performance of the Penta-S network. Moreover building own simulator enables us to describe our events precisely, beside freeing us from the restrictions of generic software packages, and giving us a better in-depth sight of the problem. So we decided to build our own simulator in Java.

A Java discrete event simulator has been written to evaluate the performance of the proposed scheme. The term "scheme" implies the expansion topology and its behavior under certain load, network size, protocol, and packet lengths. The behavior of the Penta-S architecture (described in chapter 3) will be investigated.

The simulator is programmed in Java for several reasons, it has many features that facilitate the proposed work, and these reasons can be summarized in the following points:

1. Java is a pure object oriented language, which means that any program designed using this approach models to characteristics of real objects using classes and objects.
   a. Objects are keys to understand object oriented technology, which means a software bundle of variables and related methods. These real world object share two characteristics: state and behavior.
   b. Class is a prototype that defines variables and methods common to all objects of a certain kind. Every class can communicate with other classes by sending messages to each other.
2. Encapsulation, which provides an object attributes or methods. This enable the change within a reasonable bounds without changing the interface which is visible to callers.
3. Inheritance is the ability to define a new class that inherits the behaviors and code of an existing class the new class is called a child while the original class is called parent class.
4. It is platform independent language.

5.2.1 Goals and Objectives

The main goal is to investigate the application of Penta-S scheme in computer networks and to see where it best fit among other topologies.

The algorithm presented in this simulation is designed to achieve the following objectives:

- To investigate the network expansion regarding its scalability, linearity and cost as compared with other network expansion methods.
- To test its performance parameters (throughput and latency) under various loads, packet length and network size (number of nodes in the network).
- To compare Penta-S with other known computer network switching topologies with respect to the above two points.

5.2.2 The Model Assumptions

The simulation model described in this thesis assumes the following points:

1. A 32x32 crossbar module which can connect 32 sources to 32 destinations is used as a building block in this scheme.
2. k modules are used in building the network, where k can have a maximum value of n+1, where n is the number nodes the switch can accommodate. Here, the model is applied for n=32.
3. Each node presents its packet in an arrival time decided by Poisson distribution depending on an arrival rate value λ calculated from the load (see Appendix D). The destination address of the packet is randomly generated between 0 and kn; So all nodes are equally likely to be addressed, regardless of being belonging to the same module as the source node or to another module.
4. The STC104 is used in this scheme with its real delay values as taken from [SGS-TH 96]. However, its extra features, like grouping and buffering are not used here for two reasons: firstly; grouping technique degrades the performance of Penta-S scheme, secondly; buffering does not add much to Penta-S scheme performance

and vastly complicates the simulation process. The proposed non-buffered switch (NBWR) behaves like a non-buffered STC104 but with smaller switching delays.

5. The packets that are presented to the switch and find their destination busy wait on the arbiter until they are granted their turn to open and use the destination link rather than waiting in the STC104 small buffer or being discarded.
6. When granted the destination link, local packets (their destinations lies within the same block) are transferred directly to their destinations, whereas the global packets are bypassed by the local client to the destination client in the destination block via the shuffle link and stored in the shuffle buffer of the destination client pending its turn to be sent to its final destination node.
7. Packets originated from the node itself and those coming to the node shuffle buffer via the shuffle link from other blocks are presented to the switch according to an optimized priority (the priority which gives the best performance).

5.2.3 The Model Scenario

The scenario here means the actual sequence of events that occurs when a node needs the service of the scheme to transfer a packet to a destination node. The Destination node within the switch can be the actual destination node or a local client node which bypasses the packet to a destination client in the destination block. The source node can be sending its own generated packet or working as a client sending a packet coming from another block (stored in its shuffle buffer) to its final destination. Figure 5-1 shows the two possibilities. Note that node xj is the client of block j in block x and node jx is the client of block x in block j. When node xi, for example needs to send a packet to any node in block j, it sends it to node xj, which directs it through the bypass to node jx in block j. If the final destination is node jx, it will be received by node jx itself. If the final destination is another node in block j, the packet will be stored in the shuffle buffer of node jx. Finally node jx sends the packet to its final destination in block j (follow the dashed line). If the packet is sent to a local node in block x (including the client, where each node in the block represents a client of another block), it will be received by the local destination node immediately after the path is established (follow the solid line).

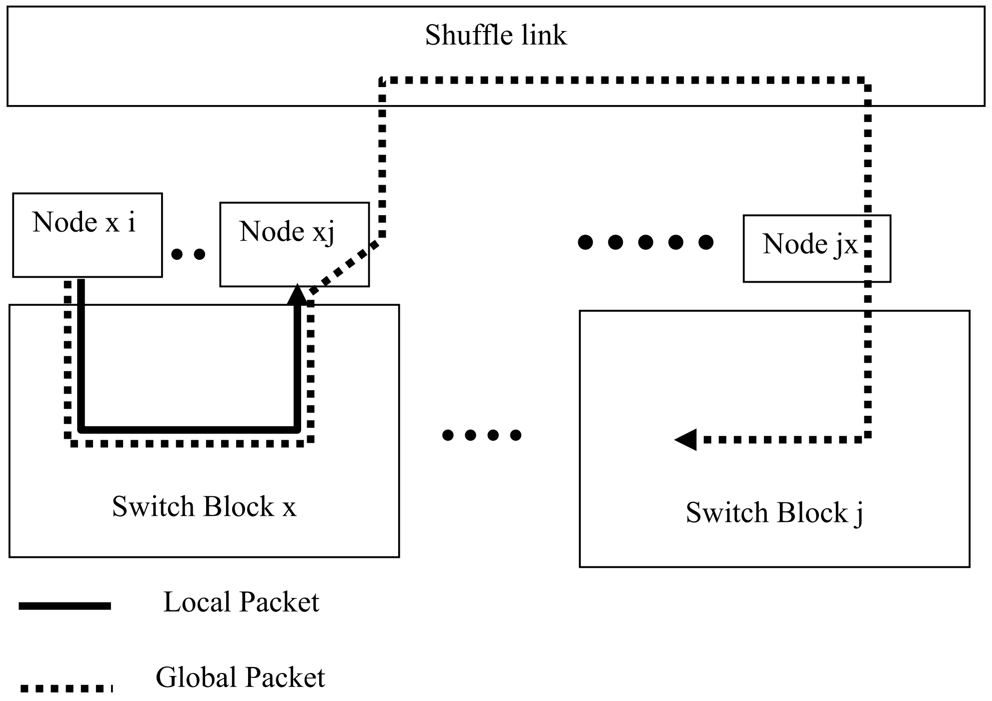


Figure 5-1: Local and global packet paths

Figure 5-2 shows the stages of sending a packet from the source node to the destination link, regardless of whether the destination link is leading to the destination node or bypassed to the shuffle to reach the client in the destination block.

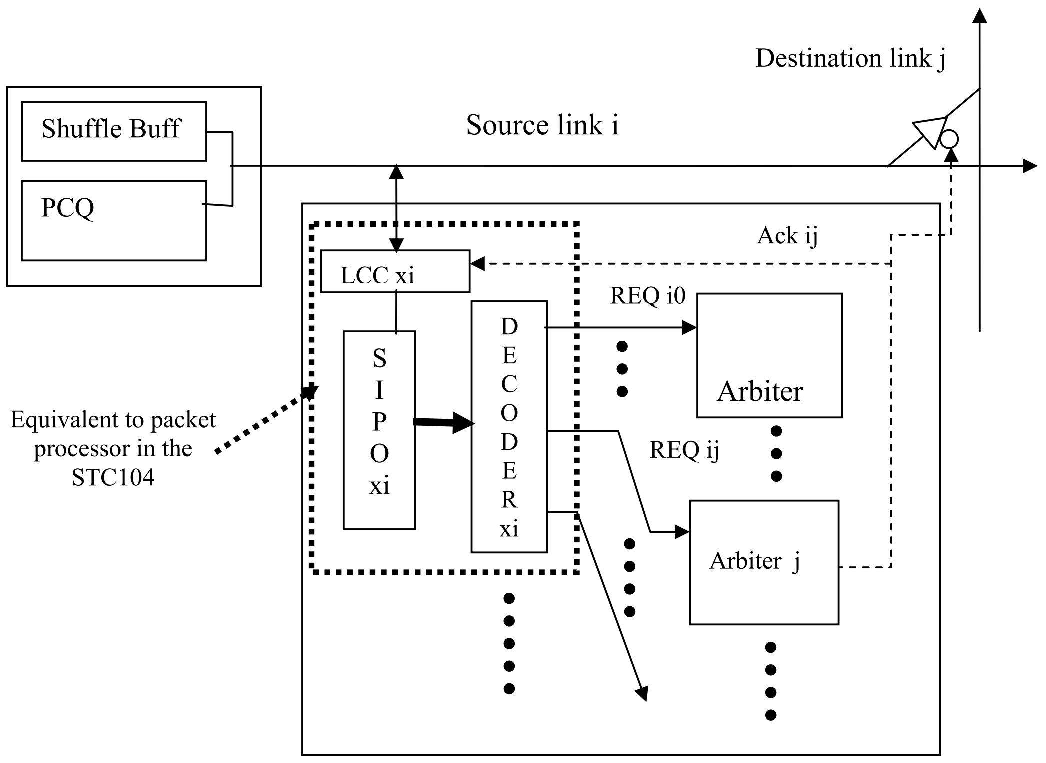


Figure 5-2: The stages of sending a packet from source (i) through destination link (j)

When the time comes to send a packet, the node sends the header first to the packet processor in STC104 or to the SIPO of NBWR switch. The header includes the destination address. This takes a number of clock cycles equal to the number of the header bits. After being processed (or decoded), a request is presented to the destination queue in the packet processor or to the destination arbiter in the NBWR switch. The node keeps waiting until it receives an acknowledge signal from the packet processor of the STC104 or the destination arbiter circuit in the NBWR switch. The acknowledge signal implies that a path is established to the destination or the client node in the destination block. So the source node starts transmitting the packet body to its destination. Upon the completion of sending the packet body, the switch to the destination link is closed and the arbiter gives another acknowledge signal to another pending request.

5.3 The Simulation Algorithm

5.3.1 General description

This algorithm represents the behavior of a Penta-S network built of non-buffered STC104 or NBWR switches according to the model and scenario discussed in section 5.2.3. The algorithm takes into consideration that all parts of the network (nodes, packet processors, links…) are working in parallel unless there is a contention for a link or the event is not due. Figure 5-3 shows the general algorithm of the simulator. An iterative mathematical model of the Penta-S network is presented in Appendix-E, and a UML model of the simulator is presented in appendix-F

The simulator is run for a large number of clock cycles (~ 10^4 clock cycles) called the Run Time (CLK_MAX). Each clock cycle equals 10 ns (100 MHZ clock frequency). In each clock cycle, all activities which are due in stages 2 to 5 of the algorithm are performed in all parts of the network.

The simulator uses a virtual clock. The variable CLKN indicates the current clock cycle number. CLKN is incremented after the simulator passes through all source and destination nodes updating all the events that are due in this cycle.

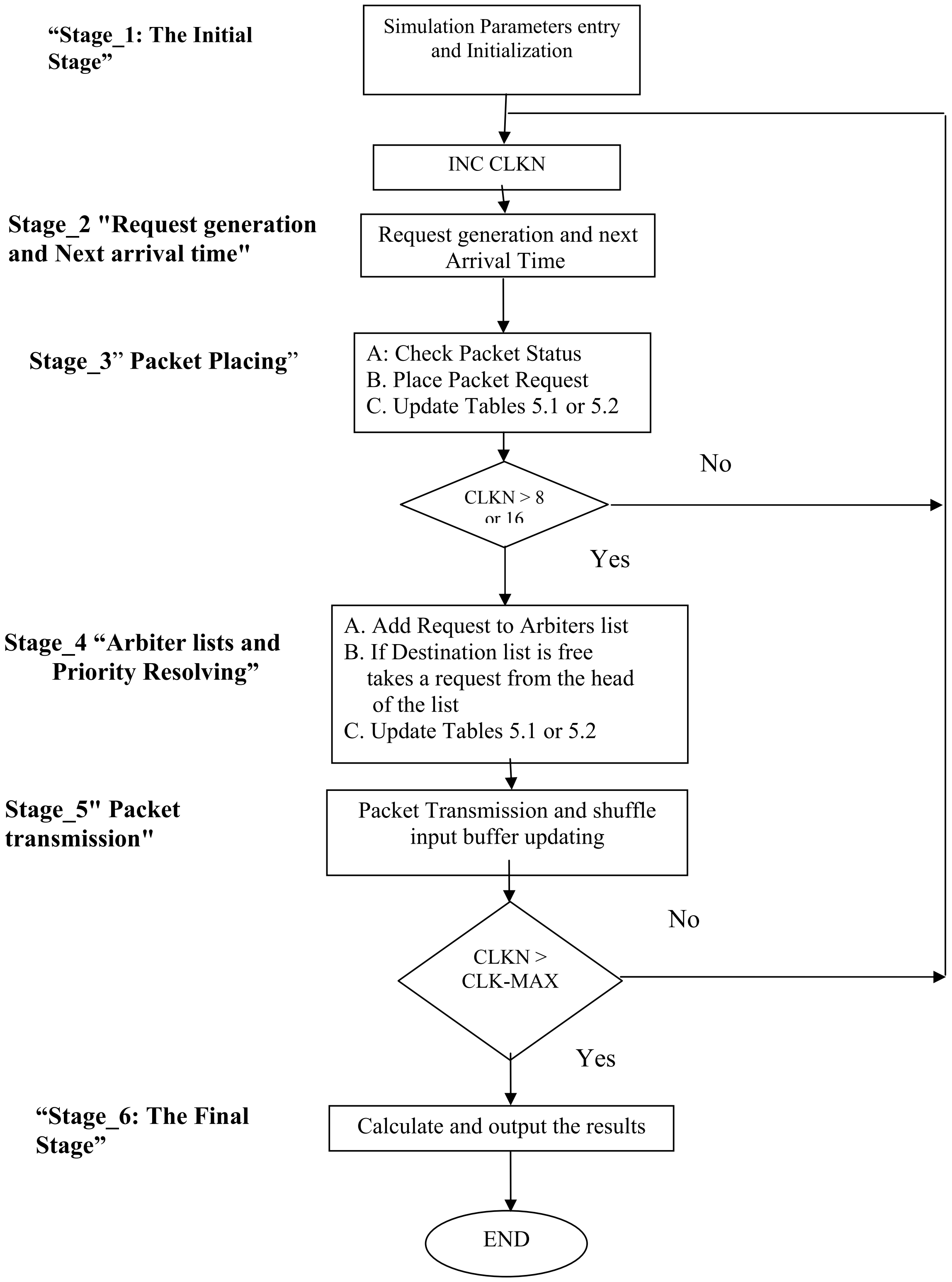

"Stage_1: The Initial Stage"
Simulation Parameters entry and Initialization
INC CLKN
Stage_2 "Request generation and Next arrival time"
Request generation and next Arrival Time
Stage_3" Packet Placing"
A: Check Packet Status
B. Place Packet Request
C. Update Tables 5.1 or 5.2
CLKN > 8 or 16
No
Yes
Stage_4 "Arbiter lists and Priority Resolving"
A. Add Request to Arbiters list
B. If Destination list is free takes a request from the head of the list
C. Update Tables 5.1 or 5.2
Stage_5" Packet transmission"
Packet Transmission and shuffle input buffer updating
CLKN > CLK-MAX
No
Yes
"Stage_6: The Final Stage"
Calculate and output the results
END

Figure 5-3 Flow chart of the algorithm

5.3.2 Stage_1: The Initial Stage

At the beginning, the program displays a form which enables the user to enter the simulation parameters as shown in figure 5-4. The simulation parameters that the user is allowed to enter are:

1. The length of the run-time (CLK-MAX) in clock cycles.
2. The packet length in bits.
3. The number of nodes in the block (module).
4. The number of blocks in the system.
5. The load percentage, where the load percentage is defined as the average number of bytes/link presented to the network divided by the maximum link capacity, all multiplied by 100%.

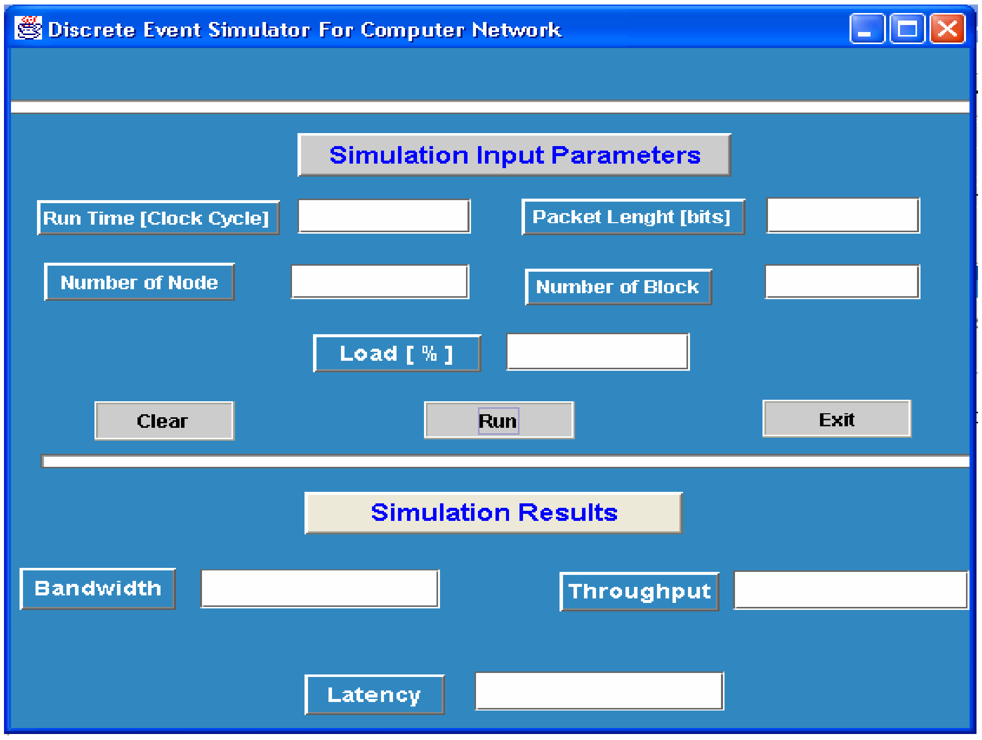


Figure 5-4: The interface page of the simulator

After that, the program creates instances of the tables, queues and lists, corresponding to the number of nodes and blocks entered by the user then initialize their attributes to zeros. It also initializes CLKN variable (The clock cycle counter) to zero.

5.3.2.1 The Queues, Lists, and Tables

In order to monitor and control the packet life cycle (from source to destination) the following support queues, lists and tables are needed:

a) The PC queue (PCQ) b) The Shuffle Buffer List (SBUFF) c) The Arbiter List d) The local packet table (table 5-1) e) The Global packet tables (Tables 5-2) f) The arbitration table (Table 5-3).

a) **The PC Queue (PCQ):** packets that are generated by the PC of the node are queued in this buffer queue. The packet is generated by the method **Req_Gen( )** when its Next arrival Time (NAT) is equal to the current CLKN.

b) **The Shuffle Buffer List (SBUFF):** This is buffer list in which the global packets coming from other blocks are stored pending sending them to their final destination by the client node.

c) **The Arbiter List:** This is a buffer list in which the packets requests presented to the arbiter of the destination link waiting the acknowledgement to start transmitting their body to the destination.

d) **The Local Packet Table (Table 5-1):** Table 5-1 is designed to accommodate the attributes and transition points of the local packet life cycles. A line of the table is reserved for each node in the system. The node is defined by the source PC number (SPCN) and the source block number (SBLKN). Only one packet per node can appear in this table. The previous packet of the node is overwritten by the new one. When the packet is completely transferred, the wait time of the packet (LPTST – LPRST) is added to the current value of TWT (Total Wait Time), and the number of totally transferred local packets TTLP is incremented by 1. If the node is representing a destination client and transmitting a global packet in its second stages, only three attributes of the table are used with the node; the Original Source PC Number (OSPCN), the Original Source Block Number (OSBLKN) and the Original Packet Number (OPN). These attributes helps the simulator to update the transition points of the packet in its original table (Table 5-2).

Table 5-1: The Local Packet Table

SPCN	SBLKN	OSPCN	OSBLKN	DPCN	LPRST	LPTST	LSTATUS	TWT	TTLP	LPN	OPN

Where the

1. SPCN: Source PC number.
2. SBLKN: Source block number.
3. OSPCN: Original source PC number.
4. OSBLKN: Original source block number.
5. DPCN: Destination PC number.
6. LPRST: Local packet request start time.
7. LPTST: Local packet transmits start time.
8. LSTATUS: Local status.
9. TWT: total wait time.
10. TTLP: Total transferred local packet.
11. LPN: Local packet number.
12. OPN: original Packet number.

e) **The Global Packet Tables (Tables 5-2)**: A copy of this table is created for each node to keep a track of its global packets. A record of a global packet should not be overwritten by the next. This is simply because when a global packet completes its first stage, the source node should be able to generate another packet (local or global). Should the global packet be overwritten after its first stage is completed, the second stage of its life cycle will be missing in the throughput and the delay time calculations.

Table 5-2: The Global Packet Table

OPN	DPCN	DBLKN	GPRST1	GPTST1	GPRST2	GPTST2	GSTATUS1	GSTATUS2
1								
2								
3								
4								

● ● ● ●

● ● ● ● ●

TWT1	TWT2	TTGP

Where the

1. OPN: Original Packet number.
2. DPCN: Destination address of PC.
3. DBLKN: Destination address of block number.
4. GPRST1: global packet request start time 1.
5. GPTST1: Global packet transmits start time 1.
6. GPRST2: Global packet request start time 2.
7. GPTST2: Global packet transmits start time 2.
8. GSTATUS1: Global status 1.
9. GSTATUS2: Global status 2.
10. TWT1: Total waits time 1.
11. TWT2: Total waits time 2.
12. TTGP: Total Transferred Global packet.

The second stage is accomplished by the destination client node. The transition points in the second stage are updated in table 5-2 using the information associated with the packet in the client's SBUFF and shown in table 5-1. This information is the original source PC number (OSPCN), the Original Source Block Number (OSBLKN) and the Original Packet Number

(OPN). This information is used to link between the destination client in table 5-1 and the original source node in table 5-2.

f) The Arbitration Table (Table 5-3): The arbiter belongs to the immediate destination in the block. This table keeps a record of the destination status. If the destination input is busy receiving a packet body (A-status = 'DB'), the system would not acknowledge another packet from the arbiter list to start transmitting its body. If the destination input is free (A-status = 'DF'), the system acknowledges a packet from the head of the arbiter list and allows its source node to start transmitting the packet body to the destination.

Table 5-3: The Arbitration Table (Destination)

ABLKN	ARBN	A-status

Where is the

1. ABLKN: Arbitration Block Number.
2. ARBN: Arbiter List Number (i.e., which destination node).
3. A-Status: Arbitration Status (DF, DB).

5.3.3 The Four Stages of the Packet Life Cycle (stages 2 to 5)

In each clock cycle of the run-time, the simulator passes through four stages (stages 2 to 5) updating the due events. The four stages represent the stages through which the packet passes from its generation time to the completion of its arrival at its destination. When the run-time elapses, i.e., CLKN > CLK-MAX, the simulator moves to the final stage, in which it collects the data and performs the calculation.

The four stages which the simulator passes through in each clock cycle of the run-time are:

1. **Stage_2 (Request generation and next arrival time calculation):** In this stage, the simulator goes to each node of the system, matches if the next arrival time (NAT) variable calculated from previous cycles is equal to the current clock cycle number (CLKN), and if so, the simulator calls the request generator (Req_Gen ( ) method) to generate a request, otherwise the simulator jumps to the next node. The generated request (represented by the destination address) associated with the current value of NAT is stored in a buffer called PCQ. The current value of NAT is stored with the generated address as the arrival time of the request (AT). After storing the request and its arrival time in the PCQ, Poisson method (poison ( )) is called to generate a new inter-arrival time (IAT). The IAT is added to the current value of the clock cycle CLKN to make the new value of the next arrival time which is stored in the variable NAT in order to be checked in the coming cycles.
2. **Stage_3 (Packet Placing):** In this stage, for each node, the simulator checks whether the node is busy transmitting a packet or not. If the node is not busy, the simulator checks the PCQ and the shuffle buffer (SBUFF) of the node for packets pending transmission according to an optimized priority. If the requests in the PCQ have higher priority, the simulator first checks the arrival time attribute of the packet at the head of the PCQ. If the current value of the clock CLKN is >= the arrival time, this means that the packet transmission is due. So the simulator takes the packet from the head of the PCQ, decides if the destination is local or global, and assumes starting shifting the header (the destination address plus the valid bit) into the SIPO register. At this point the

simulator updates the packet request start time in table 5-1 for local request (LPRST = CLKN) and in table 5-2 for global requests (GPRST1 = CLKN). If the PCQ is empty, the simulator checks the SBUFF for pending requests (coming from other blocks) and if the SBUFF is not empty, it takes a request from the head of the SBUFF lists, assumes starting shifting the header into the SIPO and updates the packet request start time of the second stage of the packet transmission (GPRST2 = CLKN) in table 5-2. If the SBUFF list is empty, the simulator goes to the next node. If a packet header from the PCQ or the SBUFF is assumed being shifted into the SIPO, the status of the node is updated as **busy** in transmitting a packet (LSTAUS = 'Busy' in table 5-1, or (GSTATUS1 or GSTATUS2) = 'Busy' in table 5-2). If the priority is given for the packets pending in the SBUFF, the above operations start with the SBUFF list then moves to the PCQ if the SBUFF list is empty. The above arguments apply to this case as well. If both lists are empty, the simulator moves to the next node.

3. **Stage_4 (Arbiter List and Priority Resolving):** This stage involves two operations; adding elements to the arbitration list and removing elements which the transmission of their packet body is due.

   a) Adding elements to the arbiter list: For each packet in each source node, the simulator checks if a number of clock cycles equals to the number of header bits has elapsed after the packet request start time (LPRST or GPRST1 or GPRST2 depending on the type and the status of the packet) or not. If yes, the packet number and its source address are added as a new element to the end of the arbiter list. The arbiter list represents the requests pending on the arbiter to be served. For each destination link there is an arbiter. The arbiter is selected by decoding the destination address of the packet.

b) Removing elements from the arbiter list: The simulator checks each destination link to see whether it is free or busy receiving a packet (A-status ='DF' or 'DB' in table 5-3). If the destination link is free and its arbiter list is not empty, the simulator takes an element from the arbiter list and assumes the start of the packet body transmission. At this point the simulator updates the status of the packet by marking its transmission start time (in table 5-1 or 5-2) equal to the current CLKN value; LPTST for local requests, GPTST1 for global packet in the first stage of transmission, and GPTST2 for global packet in its second stage of transmission. Also it updates the status of the destination link as "busy" (A-status = 'DB' in table 5-3).

4. **Stage_5 (packet transmission):** In this stage of simulation, for each node, for each packet, the simulator checks the packet status. If the transmission starts time is not zero and the difference between the packet transmission start time and the current CLKN is equal to the packet body number of bits, this means that the packet has completed transmission and the packet status has to be updated as "completed". The status of the relevant nodes (the source node, the mediator client nodes, and the destination node) is updated according to the following cases:

a. If the packet is local, the status of both source and destination is updated as "free" (LSTATUS ='Free' in table 5-1 and A-status = 'DF' in table 5-3).

b. If the packet is global in its first stage (transmitted from the source, bypassing the local client to the SBUFF of the client in the destination module), the status of the source node and the local client is updated as “free” (GSTATUS1 = ‘Free’ in table 5-2 and A-status = ‘DF’ in table 5-3), and the packet number (OPN), the packet final destination header, and the source address (source node number and source module number) are added as an element in the SBUFF list of the client in the destination module (stored as OSPCN, OSBLKN, and OPN).

c. If the packet is global in its second stage (transmitted from the client in the destination module to its final destination), the status of the client and the final destination nodes is updated as “free” (LSTATUS =’Free’ of the destination client in table 5-1, GSTATUS2= ‘Free’ in table 5-2, and A-status = ‘DF’ in table 5-3). In all cases the total wait time TWT is updated in the relevant table.

5.3.4 The Final Stage

After reaching the clock cycle CLKN> CLK_MAX, the simulator exits the simulation loop run between stages 2 to 5, and starts collecting the data from tables 5-1 and 5-2 in order to calculate the bandwidth, the throughput, the delay and the latency.

In order to calculate the bandwidth of the network in (bits/sec), the program takes the sum of totally transferred local packets (TTLP) value from table 5-1, then finds the totally transferred global packets from table 5-2 (TTGP) by summing the last (TTGP) value of table 5-2 for all nodes of the system. The two sums are multiplied by the packet length (PL) in bits then divided by the (run time multiplied by the clock period). The throughput of the network in (Bytes/sec) is obtained by dividing the bandwidth by 8. Equations 5.1 and 5.2 show the bandwidth and the throughput of the system.

$$BW = \frac{\left(\sum TTPL + \sum TTGP\right) * PL}{(CLK\_MAX * 10^{-8})} \quad \text{(Bit/Sec)} \qquad 5.1$$

$$TP = \frac{BW}{8} \quad \text{(Byte/Sec)} \qquad 5.2$$

The Program calculates the delay and latency also from tables 5-1 and 5-2. The delay is defined as the time elapses between presenting the packet request to the network and starting transmitting the packet body; (LPTST-LPRST) for local packets in table 5-1 and (GPTST2-GPRST1) for global packets in table 5-2. At the end of each packet transmission (in the stages 2 to 5), the value (LPTST-LPRST) is added to total wait time TWT in the node line of table 5-1 and the value (GPTST2-GPRST1) is stored in the TWT2 for each packet line in table 5-2. In the final stage, order to calculate the packet delay, the program takes the sum of TWT column in table 5-1 and adds it to the sum of all TWT2 columns of tables 5-2 then divides the total by the number of totally transferred packets. This gives the delay in clock cycles. Multiplying the result by the clock period gives the delay in seconds. Equation 5.3 shows the delay calculation;

$$Delay = \frac{\left(\sum TWT + \sum TWT2\right) * 10^{-8}}{\left(\sum TTLP + \sum TTGP\right)} \qquad 5.3$$

The latency is defined as the time elapses between presenting the packet transmission request and the arrival of the last bit of the packet to its destination, which means that

Latency = [Delay + (Payload*10^{-8}) 5.4

Where Payload = Packet-length + extra bits 5.5

And the extra bits are the bits of the source address, the destination address, the ‘packet length’ value, and the end character associated with the packet body.

The cost-efficiency of the network is obtained by dividing the throughput by the number of switches (crossbar module), k, used in building the network. The cost-efficiency is given by:

$$CE = \frac{TP}{k} \qquad \text{(Mbytes/sec/switch)} \qquad 5.6$$

5.4 Running the Program

For each set of entry vales of figure 5-4, we had to run the program ten times in order to obtain the average of the throughput and the latency of that set. Before the ten times running, various values were tried in order to reach to the steady state calculations. The run-time value (CLK_MAX $=10^5$) was adopted for all tests. We tried to make the program to run the ten times for different sets of entry values in order to create tables of throughput and latency versus different values of active nodes, loads, and packet lengths, but unfortunately, the PC started giving “out of memory” system error message. So we had to run the program as mentioned above. The average values obtained from the ten runs, are stored in Excel tables for throughput and latency for various sets of entry values. The results presented in chapter 6 are obtained from these tables.

Chapter 6

Results

6.1 Introduction

This chapter presents the results obtained from the simulator described in chapter 5. The performance of Single switch, Penta-S built of STC104 and Penta-S built of NBWR have been studied and compared with other topologies like multistage Clos n, and 2D-Grid networks. The results presented here for different network sizes, packet lengths and loads. In order to obtain steady state results, a run-time value (CLK_MAX) $=10^5$ (discussed in section 5.4) was entered in each running of the program. In this thesis all results are measured under two constraints, first using random traffic and second taking all results at saturation case, where the saturation means the maximum achieved throughput. This is because the results of this thesis are compared with standard version of Macramé and this version applied these constrains.

After defining the simulation terms for the STC104 (see section 6.1) we start running the simulator to obtain results for the STC104-based Penta-S. In a second stage we repeat the same experiment for the NBWR-based version of Penta-S.

We start running the simulator by entering a set of values of the run-time, the Load%, the packet length, and the number of active nodes (number of blocks (k) and the number of nodes in each block), then running the program to obtain the throughput and latency for the entered set.

The simulator is run ten times for each set of entry values, and then the average of the ten results is taken and stored in the relevant Excel table. For example a certain set of load percentage, number of active nodes, and packet length is considered an entry and the simulator is run ten times then the average of the throughput and the latency are taken and stored in the relevant Excel table. This process was repeated until we obtained tables of the throughput and latency for all possible entry sets which enabled us to plot any set of results versus any set of entries. These tables enabled us to plot result under the same entries and conditions used in Macramé project. The calculations of the throughput were obtained by the simulator from tables 5-1 and 5-2 using equation 5.2 for the throughput and equation 5.4 for latency.

In section 6.2 we discuss the single switch performance by measuring two factors; the switch throughput and the latency in sections 6.2.1 and 6.2.2 respectively. In section 6.3 we discuss priority optimization. Section 6.4 discusses the results of the network expansion. In section 6.5 a comparison of the NBWR-based and the STC104-based Penta-S networks is shown. Finally section 6.6 discusses the cost and cost-efficiency of Penta-S as compared to other network topologies.

6.2 The Simulation Terms

Before running the simulator some terms had to be defined and given values inside the program body. Things like switching delay, the delay of sending the header to the switching circuit, the packet body (the payload), the link capacity, and the equation which defines the relationship between the Load and the arrival rate must be defined in the program body for the STC104-based and the NBWR-based versions of Penta-S network. These terms and their values are defined as follows:

1. The switching delay of STC104 =0.92 µs = 920 ns = 92 clock cycles of 100 MHZ clock.

   For NBWR switch = 5.6 µs = 560 ns = 56 clock cycles of 100 MHZ clock.

2. Delay of sending header to the switching circuit:

   2bytes=16 clock cycles for STC104 Local packet.

   2bytes=16 clock cycles for STC104 Global packet.

   1byte=8 clock cycles for STC104 mediator node.

   For NBWR Switch

   1byte=8clock cycles in all cases.

3. Payload= PL + extras

   Extras for STC104 = 1 byte for Global packets.

   = 2 bytes for source address.

   = 2 bytes for packet length

   = 1 End characters.

   The total extras are 6 bytes = 48 bit for global packets and 5 bytes = 40 bit for local and mediator packets.

4. Each link of the switch works at 100 MHZ clock= 100 Mbit/sec theoretically = 10 Mbyte/s practically.

5. The Network Load

   The network load is defined as the node transmission bit rate.

   The load equals

$L=\lambda * PL;$ λ is arrival rate.

PL is packet length.

eg.. For $\lambda = 10^{-3}$ packet /clock and PL = 64 byte

$$L = 10^{-3}/10^{-8}$$

$$L = 10^{5} \text{ packet / sec}$$

$L=10^{5} * 64 = 6.4$ Mbyte/sec

The load percentage = (Load / Maximum link bandwidth) * 100%

In this case

L%= (6.4 Mbyte/sec / 10 Mbyte/sec) * 100% = 64%

The packet arrival rate is calculated for each run of the simulator from the assumed load and the packet length.

6.3 Single switch performance

In order to improve the correctness and accuracy of proposed switch and expansion mechanism, first we applied a single STC104 switch parameters during the simulation, taking into consideration the packet processing delay time for this switch without the buffering idea. After that, a comparison between the results from the Macramé for a single buffered STC104 with the simulation results of the non-buffered STC104 switch was made.

The performance is measured according to two factors; throughput and latency, as presented in the following sections.

6.3.1 Switch Throughput

The simulator calculates the throughput of the single switch from table 5-1 (all requests are local in the case of a single switch) using equation 5.2 under the condition k=1, i.e., number of switches in the network equals one. The simulator is run for several values of packet lengths (64,128,256,512, and 1024 bytes), under several load values (20%, 30%, 40%, …, 100%) for each packet length. The saturation throughput then is taken and plotted against the packet length in order to see the effect of packet length on the throughput. This test also produces results for latency presented in section 6.3.2. Figure 6.1 shows the saturation throughput for a buffered STC104, a non-buffered STC104 and the proposed NBWR crossbar switch with varying packet length.

It is clear from the figure that the non-buffered STC104 switch has a better performance than non-buffered STC104 for packet lengths greater than or equal to 64 byte. Compared with proposed NBWR, it has nearly the same performance for packet length 32 and the same performance at packet length 512 bytes. For the rest of packet length values, the proposed NBWR switch has a better performance than the non-buffered STC104 switch. Also, for packet lengths less than 64 bytes the NBWR switch has less performance than the buffered STC104 switch.

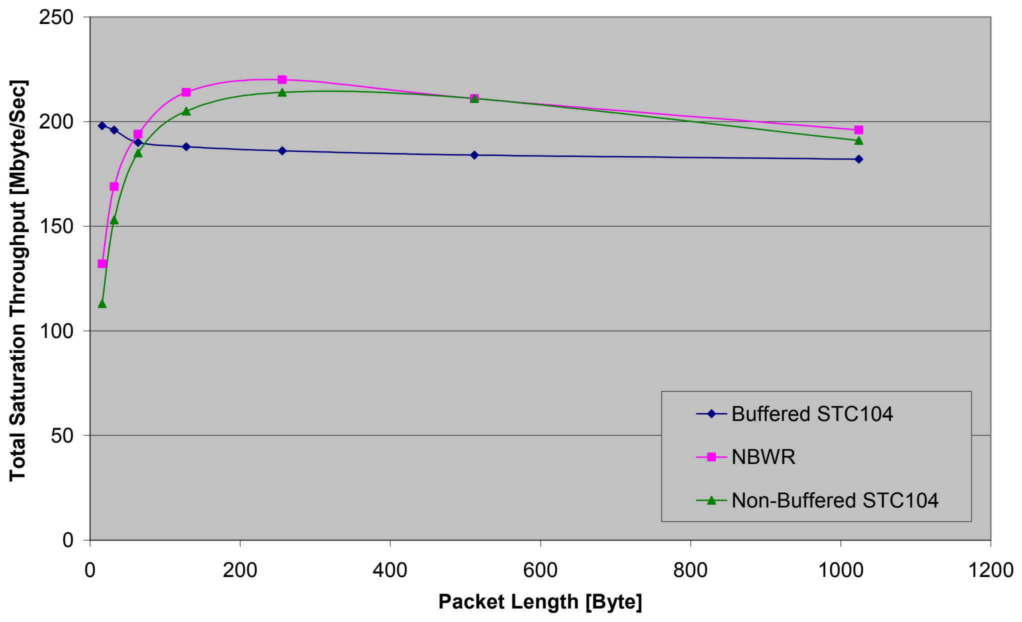


Figure 6-1: The total saturation throughput for the buffered, the non-buffered STC104 and NBWR switches under random traffic

These results indicate that the buffered mechanism suits a small packet length. This is because the data can be transferred from input buffer to output buffer faster. For larger packet length the contention and delay increases because the size of input buffer is 45 characters and output buffer is 25 characters, theses sizes are not enough for larger packet length. Also indicate that the proposed switch is suitable for large packet lengths. This is because in any case of transferred data using this switch, the header is first sent to reserve the link between a source and destination then the packet body is sent through the established link. So, the overhead of reserving the link (the delay) is the same for all packet lengths. The difference is that the delay represents smaller percentage of the total packet transfer time for longer packets. This primarily means the throughput must be better for longer packets.

6.3.2 Packet Latency

The simulation process discussed in section 6.3.1 also produces results for latency versus various values of packet length depending on table 5-1 and equation 5.4. As discussed in chapters 2 and 5, the network latency means the delay between the head of the packet entering the network at the source to the packet being completely received at the destination.

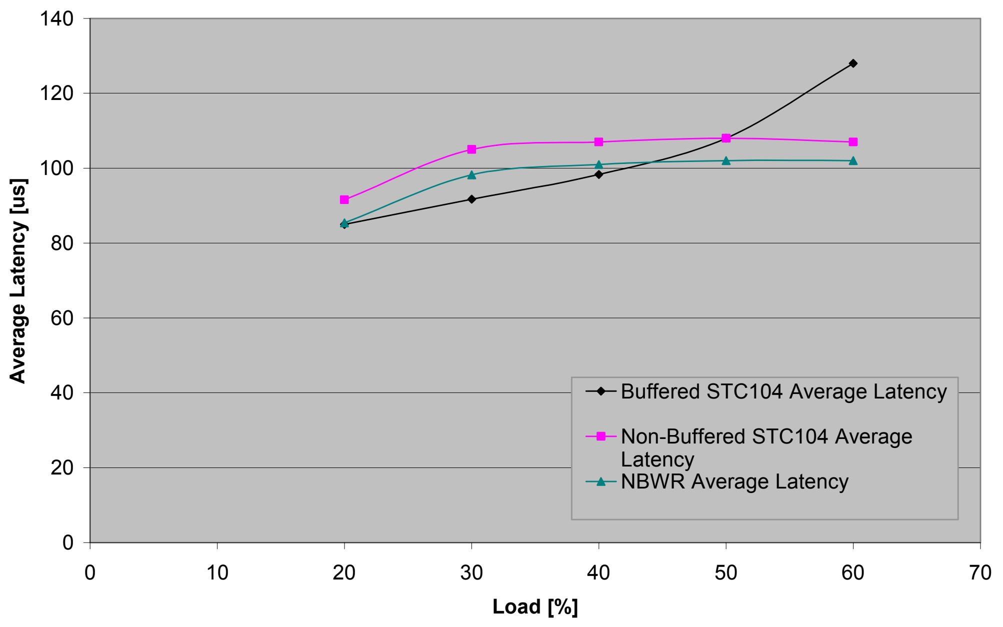


Figure 6.2: Latency for buffered STC104, Non-Buffered STC104 and NBWR switches

The latency of a single buffered and non-buffered STC104 switch was measured by using 30 traffic nodes rather than 32 nodes because the Macramé STC104 remaining two links of the switch were used to connect the transmitting and receiving timing nodes [Haas 98]. Figure 6-2 shows the average latency as a function of theoretical applied load for the three switches.

In general, the average latency increases with the network load for three topologies but with different factors. In Buffered STC104, for low loads, only few packets are blocked due to the existence of the buffer. As the network load increases, more packets are blocked and therefore the queuing delay also increased. The latency of the buffered STC104 switch crosses with the latency of the non-buffered switch at load 50% and with the proposed NBWR at 45%. Then the buffered STC104 latency becomes larger than the latency of non-buffered STC104. The proposed switch has a smaller latency than the non-buffered STC104 switch for all load values. This is because the switching delay time used in proposed switch is less than delay used in non-buffered STC104 switch. Also, it has a better latency than the buffered STC104 for load greater than 45%. Generally, the latency of non-buffered switches tends to be stable for large loads.

6.4 Priority Optimization

This priority is embedded inside the program body. While testing the program, we ran it several times over various network and packet sizes with different priorities between the shuffle incoming packet buffer (SBUFF) and the PC originated Packet Buffer (PCQ). The priority which gives the best performance is used in the simulators in all subsequent tests. The priority optimization was examined under random traffic for various samples of packet lengths and network sizes. The same results were obtained. Figure 6-3 shows the results for 512 nodes and 64 byte packet under random traffic.

Figure 6-3 shows also that the best performance is obtained at priority 32:1 for SBUFF to PCQ. This simply implies that in order to prevent long latency in the SBUFF, in 32 clock cycles the system must give higher priority to the SBUFF packets versus one clock cycles to the PCQ. This means clearing the pending global packets in the midway before serving new packets.

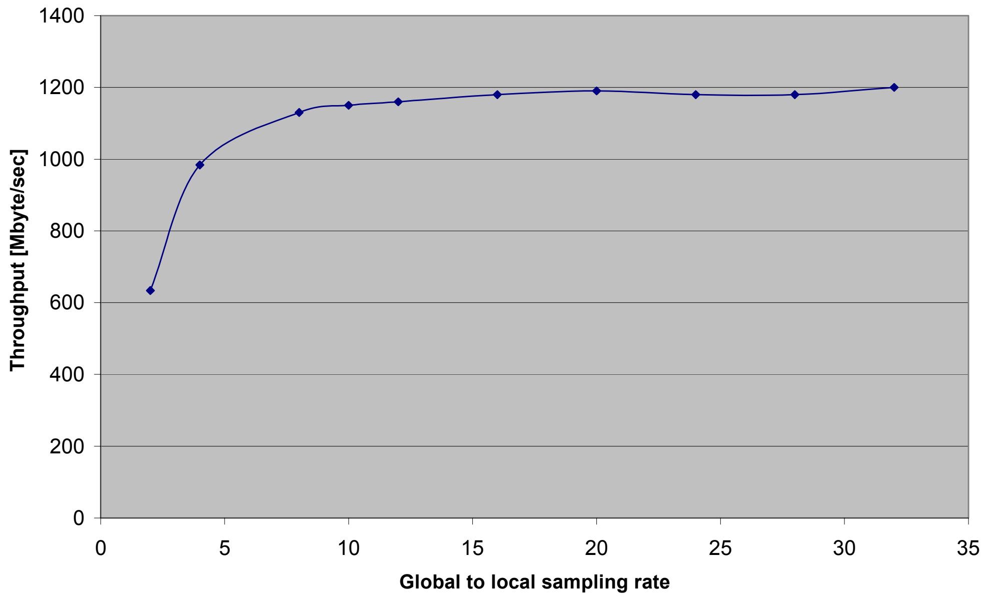


Figure 6-3: Optimization

This goes with the fact that in Penta-S, as all nodes are equally likely to be addresses, regardless of being in the same block or not, it is expected that the number of global packets (which are going to reside in the SBUF after completing the first stage) is always greater than the local packets which reach their destination in one stage. So, in order to achieve better throughput, higher priority must be given to the second stage of the global packet. So this priority is adopted throughout the whole simulation process.

6.5 The Results of Expansion Network

After testing the single buffered STC104, non-buffered STC104 and proposed switches by simulator, the simulator gave good results matching the results obtained from Macramé. This enables us to simulate the proposed shuffle expansion network on the non-buffered switch and proposed NBWR switch, then comparing the results with other network topologies like 2-Dimensional grid, and multistage Clos networks as presented in [Haas 98].

6.5.1 Expansion Scalability

In this test the simulator was run for various network sizes (number of nodes= 64, 128, 256, 512, and 1024 nodes), packet length =64 bytes, and various loads (20%, 30%,…., 100%). The throughput saturation values are plotted against the number of active nodes.

Figure 6-4 shows the saturation throughput for different sizes of Penta-S built of NBWR (Non-Buffered Wormhole Routing) switches, Penta-S built of non-buffered STC104 switches, Clos and 2D-Grid networks built of STC104, under random traffic for 64 byte packets.

Figure 6-4 also shows that Clos network has the best performance. For number of nodes < 320 the 2D- grid has better performance than Penta-S. For number of nodes >= 320 the Penta-S considerably improves, so that its performance becomes comparable to that of Clos network. For number of nodes >= 256 the three networks scales linearly.

The reason for the low performance of Penta-S for small number of nodes is the small number of shuffle links available among the blocks; this represents bottleneck for small size of this type of network topology.

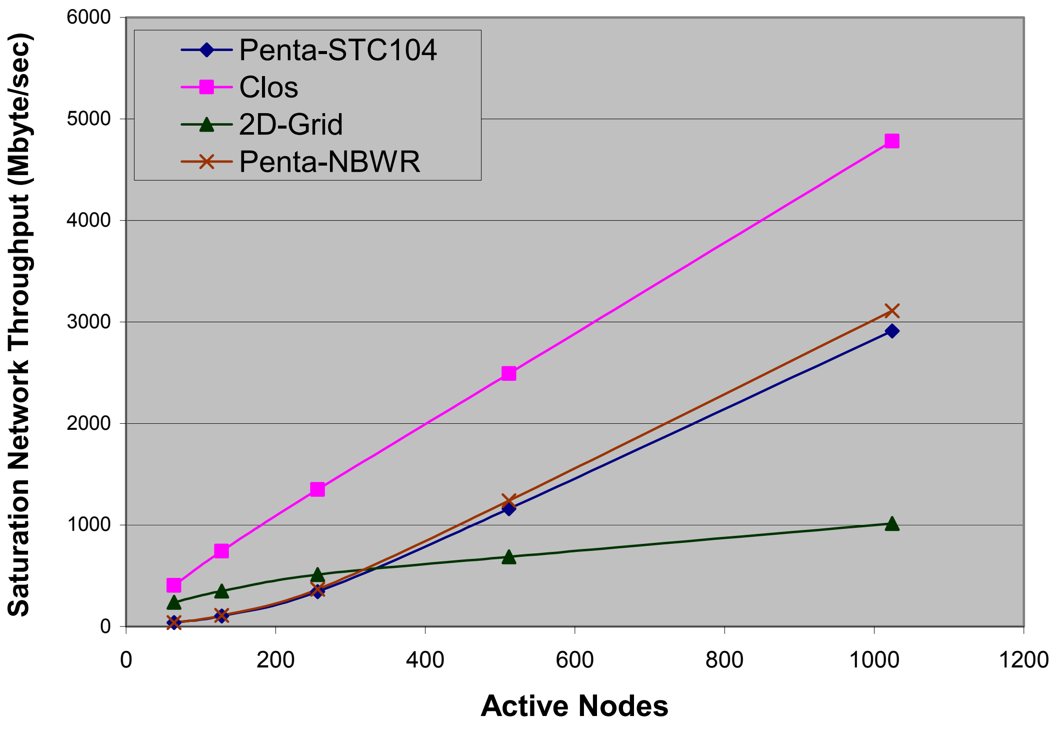


Figure 6-4: the saturation throughput for different sizes of Penta-NBWR, Penta-STC104, Clos and 2D-grid Networks

6.5.2 Network Node Throughput

The Per-node throughput shows the share of the node in the total throughput. The higher this share the better the performance and the less traffic contention in the network are. The per-node throughput, like throughput is affected by the packet length and the number of active nodes. To test this feature, we ran the program for several values of packet length (64,128,256,512, and1024 bytes) and loads (20%, 30%, …. 100%) two times; one for number of active nodes = 256 nodes and the other for number of active nodes = 1024 nodes. The saturation throughput values are obtained and divided by 256 to obtain the first curve of the graph in the first time and by 1024 to obtain the second curve in the second time. Macramé project has the same set of results for the 2D-Grid network. So we can compare our results with their results.

Figure 6-5 shows the per-Node saturation throughput versus packet length for two sizes of Penta-S and 2D-Grid networks under random traffic. As expected the 2D-Grid network has a better performance than Penta-S for networks of 256 node size. For number of nodes equal 1024 the Penta-S become considerably higher than that of 2D-Grid network.

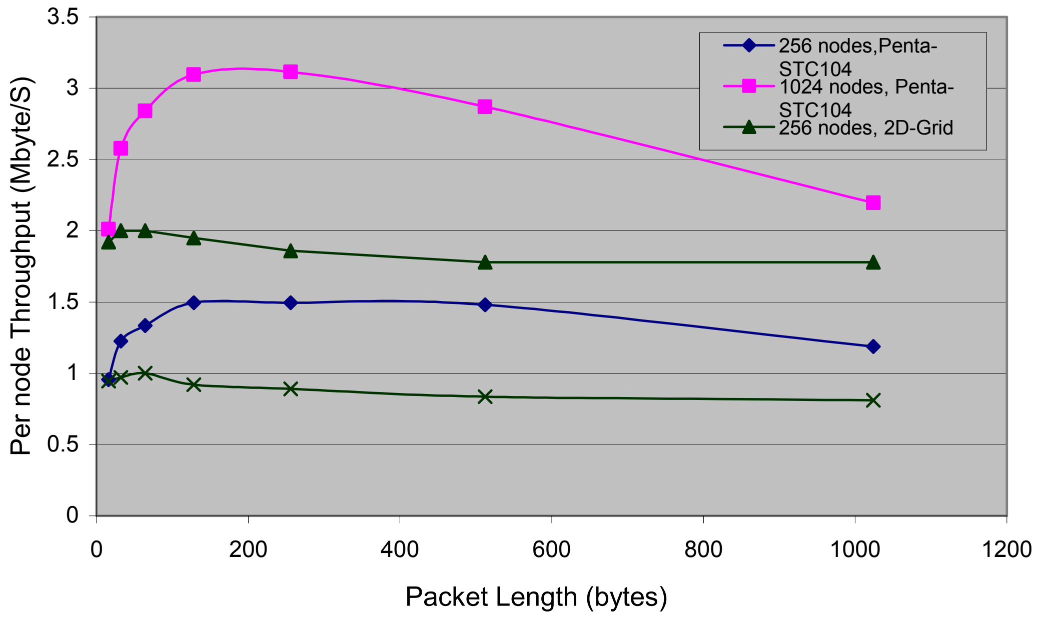


Figure 6.5: Per node throughput versus packet length for various sizes of Penta-S and 2D-Grid networks

Also the performance of both network grow with the packet length to a certain point then starts dropping. The maximum per-node throughput for 2-D-Grid occurs at 64 byte packet length for both network sizes. In Penta-S network, the maximum per-node throughput occurs at 128 byte packet length for both sizes of the network.

These results indicate that Penta-S network suits large number of nodes. Its performance grows with the network size. 2D-Grid network performance is degraded as the network size increases. This can be explained by the fact that in 2D-Grid network, the switch is connected by limited number of links with its four neighboring switches (4 links with each switch). These links are good enough if the packets are sent to the neighboring switches. However, packets going to farther switches have to cross number of switches before reaching these destinations. In Penta-S the number of links grows with the network size. However, regardless of the destination, the packet passes through one extra switch maximum.

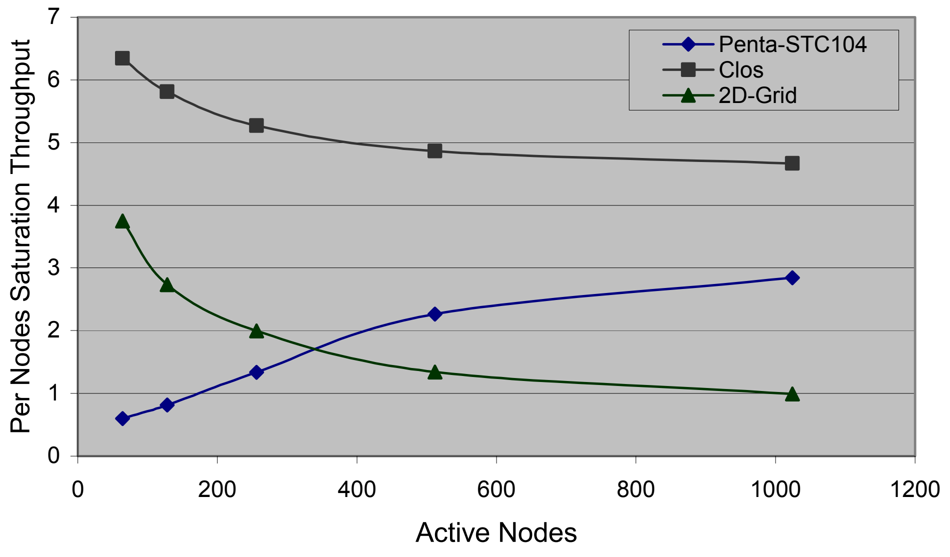


Figure 6-6: Per Node saturation throughput versus number of nodes for Penta-S, Clos and 2D-Grid Networks

In order to see the effect of the network size on the per-node throughput, we just divide the results obtained in section 6.5.1 by the corresponding number of active nodes. The results are shown in figure 6-6.

Figure 6-6 shows that Clos still has the best per-node throughput among the tested networks. This is due to the fact that the packets, unlike in Penta-S, are pipelined in the network stages and that the packet being transmitted does not wait in a middle buffer and contend with the newly generated packets for the switch links.

6.5.3 Network Normalized Throughput

In order to obtain a sample of the normalized throughput, we ran the program for several "number of active nodes' values (64,128,256,512 and 1024 nodes) for a packet length = 64 bytes under several load values (20%, 30%, 40%…, 100%). To obtain the normalized throughput, the saturation throughput was obtained and divided by the maximum throughput (the number of nodes X the maximum link capacity), where the maximum link capacity here is 10 Byte/sec as defined in Macramé project and adopted by our project.

Figure 6-7 shows the normalized network saturation throughput for Penta-STC104 and Clos for different size under random traffic with 64 byte packets.

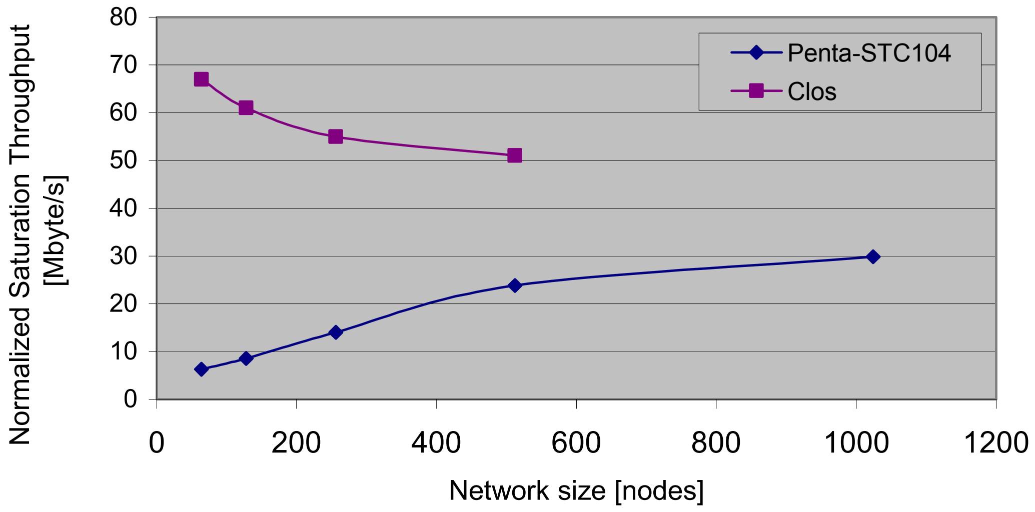


Figure 6-7: Normalized throughput for different sizes of Penta-STC104 and Clos networks under random traffic

It is clear from figure 6-7 that the normalized throughput for Clos network is better than Penta-STC104, this refers to the fact that the Clos network contains a central stage, this stage increase the number of links between the stages, i.e. there exist multiple parallel links between stages.

6.5.4 Network Latency

Network latency is defined as the delay from the transmission of the packet header at the source to the arrival end of packet at the destination. This section presents the average latency as function of network loads; also examine the effect of packet length on latency. As mentioned earlier, each time we run the program we obtain the throughput and its corresponding packet latency.

6.5.4.1 Average Network Latency

To test the average network packet latency, we ran the program for all packet values (64,128, ….,1024 bytes) under the loads (30%,40%,50%, and 60%) three times; two for number of active nodes = 256 and 512 respectively, and the third for 1024 nodes. For each load, the average of the latency obtained over all packet length is taken and plotted against the load.

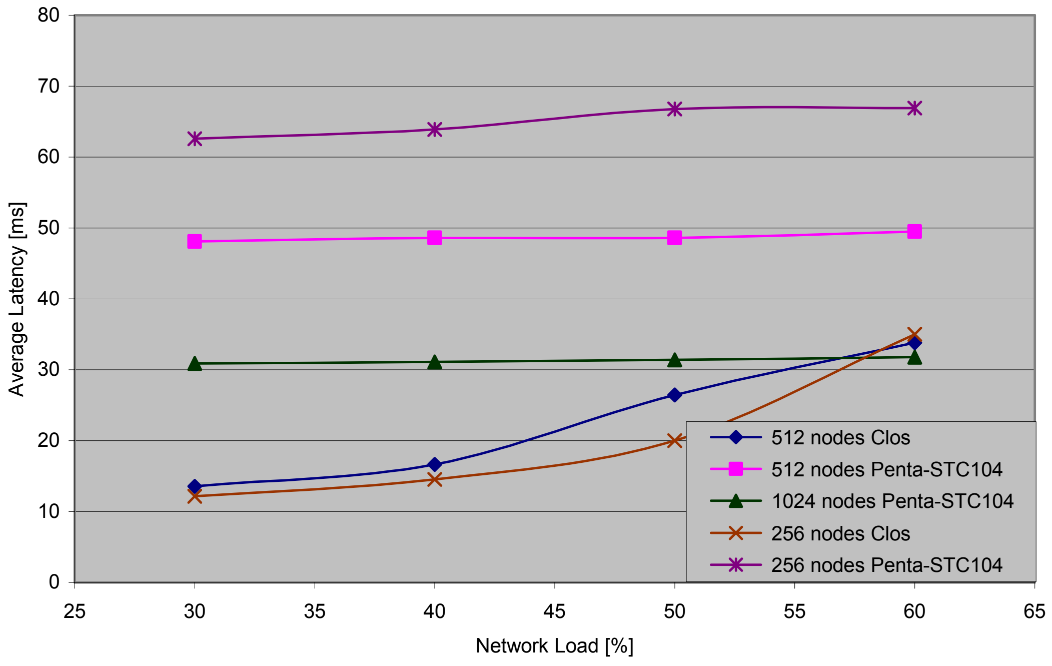


Figure 6-8: Average latency versus network load for different size Penta-STC104 and Clos networks

Figure 6-8 shows the average latency of three different sizes of Penta-STC104 and Clos networks versus the applied network load.

It can be seen that the average latency for Penta-STC104 is nearly stable versus the loads. This means the load value has a little effect on the Penta-S latency. This is because to send any packet between two different nodes at any load first need to send header to established the link then send data directly between nodes. However the latency in Clos network increases when the load increase until the saturation throughput. After that, it stays constant when the network is saturated. This is because all the queues are full and the link flow control prevents new packets from entering network, while the path is blocked. It is worth noting that the Penta-S latency decreases as the network size increases. This is due to the extra links that become available and the better load balancing associated with the Penta-S network growth.

6.5.4.2 Effect of Packet Length on Latency

To test the effect of the packet length on packet latency, the program was run for various packet lengths (64,128,256,512, and 1024 bytes under loads 30%, 40%, and 50% for number of active nodes = 512 nodes. The packet latency values were obtained and plotted against the corresponding packet lengths.

Figure 6-9 shows the effect of packet length on the average latency for a 512 node Penta-S and Clos networks under random traffic for different load values. The curves obtained are compared with Clos results in Macramé project under the same conditions. Both Clos and Penta-S network latency grows with packet size. However, for Clos, the latency grows with the load % with clear differences, but For Penta-S the latency grows with the load % but with small differences. These differences tend to vanish at packet size equal 1024.

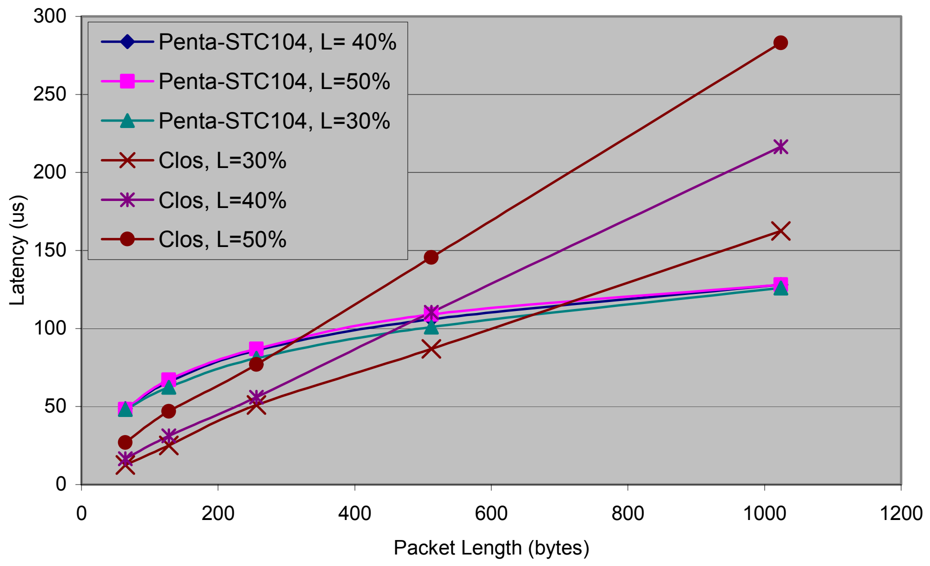


Figure 6.9: Latency versus packet length for various load values for 512 node Penta-S and Clos networks.

At load 30% the latency of Penta-S starts becoming smaller than that of Clos at packet length greater than or equal (>=) 675 bytes. At load 40% the latency of Penta-S starts becoming smaller than that of Clos at packet length greater than or equal 509.4 bytes. At load 50% the latency of Penta-S starts becoming smaller than that of Clos at packet length greater than or equal 328 bytes.

The above results are explained by the fact that large packets create congestion in Clos network, where as in Penta-S, there is a large buffer with each node that stores the coming packets from the shuffle in addition to the fair arbitration between the shuffle packets and the node originated packets.

6.6 Performance of NBWR-based Penta-S

After comparing Penta-S built of STC104 with Macramé, this section presents the throughput and latency for proposed switch NBWR (Non-Buffered-Wormhole-Routing) as compared with Penta-STC104. So, we do not need to compare the results with those obtained from Macramé project because comparing the NBWR-based Penta-S with that of the STC104-based Penta-S is enough to give us an idea about the performance of NBWR-based Penta-S

6.6.1 Throughput of NBWR-based Penta-S

In order to obtain a sample the throughput of Penta-S built of NBWR switches, we ran the program for several packet length values (64,128, 256, 512, and 1024 bytes) under load 64% for 512 nodes. The test was repeated under the same conditions for STC104-based Penta-S. Figure 6-10 shows the network throughput for Penta-NBWR and Penta-STC104 at size 512 nodes under random traffic for different packet lengths. In general the maximum throughput for Penta-S occurs at 128 byte packet length and reaches to 1290 Mbyte/s for Penta-NBWR and 1230 Mbyte/s for Penta-STC104.

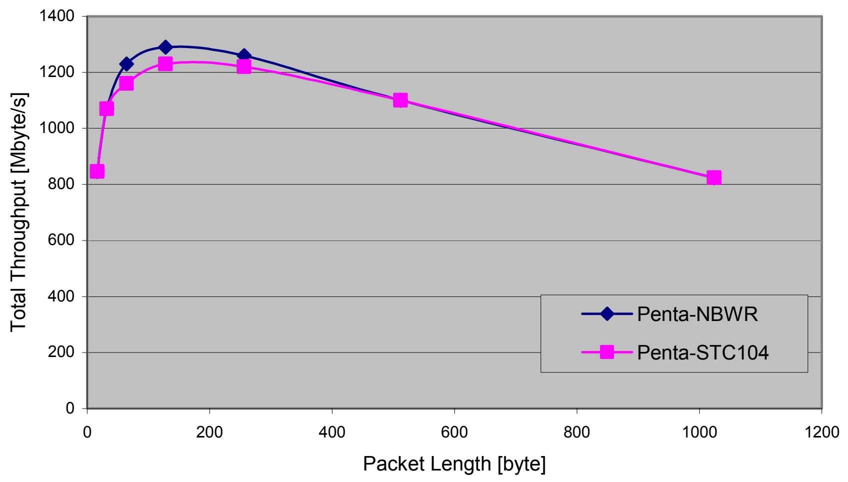


Figure 6-10: Total throughput for NBWR at size 512 nodes and different packet sizes

The NBWR Penta-S has a better throughput for packet lengths between 64 and 512 bytes. For the rest of the packet lengths the two schemes give the same results. The difference appears in the building of these schemes, the NBWR is simpler in design than STC104 as discussed in chapter 4.

6.6.2 Latency of NBWR-based Penta-S

Results produced from tests discussed in section 6.5.1 are used to show the latency of Penta-S. Figure 6-11 shows the latency of Penta-NBWR and Penta-STC104 for different packet length and load 64%. The network size is 512 nodes as obtained from the previous test. It can be seen that the latency always increases as the packet length increases.

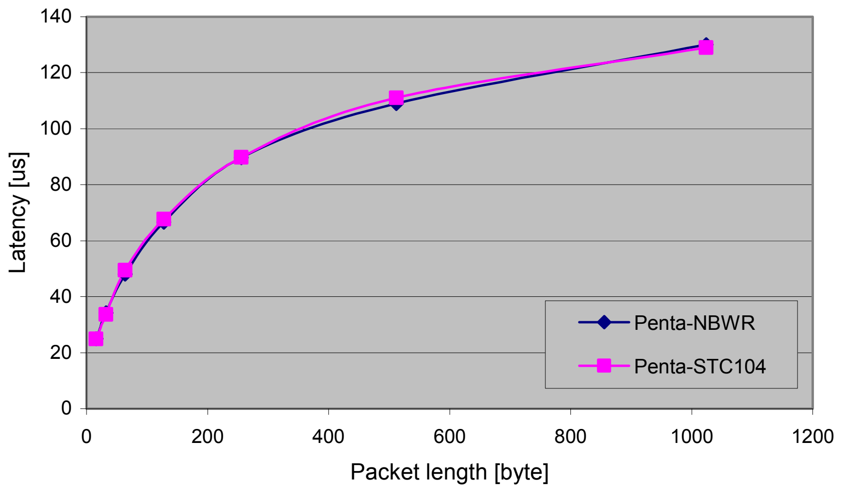


Figure 6-11: Latency versus packet length for 512 nodes Penta-NBWR and Penta-STC104

The minimum achievable latency is 25 μs at packet length 16 byte. And the maximum latency occurs at packet length 1024 byte and recorded as 130 μs.

6.7 The Cost and Cost-Efficiency

In this section the cost of Penta-S, Clos and 2D-Grid networks in terms of the number of (32X32) crossbar switches used in building various sizes of theses networks. Calculating the cost allows us to calculate the cost-efficiency according to equation 5.6 by dividing the throughput by the cost.

6.7.1 Comparing the Cost of the Network Topologies

This section presents the benefits of using Penta-S over using other expansion topologies. These benefits appear in the easiness of building this scheme and its modularity. Also in the small numbers of switches needed to build this scheme compared to others topologies and in the easy routing algorithm of this scheme. One also can note that on the opposite of Clos and 2D-Grid topologies, the individual (32X32) switches of the Penta-S need not to be in the same place, rather they can be distributed in various places (within the allowed LAN distances) and connected via cables to the shuffle link. The shuffle link will be placed in a central position with respect to the switches. Table 6-1 present the comparison between Penta-S, the 2D-Grid, Torus and Clos networks.

Table 6-1: Number of switches for Penta-S, Clos, 2D-Grid and Torus

Number of nodes	64	128	256	512	1024
Network Type	Number of (32X32) Switches needed				
2D-Grid and Torus	4	8	16	32	64
Clos	6	12	24	48	96
Penta-S	2	4	8	16	32

From the previous table it is clear that the Penta-S uses a small numbers of switches compared with other topologies.

6.7.2 The Cost-Efficiency

This cost-efficiency means the network saturation throughput divided by cost (here, cost is measured in terms of the number of (32X32 switch modules). In order to obtain the cost efficiency, the results obtained in section 6.5.1 are divided by the corresponding cost in table 6.1 Figure 6-12 shows the cost-efficiency of STC104-built Penta-S, Clos, and 2D-Grid networks versus the number of nodes.

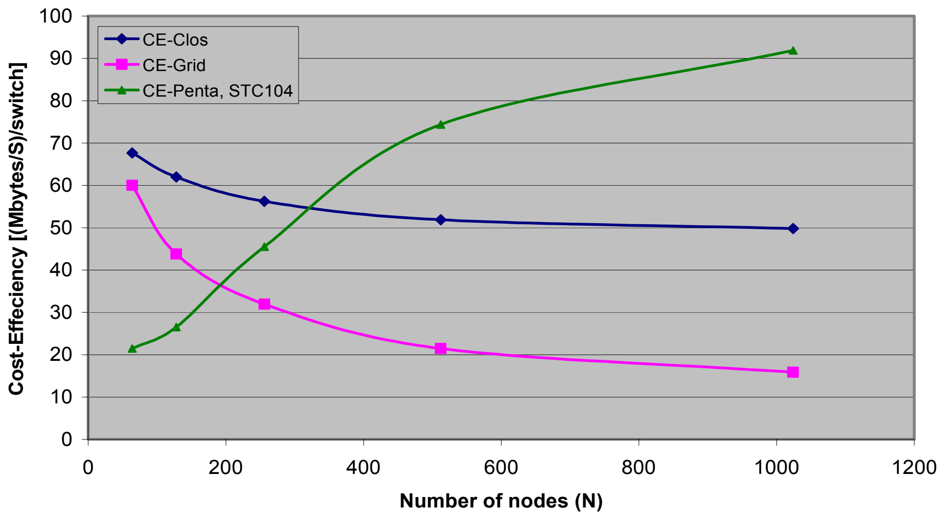


Figure 6-12: The cost-efficiency of STC104-built Penta-S, Clos, and 2D-Grid networks versus the number of nodes

It is clear from the figure that Penta-S network starts having better cost-efficiency than the 2D-Grid for number of nodes > 200, and better than Clos for number of nodes > 350. On the opposite of other networks, the cost-efficiency of Penta-S grows as the network size grows. It can be concluded that Penta-S has a better position than other networks as larger networks are needed.

Chapter 7

Conclusion

In this work, a simulator for the proposed Penta-S computer network topology was built to simulate its behavior under various traffic conditions. A well established project called Macramé project was carefully analyzed and understood and a well known, widely used crossbar switch also was investigated and examined. This was to achieve the goals and the objectives stated in chapter 1.

7.1 Achievements

Through simulating the proposed topology under various traffic conditions, the position of this topology among other topologies was defined to a good degree of accuracy. The features of STC104 that suit or disagree with this type of network topology was found and new simple crossbar design with wormhole routing capability was proposed. As explained in chapter 4, the proposed switch is simpler, cheaper and faster than the STC104 switch.

A simulator that can be modified for future work was designed to simulate this topology. This simulator can be modified by nesting the loop of stages 2 to 5 of its algorithm in a bigger loop to measure the performance of a multi-layer version of this network (this network can be used as a layer in a further multilayer expansion). A mathematical model derived (See Appendix E) and the per-node throughput was plotted and compared with the results obtained from the simulator. The results of the simulator match with the theoretical prediction.

The performance (throughput, latency, and cost efficiency) of this network was measured under various network sizes, packet lengths and loads to determine where best it can be used and to compare it with other topologies. Extensive simulation tests and measurements have shown that Penta-S suits large networks, long packets and heavy loads. Its latency tends to be stable for long packets, and decreases as the load increases. Also it cost efficiency becomes better for large number of nodes.

As compared with other topologies it costs less, more modular and benefits from every single link of the switch used as a building block. Note that 32 nodes can be connected to the (32x32) STC104 switch in Penta-S, whereas only 16 out of the 32 links of the STC104 switch can be used in Clos and the 2D-Grid network topologies. Also it is easy to use it in integrating the switches distributed in a LAN area. Each switch of this network can be residing in different site and cables (multi-fiber cables of 96 hairs are available in the market) from all site are connected to a central shuffle, whereas in Clos network, the whole body of the network (all the switches) must reside in one place. This feature makes it easy to add switches to Penta-S without altering the design.
Regarding the scalability, the results discussed in section 6.4.1 show that the network is scalable at a rate equivalent to that of Clos network (The growth of the throughput versus the network size has the same slope for both network for number of nodes > 250). It also has a better slope of scalability than that of the 2D-Grid network. The linear slope of the network shown in figure 6-4 implies that this network is aggregate.

7.2 The Future Work

It is right that the number of switch modules this topology can integrate is limited to n+1 modules where n is the number of nodes that the switch can accommodate, but its expansion is easy. Multi-layer versions of this network can be designed easily. One can think of two things to expand this topology further:

1. The nodes extra switch module (the n+1th) module in each layer can be used as clients for up to 32 layers. Note that using (32x32) crossbar switch as a building block, a Penta-S which can accommodate [(32+1) blocks x 32 nodes = 1056] nodes. By using all the node places in the 33rd block in each layer as clients for other layers, one can build 33 layers, each of which can accommodate 1024 nodes, i.e., a total of (33x1024) nodes (computers) can be connected.
2. Little modifications on the simulator of this project make it capable of simulating the multi-layer Penta-S.

List of References

[Andr 01] Kimara Andreas," National Texas Instruments, A White paper on KVM Switch Solutions", White paper/nti_white_paper_2001.doc (1 of 38) [March 1,2001].

[Ayya 04] A. Ayyad, "Prnta-S: A Scalable Interconnection Scheme for Distributed Shared Memory Multiprocessor Systems", to appear in the proceeding tand the Transaction Journal of the 6th WSEAS Conference on Scientific Computing, Simulation and Modelling, 12-15 May 2004, Cancun, Mexico.

[Buhr 98] Magnus Buhrgard, Tommy Westin and Goran Wicklund," A scaleable ATM switching solution", Ericsson Review No 1, 1998.

[Collier 94]M. Collier and T. Curran, "The strictly nonblocking condition in three-stage networks," *Proceedings of the 14th International Teletraffic Congress*, Antibes Juan-Les-Pins, 6-10 June 1994. (Two typos in the inequalities in Section 4.3 of the version of this paper in the ITC-14 proceedings are fixed here.).

[Gram] M.D. Grammatikakis, E. Fleury, and M.Kraetzl,: Continuous Routing in Packet Switches", International journal of Foundation of Computer Sciences, World Scientific Publishing Company, pp.1-17.

[Haas 98] Stefan Haas, **"**The IEEE 1355 Standard: Developments, Performance and Application in High Energy Physics", PhD thesis, Liverpool University, 1998.

[Heel 96] Roger Heeley. "Real Time HEP Applications using T9000 Transputers, Links and Switches", PhD thesis. Liverpool University, 1996.

[Hwa 93] Kai Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability", McGraw-Hill Inc., N.Y, U.S.A,1993.

[IDT-DB 92] “Specialized Memory and Modules Data Book”, Integrated Device Technology, Inc. (IDT), 1992.

[Khan 01] Mohammad Hossein Khansari, Ali Movaghar-Rahimabadi and Shaahin Hessabi " Design and Simulation of a Tandem ATM switch based on FPGA , Dept. of computer Engineering, Sharif University of Technology, Azadi Ave.Tehran, Iran IST 2001 Tehran-Iran 1-3 Sept 2001.

[Lys 99] O. Lysne. Deadlock Avoidance for Switches based on Wormhole Networks, Proceedings of the 1999 International Conference of Parallel Processing, Aizu-Wakamatsu (Japan), pp 68-74, IEEE Computer Society, 1999.

[Mud 01] Muhammed Mudawwar and Aya Saad, “The k-ary n-cube Network and its Dual: a Comparative Study”, Proceedings of the 13th IASTED International Conference on Parallel and Distributed Computing and Systems, August 2001.

[NI 01] National Instruments," Matrix Switch Expansion Guide", National Instrument Corporation. June 2001, Application Note 174.

[Patt 03] Achille Pattavina and Claudio Catania,” Performance analysis of ATM Replicated Banyan Networks with External Input-Output Queueing”, Telecommunication Systems 23:1,2, PP. 149-170, 2003.

[Sapo 03] Georgios Sapountzisy and Manolis Katevenisy, “Benes Switching Fabrics with O(N)-Complexity Internal Backpressure”, IEEE, Proc. of Workshop on High Performance Switching and Routing (HPSR), June 2003, pp. 11-16. 11.

[SGS-TH 96] “STC104 Data sheet”, SGS-THOMSON Microelectronics, Sheet No.42 1470 07, Italy, 1996.

[shi 01] Hongyuan Shi and Harish Sethu, “Virtual Circuit Blocking Probabilities in an ATM Banyan Network with *bXb* Switching Elements”, (part of Advanced Simulation Technologies Conference) Seattle, Washington, USA, April 22–26, 2001.

[Sten 88] P. Stenstrom,” Reducing the contention in Shared memory Multiprocessors”, IEEE Computer, V. 21, No. 11,Nov, 1988, PP 26-37. Proceedings of the Applied Telecommunication Symposium.

[Sven 01]: Sven-arne Reinemo*, Olav Lysne and Tor Skeie ,"Topologies and routing in Gigabit switching fabrics" , Departement of Informatics, University of Oslo, Box 1080 Blindern, N-0316 Oslo, Nrway, November 2001.

[Tan 96] Andew S. Tanenbaum, "Computer Networks", Prentice Hall PTR, Upper Saddle River, New Jersey 07458,1996.

[Thak 98] Purvesh B. Thakker," Survey of Switch Architectures, University of Illinois at Urbana-Champaign, (1 of 27) [December 11, 1998].

[Thomp 94] P. Thompson and J. Lewis. “The STC104 Packet Routing Chip”. *Journal of VLSI Design*, vol. 2, no. 4, pp. 305-314, 1994.

[Thomp 97] P.W. Thompson and J.D. Lewis, "The STC-104 Packet Routing Chip", SGS-Thomson Microelectronics Limited, Bristol, England, 1997, pp.1347-1356.

[Tuts 02] Tutsch, D., Hommel, G.: Comparing Switch and Buffer Sizes of Multistage Interconnection Networks in Case of Multicast Traffic. *Proceedings of the High Performance Computing Symposium 2002 (HPC 2002).* San Diego, USA, 2002, pp. 300-305.

[Vir 97] Rahul Vir, "LAN switching", http:// www.cis.ohio-state.edu~jain/cis788-97/lan_switching/index.htm(1 of 20) [2/7/2000 10:58:36 AM].

[Wilk 96] Barry Wilkinson, "Computer Architecture; Design and Performance", 2th edition, Prentice Hall Europe, Hertford shire, U.K, 1996.

[Webo] http://www.webopedia.com.

[Yang 99] Yuanyuan Yang, Gerald M. Masson: The Necessary Conditions for Clos-Type Nonblocking Multicast Networks. IEEE Trans. Computers 48(11): 1214-1227 (1999).

Appendix A

An Overview of the IEEE 1355 Standard [Haas 98]

The IEEE 1355/1995 standard (ISO/ICE 14575) of scalable heterogeneous interconnect, defines the physical implementations and logical protocol layers for point to point serial interconnect, operating at speed ranges from 10-200 MBaud in copper and up to 1 GBaud in fiber optic technologies.

The IEEE 1355 standard enables the construction of low-cost, high-speed scalable interconnects. Although this technology has been initially designed as a multiprocessor interconnects, it is equally appropriate for applications in communication systems and local area networks. It also allows the transparent implementation of a range of higher level protocols such as ATM, and Ethernet. The routing techniques discussed in chapter 4, like adaptive grouping, Universal routing, and wormhole routing are part of this standard.

Some potential application areas for IEEE 1355 links and switches are listed below:

- LAN switching hubs
- Parallel computers
- Data acquisition systems
- ATM switching
- Industrial control systems
- Multimedia servers

Appendix B
Packet Header Deletion

Header deletion allows networks to be connected together, as shown in figure 3.8. In this example a packet is routed through two networks and then to a processing node. All of the terminal links of the two networks are set to header deletion mode. Figure 3.8 shows the header as it is routed through the network

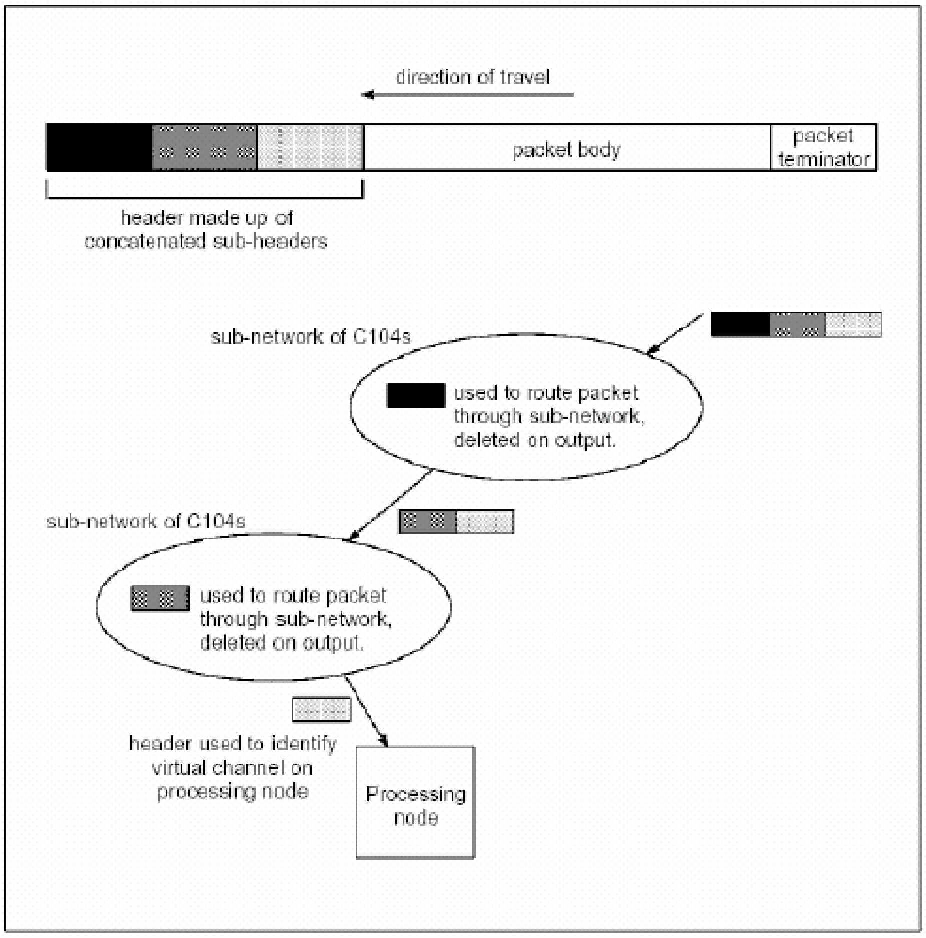
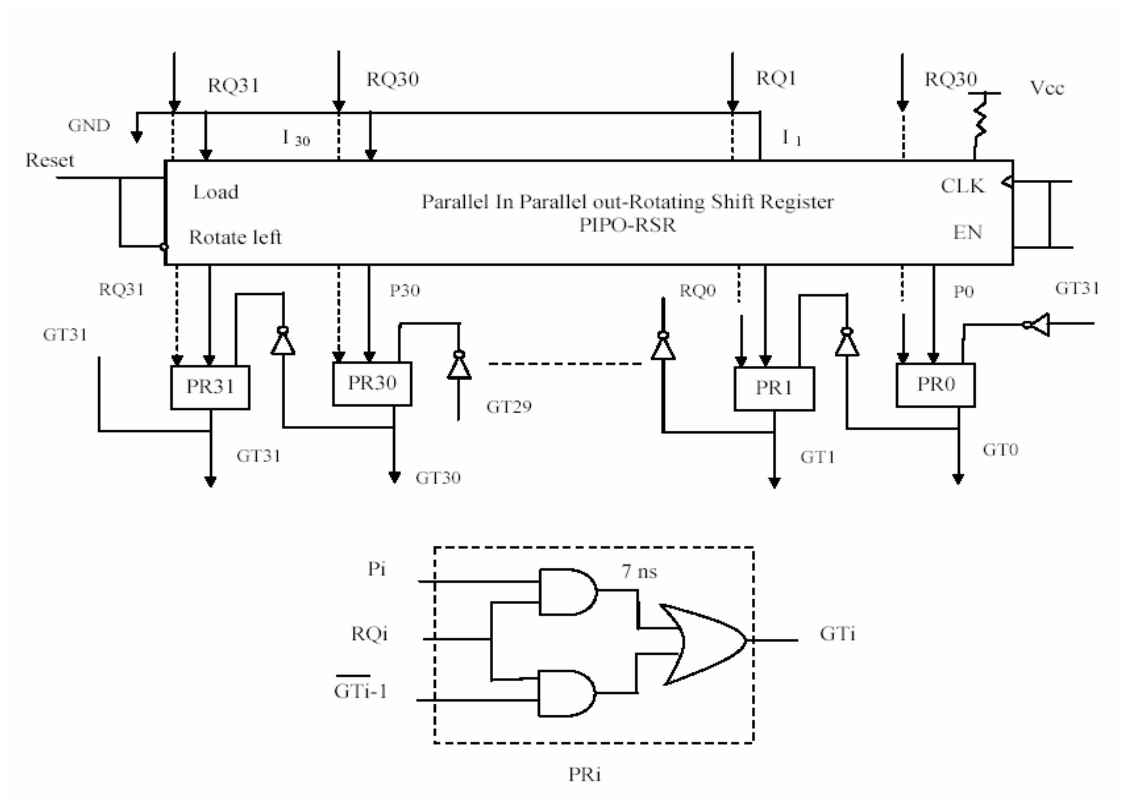

direction of travel
packet body
packet terminator
header made up of concatenated sub-headers
sub-network of C104s
used to route packet through sub-network, deleted on output.
sub-network of C104s
used to route packet through sub-network, deleted on output.
header used to identify virtual channel on processing node
Processing node

Figure B-1: Hierarchical composition of networks using header deletion

The header of the packet in this case is made up of three concatenated sub-headers. The first sub-header routes the packet across the first network and is deleted as the packet leaves the terminal link of the network. The second sub-header routes the packet across the second network in the same way. Finally the third header is exposed to identify the destination virtual channel on the processing node. This can be applied to hierarchically constructed networks, in which case the sub-headers are similar to the local/national/international hierarchy of telephone numbers. In the case in which each STC104 is treated as a separate network and has its link outputs set to header deletion mode, packets can be explicitly steered across a network. This is at the expense of having 1 byte of header for each STC104 traversed. A major advantage of extending the capabilities of the STC104, through header deletion, is that headers can be minimized for small systems, thus optimizing network latency and network bandwidth, whilst still enabling more complex, larger, systems to be constructed efficiently.

Appendix C

The Arbiter Circuit



t_{PD} for f/f (shift Register)= 25 ns

Max propagation of GT through 32 priority resolver =15X31

=465ns

tpd HCT 08 =7 ns tpd HCT32=7.5

Notes:

PR= Priority Resolver

P= Priority

RQ= Request

GT= Gant

Total sampling (settling time)= 465+25=490 ns.

Appendix D
Poisson Mathematical Model

1. **Mathematical Model:** When arrivals into a queuing system are independent (i.e. the fact that a person or party enters the shop during a given minute has no effect on whether another party enters the shop during any other minute), the arrival process is said to be a Poisson process, and the time between arrivals is a random variable which follows the Exponential distribution. This distribution is described by the following probability distribution function:

   $f(t) = \lambda e^{-\lambda t}$

   ${}^{1}/_{\lambda}$ = average inter-arrival time

   The same distribution will be used for the service time (the time from the moment the customer's order is taken to the moment the order is filled), but it is customary to employ μ (instead of λ) for the service rate:

   $f\,(t) = \mu e^{-\mu t}$

   ${}^{1}/_{\mu}$ = average time to service an order

2. **Computational Model:** There are many ways to implement a Monte Carlo simulation of the working problem described above. Almost all general purpose programming languages (e.g. C/C++, Java, BASIC, FORTRAN), as well as a number of commonly-used scripting languages (e.g. JavaScript, VBScript) and application-hosted languages (e.g. Excel with VBA) can be used successfully. (There are special-purposes languages as well, specifically

designed for such simulations; we will not be using any of these.) The usual approach is to program the simulation as a "discrete event simulation", where a variable is used to keep track of the current simulation "clock", and a pseudo-random number generator is used to calculate the times of the upcoming simulation events (e.g. customer arrival, completion of an order for one of the customers currently being served). At each iteration of the simulation, the clock is advanced to the time of the earliest upcoming event, the simulated statistics (e.g. average waiting line length, average waiting time, etc.) are updated, and the simulation state is modified as necessary (e.g. length of the waiting line increased by one when a customer arrives).

Most general-purpose programming languages provide variations on a very simple pseudo-random number generator, which generates numbers from the Uniform distribution, on the interval [0,1). Since our simulation will need to generate numbers from the Exponential distribution, we will need to transform the numbers returned from the Uniform distribution, as follows:

$t_{arrival} = -\ln(1-u_{arrival}) / \lambda$

$t_{service} = -\ln(1-u_{service}) / \mu$

$u_{arrival}$ = number in interval [0,1], returned from uniform pseudo-random number generator, for arrival time

$u_{service}$ = number in interval [0,1], returned from uniform pseudo-random number generator, for service time

Appendix E

Mathematical Model of Penta-S

Penta-S; A mathematical Performance Analysis

The behavior of Penta-S is described in the thesis. The following assumptions represent the basis for deriving the mathematical model:

1. The building block of the network is as n x n crossbar switch module (here n=32).
2. K modules are used, where K can have a maximum value = n +1.
3. All nodes are equally likely to be addressed by any node. So if K modules are used, then in each module we have $\frac{1}{K}$ requests local and $\frac{K-1}{K}$ requests global.
4. Full load is assumed L%=100%.
5. Packet cycle length=Number of packet bits x system clock period.
6. Probability of a packet arrived during the above cycle r = λ x Packet length; in this case r =1.
7. Probability of acceptance of n x n crossbar switch with r =1;

   $$P_a = 1-\left(1-\frac{1}{n}\right)^n \text{; in this case}$$

   $$P_a = 1-\left(1-\frac{1}{32}\right)^{32} = 0.638$$

8. The rejected requests are not discarded; rather they wait in a queue for their turn to be transmitted.

The final equations of the model are as follows:

1. Number of expected nodes making new requests at any cycle i

$$\text{Ni} = \text{n} - (W_{l(i-1)} + W_{g(i-1)})$$

n= the number of nodes connected to each switch module.

$W_{l(i-1)}$ = number of requests waiting for transmission to local destination from the previous cycles.

$W_{g(i-1)}$ = number of global requests waiting to be served from previous cycles.

2. Number of local packets served by the switch in the ith cycle

$$T_{rli} = W_{l)i-1)} + \frac{p_a}{K} N_i$$

Number of global packets served in each cycle

$T_{rgi} = P_a(K-1)$; K -1 global link through the shuffle are available.

3. $W_{li} = \frac{(1-P_a)}{K} N_i$

$$W_{gi} = (K-1)\left[\left(\sum_{1}^{i} \frac{N_i}{K}\right) - iP_a\right]$$

The number of cycles needed to complete serving the waiting global packets

$$X = \frac{1}{p_a^2}\left(W_{gi} - \frac{P_a}{K}\right) + 2$$

The bandwidth BW in packets served by the switch

$$BW = XP_a(K-1) + \sum_{i=j}^{j+x} Trli \quad \text{; Where j is the current cycle.}$$

Per-Node throughput

PNTP = $\left(\frac{BW}{X}\right)/32$ in (Packet/node/packet cycle)

$$\text{PNTP}=\frac{BW}{32}\frac{(pl/8)}{pl*10^{-8}}=\frac{BW}{256}*10^{8}\ \text{byte / sec.}$$

$$=\frac{BW}{256}*10^{2}\ \text{Mbyte / sec.}$$

10^{-8} is the clock period of 100 MHZ clock used in the simulator.

The mathematical and simulation results of PNTP are plotted in the following figure.

With reference to Penta-S topology, the following:

K **Number of Blocks**	**No of transferred** **Packets**	**No Packet cycles** **Elapsed**	**PNTP**
2	49	39.940	0.47
4	36.074	20.87	0.675
8	33.495	11.435	1.178
16	34.411	6.717	2.17
32	32	4.359	2.868

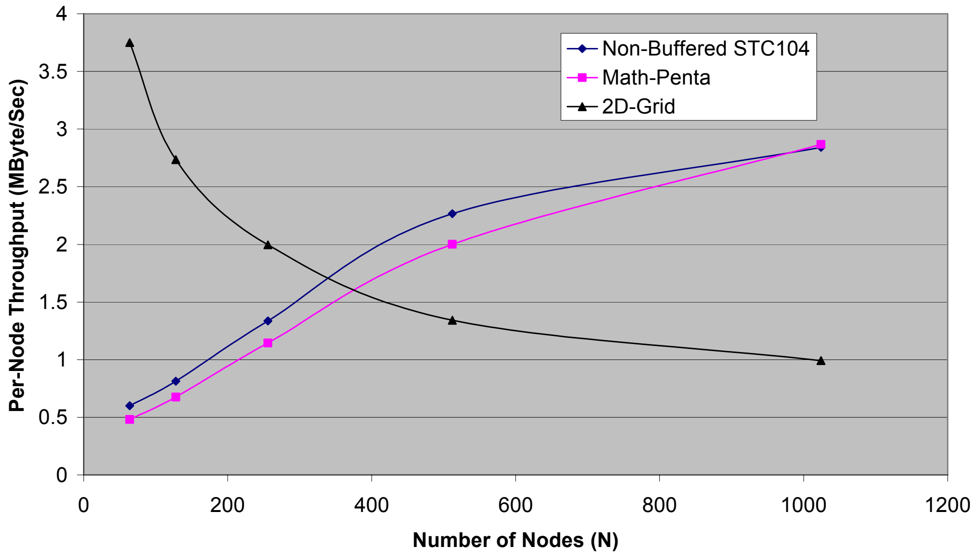


Figure E-1: The per-node throughput of Penta-S as taken from the mathematical model (Math-Penta) as compared to the simulation results of Penta-S built of STC104 switches. The per-node throughput of 2D-Grid is shown also

Note: This mathematical model was derived by Dr. Abdulkarim AYYAD, Computer Engineering Department, Al-Quds University.

Appendix F

The Simulator UML Model

(Important Classes)

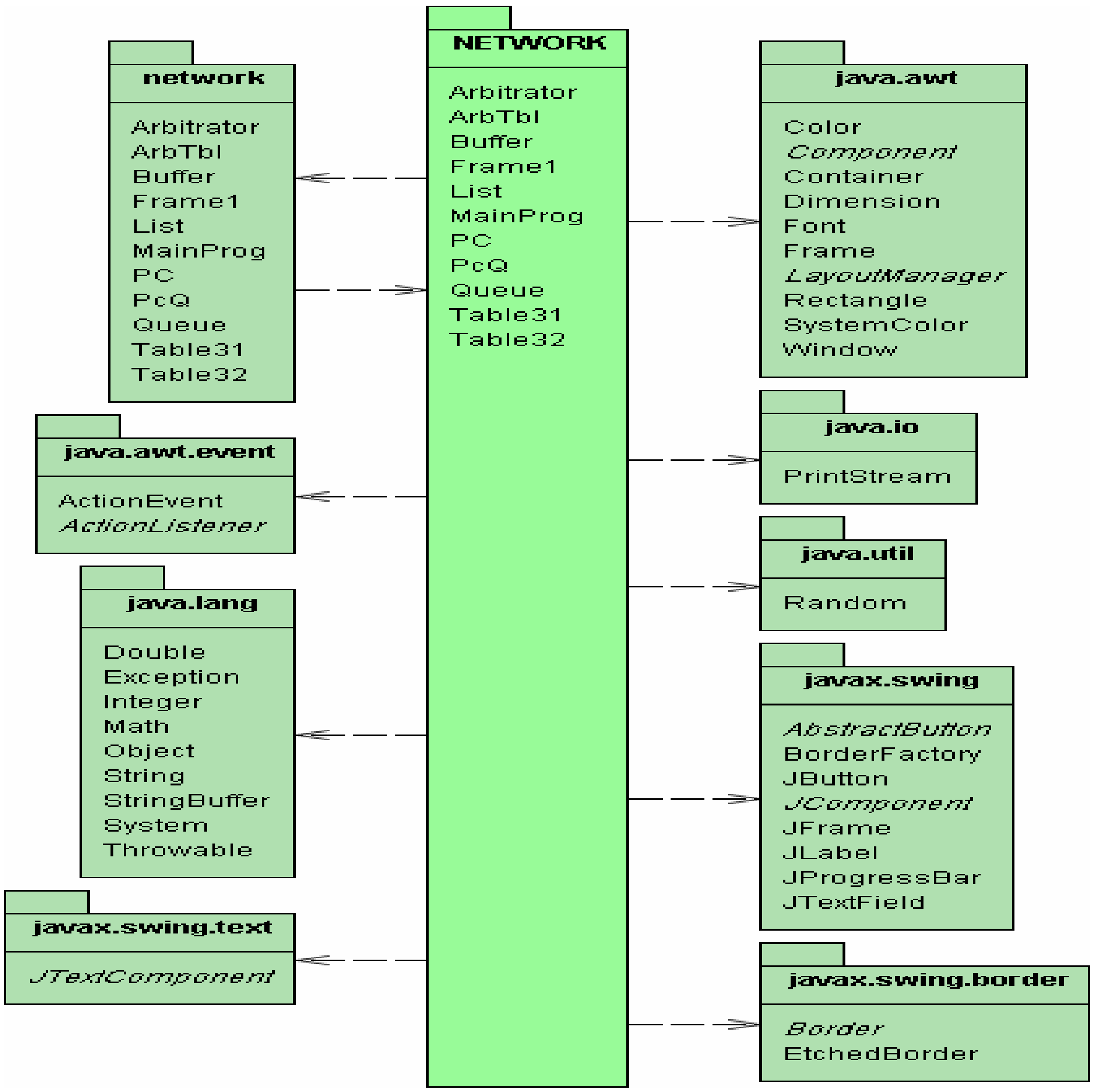


Figure F-1 Simulation Package

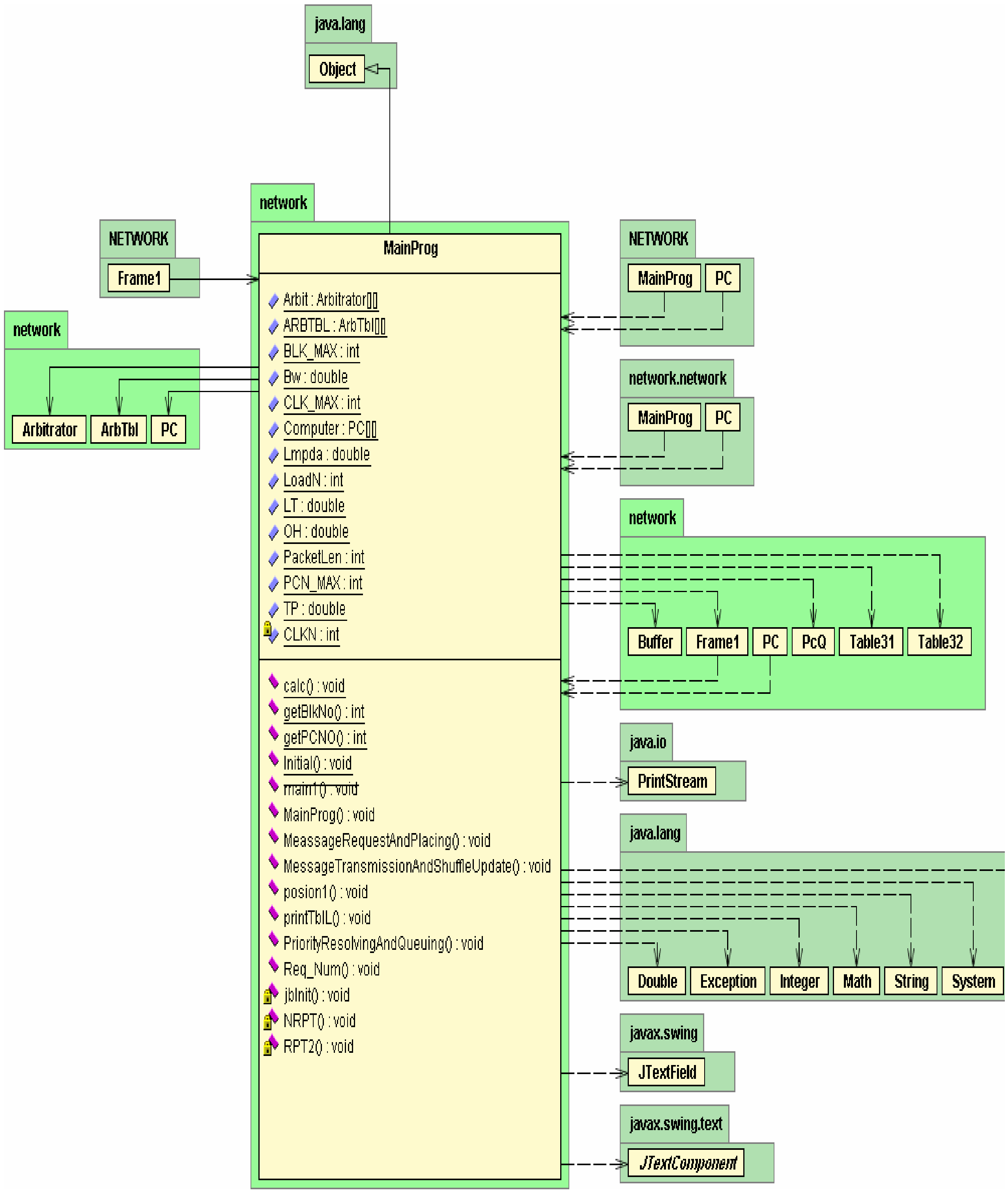


Figure F-2 Main-program Class

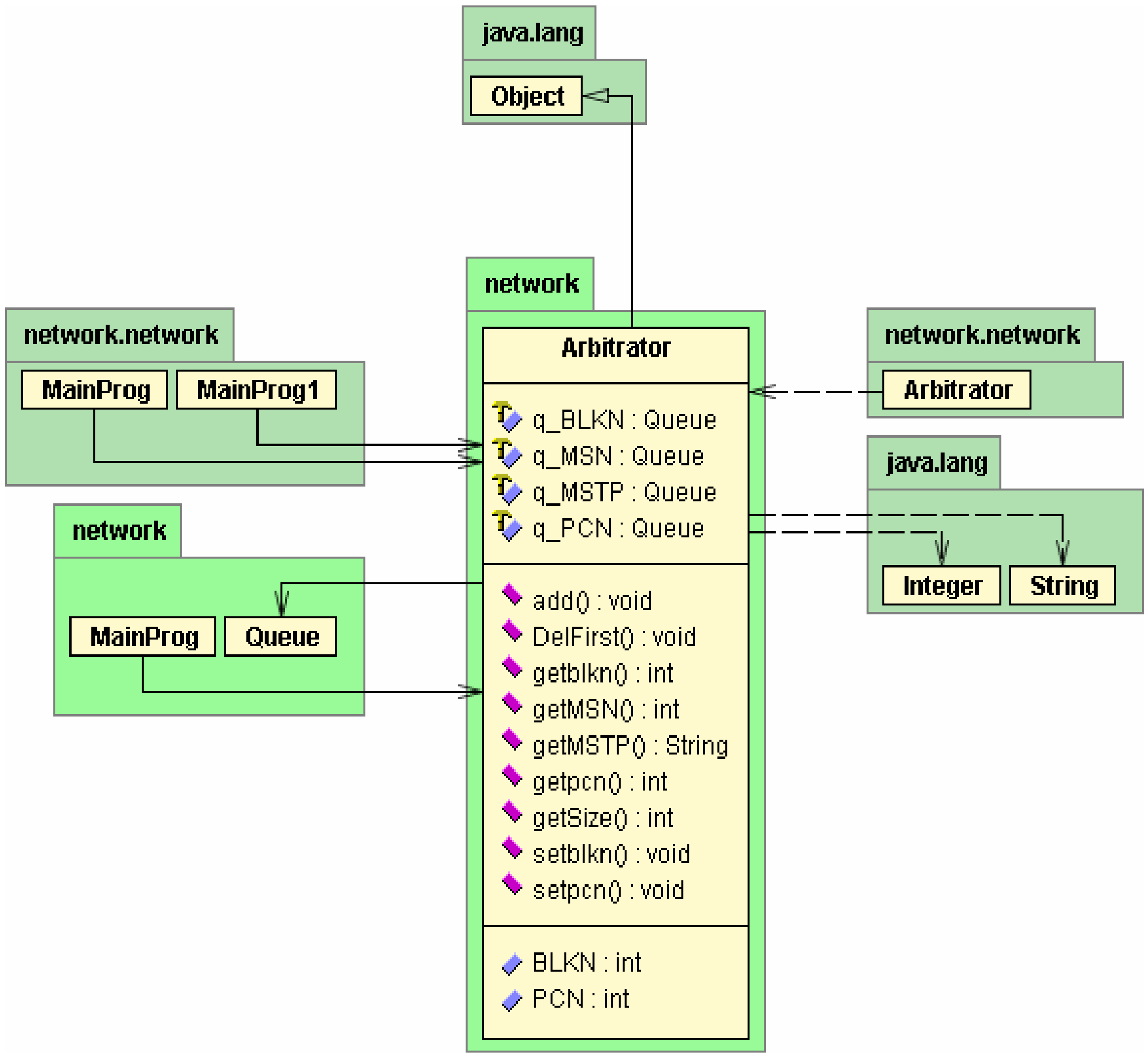


Figure F-3 Arbitrator List Class

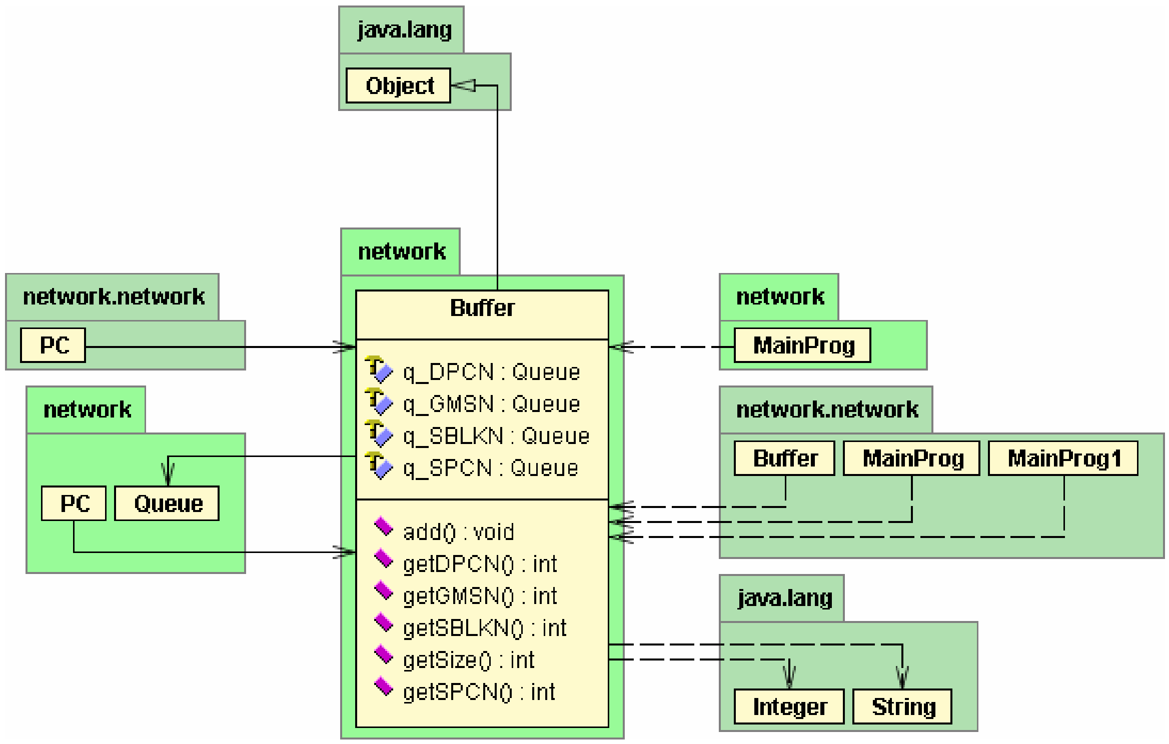


Figure F-4: Shuffle-Buffer Class

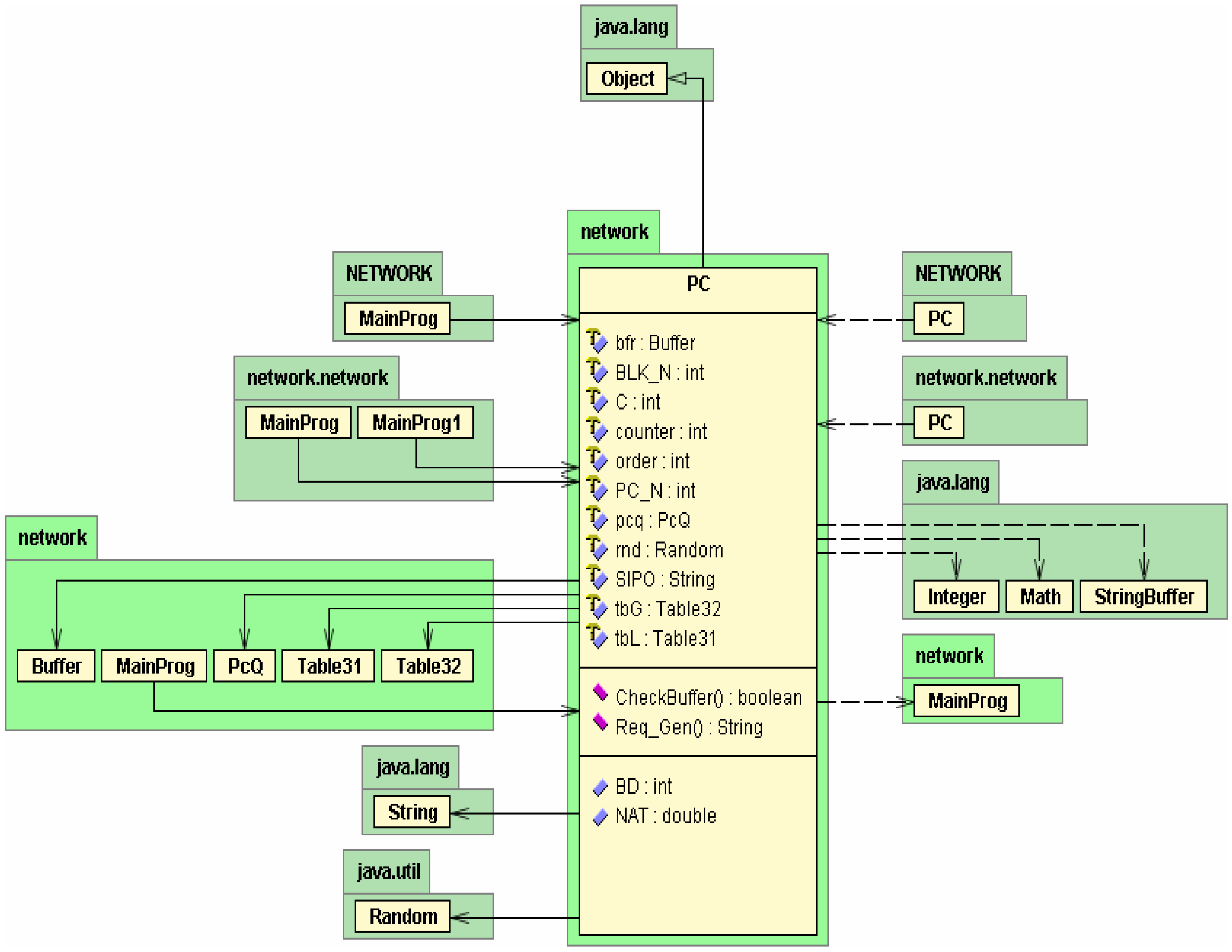


Figure F-5: PC Class

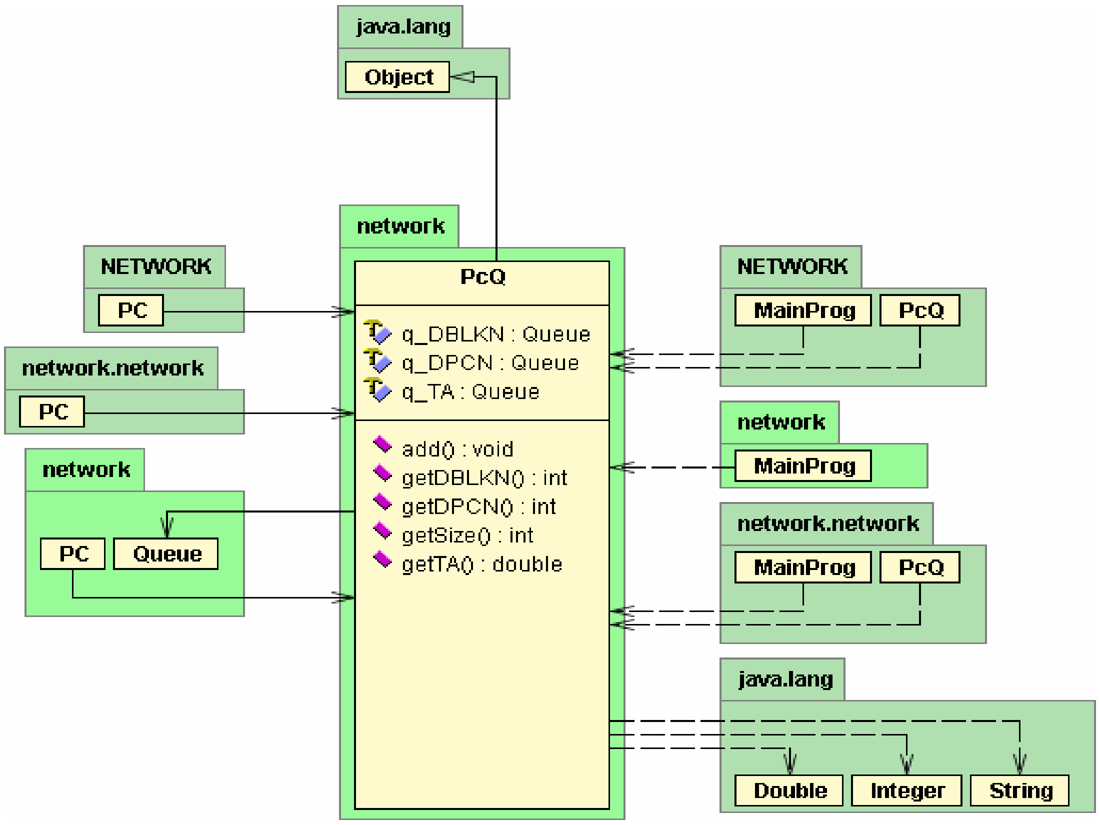


Figure F-6: PC queue Class

ملخص

في هذة الرسالة تم تشبيه أداء عمل توبولوجيا جديدة لشبكات الحاسوب تسمى Penta-S. هذه الشبكة الجديدة مبنية من وحدات مفتاح تقاطعي (Crossbar Switch). ويتسع كل مفتاح لاثنين وثلاثين محطة حاسوب. وكل واحدة من هذه المحطات مزودة بميناءين (Ports)، الأول يصلها مع وحدة المفتاح التقاطعي والثاني مع محطة مراسلة لها في وحدة مفتاح أخرى عبر وصلة إزاحية (Shuffle link). وقد تم فحص أداء هذه الشبكة بواسطة برنامج تشبيه تحت ظروف مختلفة من حجم الشبكة وطول الظرف الحامل للمعلومات (Packet) والحمل (Load). تم مقارنة النتائج التي تم الحصول عليها مع نتائج مشروع Macramé لشبكات Clos و 2D-Grid. كانت أعلى إنتاجية (تدفق معلومات) (Throughput) لشبكة Penta-S على طول ظرف 128 بايت. كما لوحظ أن إنتاجية هذه الشبكة تنمو خطيا مع حجم الشبكة. وعلى العكس من شبكات Clos و 2D-Grid فإن إنتاجية المحطة الواحدة تنمو مع زيادة حجم الشبكة، أما بالنسبة لمعدل الوقت الذي يأخذه ظرف المعلومات للوصول إلى محطته النهائية (Latency) فقد ثبت أنه لشبكة Penta-S أفضل منه لشبكة Clos لظروف المعلومات ذات الحجم الكبير وكذلك للأحمال الكبيرة. كما أن هذا الوقت Latency يميل إلى الثبات مقابل الأحمال المختلفة على شبكة Penta-S . وبذلك نستطيع القول بأن هذه التوبولوجيا Penta-S مناسبة للشبكات الكبيرة والأحمال العالية.