

2003

Design and analysis of traffic shaping and scheduling in a fast switch network

A. Alfred Joseph Jude
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_theses

Recommended Citation

Jude, A. Alfred Joseph, "Design and analysis of traffic shaping and scheduling in a fast switch network" (2003). *Master's Theses*. 2475.
DOI: <https://doi.org/10.31979/etd.veng-bmc5>
https://scholarworks.sjsu.edu/etd_theses/2475

This Thesis is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Theses by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

DESIGN AND ANALYSIS OF TRAFFIC SHAPING AND SCHEDULING
IN A FAST SWITCH NETWORK

A Thesis

Presented to

The Faculty of the Department of Electrical Engineering

San Jose State University

In partial fulfillment

of the Requirements for the Degree

Master of Science

By

A. Alfred Joseph Jude

December 2003

UMI Number: 1418704

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI[®]

UMI Microform 1418704

Copyright 2004 by ProQuest Information and Learning Company.

All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

© 2003

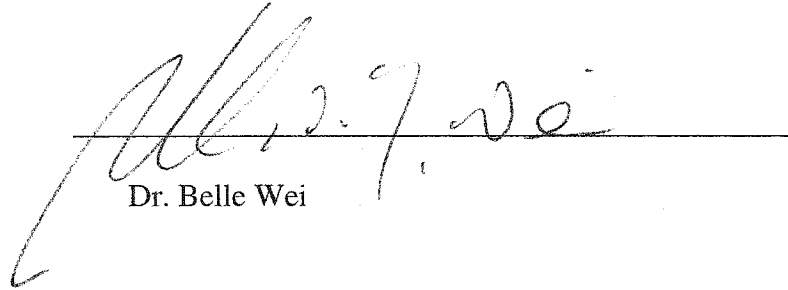
A. Alfred Joseph Jude

ALL RIGHTS RESERVED

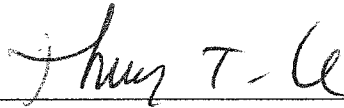
APPROVED FOR THE DEPARTMENT OF ELECTRICAL ENGINEERING



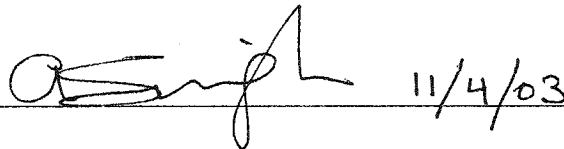
Dr. Nader Mir
(Thesis Committee Chair)



Dr. Belle Wei



Dr. Thuy Le



Dr. Avtar Singh
(Graduate Coordinator)

APPROVED FOR THE UNIVERSITY



ABSTRACT

DESIGN AND ANALYSIS OF TRAFFIC SHAPING AND SCHEDULING IN A FAST SWITCH NETWORK

by A. Alfred Joseph Jude

In this thesis, the Input Port Processor (IPP) section of a high speed switching network is designed and analyzed. The motivation is obtained from a proposed switching fabric called the “Spherical Switching Network” for high speed applications in next generation networking. The system evaluation is carried out by building a stochastic model and deriving various performance metrics. The switching system consists of four main blocks: namely, the input port processors, output port processors, crossbar switch, and crossbar controller. The switching network considered has unique features like a single buffer at the IPP for the entire network without any internal buffer. Hence the IPP has to be intelligent to enhance the performance of the network in terms of the quality of service (QoS) and speed. This thesis will display a stochastic evaluation of packet scheduling, traffic shaping, and multicasting, and present a queuing model for the entire IPP network.

TABLE OF CONTENTS

<i>LIST OF FIGURES</i>	<i>viii</i>
<i>LIST OF TABLES</i>	<i>x</i>
<i>Chapter 1 Introduction</i>	<i>1</i>
1.1 Networking Overview.....	1
1.1.1 Network.....	1
1.1.2 Advantages of a Network.....	3
1.1.3 Networking Protocols	5
1.1.4 Network Operating System (NOS).....	8
1.1.5 Networking Hardware.....	11
1.1.6 Network Cabling.....	19
1.1.7 Wireless LANs.....	24
1.2 Latest (Current) Trends in Networking.....	26
1.2.1 Wired Networks.....	26
1.2.2 Wireless Networks.....	30
1.2.3 Consumer Network Technologies.....	33
1.2.4 Additional Topics.....	38
1.3 Packet Switching.....	39
1.3.1 Crossbar Switches.....	40
1.4 Organization of Thesis.....	41
<i>Chapter 2 Network Interfaces</i>	<i>44</i>
2.1 Basic Interfaces.....	44
2.1.1 Port.....	44
2.1.2 Transducer Operation.....	44
2.1.3 Interface Drivers	45
2.1.4 MAC Address	45
2.1.5 IP Address.....	46
2.1.6 Interface Cards.....	46
2.1.7 Programmable Network Interfaces	46
2.1.8 Workload for PNIs.....	47
2.2 Switch Fabric Architecture	48
2.2.1 Traffic Bottlenecks and QoS: Main Concern in Switch Designs	48
2.2.2 Traditional View of Switching Systems	49
2.2.3 Sharing the Load.....	51
2.2.4 Alternate Designs.....	51
2.2.5 Multi-Stage Interconnection Network (MIN).....	52
2.2.6 The Crossbar Approach	52
2.3 Functionalities of a Switch.....	55

2.4	Traffic Management.....	58
2.4.1	Traffic Control	59
2.4.2	Traffic Contract.....	60
2.4.3	Congestion Control	61
2.5	Traffic Engineering.....	61
2.5.1	Queuing Theory Overview	62
<i>Chapter 3 Spherical Switching Network (SSN)</i>		<i>64</i>
3.1	Introduction.....	64
3.2	SSN Architecture	66
3.2.1	Self-Routing Feature.....	68
3.2.2	Proposed Switch Element	70
3.3	Performance Evaluations	71
3.3.1	Performance Results and Comparison.....	72
3.4	Reasons for Choosing SSN for Thesis.....	75
<i>Chapter 4 Hardware Design of Input Port Processor (IPP).....</i>		<i>77</i>
4.1	Multicasting	77
4.2	Scheduler.....	80
4.3	Traffic Shaping	81
4.3.1	Reasons for Traffic Shaping in the IPP.....	84
4.3.2	Leaky Bucket	86
4.3.3	Token Bucket.....	88
4.3.4	Reasons for Choosing Leaky Bucket over Token Bucket	89
4.3.5	Datapath of Leaky Bucket Design	92
4.3.6	Description of the Blocks and Pins.....	92
4.3.7	RTL Codes	94
4.4	Other Design Blocks of IPP – An Overview	101
4.4.1	Serial to Parallel Converter.....	102
4.4.2	Packet Encapsulator	102
4.4.3	Routing Table.....	103
4.4.4	Routing Table Memory.....	104
4.4.5	Routing Table Lookup	104
4.4.6	First In First Out (FIFO)	104
4.4.7	Free Memory Stack.....	105
4.4.8	Memory	106
4.4.9	Scheduler.....	106
<i>Chapter 5 Performance Analysis of Input Port Processor (IPP).....</i>		<i>110</i>
5.1	IPP Design Modification.....	112
5.2	Statistical Analysis of the IPP.....	112
5.2.1	System Models.....	113

5.3	Modeling Real Time Packets (RTP).....	113
5.3.1	Mean Total RTP Packet Delay	116
5.4	Total Time of a Packet inside the Shaper-Scheduler.....	119
5.5	Modeling Non-RTP Queues	120
5.5.1	Carried Load (vs.) Offered Load for Non-RTP Queues	122
5.5.2	Mean Packet Delay (vs.) Offered Load in Non-RTP Queues.....	123
5.6	Modeling the IPP as a Multi-Server System (M/M/c, M/M/c/c).....	124
5.6.1	Important Estimations.....	125
5.7	Comparison of Scheduler with Different Server Powers.....	129
5.7.1	Scheduler with a Single Server.....	130
5.7.2	Scheduler with Multi-Servers	130
5.8	Waiting Time Distribution of Multi-Server IPP	132
5.8.1	Multi-Server Scheduler with Variable Capacity.....	132
5.9	Finite-Source Queuing System IPP Design	133
5.9.1	Finite-Source Queuing System Example	136
5.9.2	Arriving Packets Distribution	137
5.10	Analysis of Traffic Shaper Output.....	138
5.10.1	Residual Service Time	139
5.10.2	Mean Delay of Shaper Traffic	140
5.11	Comparison of Mean Waiting Times of IPP Queues with Exponential and Constant Residual Service Times	141
5.12	Priority Service: Mean Delay Calculations.....	143
5.12.1	Practical Methods to Compute Mean Waiting Time	144
5.12.2	Shaper with Embedded Markov Chains	146
5.12.3	Embedded Markov Chain	147
5.13	Number of Packets in the Shaper Buffers.....	148
5.14	Delay and Waiting Time Distributions in Shaper FIFOs.....	149
5.15	Departure from Shaper-Scheduler Systems	150
5.15.1	Application of Burke's Theorem on IPP	152
5.15.2	Network of Queues	154
5.15.3	Closed Queue Networks	154
5.15.4	Closed Queuing Network in IPP.....	154
 <i>Chapter 6 Conclusion</i>		 <i>158</i>
 <i>Chapter 7 Future Work.....</i>		 <i>162</i>
 <i>REFERENCES</i>		 <i>165</i>

LIST OF FIGURES

Figure 1.1: LAN – Simple Ethernet Network using Crossover Cables	2
Figure 1.2: Computers on a Standard Ethernet LAN.....	4
Figure 1.3: Client Server Network.....	10
Figure 1.4: Ethernet Hub Network.....	15
Figure 1.5: Simple Switch Network.....	17
Figure 1.6: Connection Devices used to link Different Network Components	18
Figure 1.7: General 802.11b Wireless Ethernet Network without WAP.....	25
Figure 1.8: Phone Line Network between Computers.....	28
Figure 1.9: Wireless Connectivity	30
Figure 1.10: 802.11b Wireless Ethernet Network with WAP	31
Figure 1.11: Packet Switching	40
Figure 1.12: Crossbar.....	41
Figure 2.1: Network Interface Design Example	48
Figure 2.2: Conventional Switch Design	49
Figure 2.3: Crossbar and Network Interfaces	54
Figure 3.1: A 16-port Proposed Spherical Switching Network	67
Figure 3.2: Routing scheme in Spherical Switching Network.....	68
Figure 3.3: The Proposed Structure of the Switch Element.....	70
Figure 3.4: Comparison of Throughput (vs.) Offered Load for the Five Networks	72
Figure 3.5: Comparison of Delay (vs.) Offered Load for the Five Networks	73
Figure 3.6: Effect of Doubling the Speed-Up Factor (Speed Advantage) on the Throughput (vs.) Offered Load	74
Figure 4.1: IPP Block Diagram for Multicasting Queues with Relationship to External Inputs and Scheduler	78
Figure 4.2: Block Diagram of the Scheduler	80
Figure 4.3: Constant and Variable Bit Rate Inputs to a Shaper	83
Figure 4.4: Traffic Analysis Diagram.....	85
Figure 4.5: Leaky Bucket Algorithm.....	87
Figure 4.6: Token Bucket Algorithm.....	89
Figure 4.7: IPP Block Diagram.....	91
Figure 4.8: Leaky Bucket Datapath	92
Figure 4.9: IPP System Block Diagram.....	101
Figure 4.10: SSN Header format	102
Figure 5.1: Preliminary IPP Design	110
Figure 5.2: Shaper-Scheduler General Design.....	111
Figure 5.3: Modified IPP Design	112
Figure 5.4: RTP Queues Transition Rate Diagram.....	114
Figure 5.5: Mean Number of RTP Packets (vs.) Utilization.....	115
Figure 5.6: Mean Total RTP Packet Delay.....	116
Figure 5.7: Mean Delay (vs.) Service Rate of RTP	117
Figure 5.8: Transition Rate Diagram for Finite-Capacity Non-RTP System	120

Figure 5.9: Probability Mass Function for $N(t)$ of Non-RTP Packets	121
Figure 5.10: Carried Load (vs.) Utilization of Non-RTP Packets	123
Figure 5.11: Mean Packet Delay (vs.) Utilization of Non-RTP Packets	124
Figure 5.12: IPP as a Multi-Server System.....	124
Figure 5.13: Mean Number of Packets (vs.) Utilization in a Multi-Server IPP.....	126
Figure 5.14: Mean Number of Packets (vs.) Waiting Times in a Multi-Server IPP.....	127
Figure 5.15: Mean Waiting Times (vs.) Total Times in a Multi-Server IPP	128
Figure 5.16: Mean Number (vs.) Total Times in a Multi-Server IPP.....	129
Figure 5.17: Scheduler with Single-Server.....	130
Figure 5.18: Scheduler with Multi-Servers.....	131
Figure 5.19: Variable Capacity Multi-Server Scheduler Transition Rate Diagram.....	133
Figure 5.20: Finite Source Single-Server Shaper-Scheduler	134
Figure 5.21: Markov Chain Transition Rate Diagram	134
Figure 5.22: Shaper Output Analysis in IPP	138
Figure 5.23: Sequence of Service Times	139
Figure 5.24: Plot of Mean Waiting Time (vs.) Server Utilization	142
Figure 5.25: Priority Scheduling.....	144
Figure 5.26: Mean Waiting Times for Priority Systems.....	146
Figure 5.27: Network of Queues.....	151
Figure 5.28: IPP according to Burke's Theorem	152
Figure 5.29: A Simple Example of Scheduler's Servers (Transition Rates)	153
Figure 5.30: Feed-Forward Network of Queues	153
Figure 5.31: Closed Network Model of IPP	155
Figure 5.32: IPP as Closed Queue	155
Figure 7.1: Parallel Architecture of Crossbar Mesh	163

LIST OF TABLES

Table 1.1: Network Protocols	8
Table 1.2: Ethernet (vs.) Local Talk	14
Table 1.3: Categories of Unshielded Twisted Pair	20
Table 1.4: Ethernet Cable Summary	24
Table 4.1: QoS Classifier Table.....	81
Table 4.2: Example of the Routing Table.....	103

Chapter 1 Introduction

1.1 *Networking Overview*

1.1.1 Network

A network can be defined as an interconnected collection of autonomous computers. A network is formed primarily for resource sharing, file sharing, and for various electronic communications and transactions. The various communication links or channels are cables, satellites, optic fibers, infra-red light beams, etc.

The three basic types of networks include:

- Local Area Network (LAN)
- Metropolitan Area Network (MAN)
- Wide Area Network (WAN)

Local Area Network:

A Local Area Network (LAN) is applied to a small area such as a building or school. It has a powerful machine called file server that runs all the software required for controlling the network. Computers connected to this server are called workstations that are less powerful, but contain hard drives that could run additional software for operation. To be connected to the network, each machine or device should possess a network interface card that would allow the cables to be physically connected to the device. An

in-depth description about cabling is given later in this section.

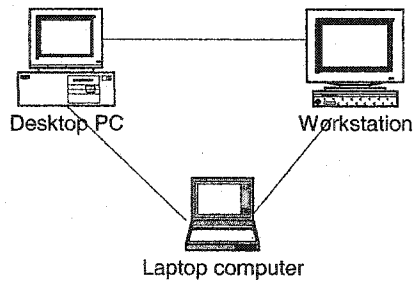


Figure 1.1: LAN – Simple Ethernet Network using Crossover Cables

Metropolitan Area Network:

A Metropolitan Area Network (MAN) covers a large area like a city or town. Basically, it is a collection of numerous smaller networks or LANs, e.g., connection of government offices to private industries.

Wide Area Network:

Wide Area Network (WAN) is used for connecting countries, states, counties, etc. Colossal levels of cabling and infrastructure are required. Satellites play an important role in WANs. Since inter-continental connectivity is possible, high paid phone calls can be avoided. Multiplexers are employed to connect LANs and MANs to WANs.

1.1.2 Advantages of a Network

Speed: Networks provide the fastest and most efficient way of transferring files and sharing resources. If networking does not exist, the traditional method of sneaker-net, which is the transferring and sharing of files through floppy and compact disks, would be the only practice.

Cost: Software programs that exist in the form of network able versions are cheaper than the licensed copies. Updating of programs or software needs to be done only on the server and not on every workstation.

Security: Very important files or private files can be designated with a header or caption to protect from viewing by all. Passwords can be set to these files to prevent access by all users. In this way, a network provides maximum security, and is equivalent to handling files on a PC.

Centralized Software Management: The greatest application of a network, as mentioned earlier, is the handling of software in a single server. Installations, downloads, updating, tracking of files, etc. can be done on a centralized server, rather than on every workstation.

Resource Sharing: Sharing resources is another area in which a network exceeds stand-alone computers. Most schools cannot afford enough laser printers, fax machines, modems, scanners, and CD-ROM players for each computer. However, if these peripherals are added to a network, they can be shared by many users.

Electronic Mail: Email is one of the most important applications of a network. It is seen

very often in real life. Sending and receiving information or data through the web or the internet is the email's functionality. Data is actually switched between systems and thus transmitted and received.

Flexible Access: Networks in universities can be referred to as flexible access function. Students are given access to files throughout the campus. If a student starts an assignment inside the classroom, he or she is allowed to work on the same, from any other lab after the class is over. In this way, access is so flexible and useful.

Workgroup Computing: Workgroup software (like Lotus Notes) can be accessed and updated by different users from various places simultaneously. In this way, a common database can be maintained by a large number of users without needing to work from a single place.

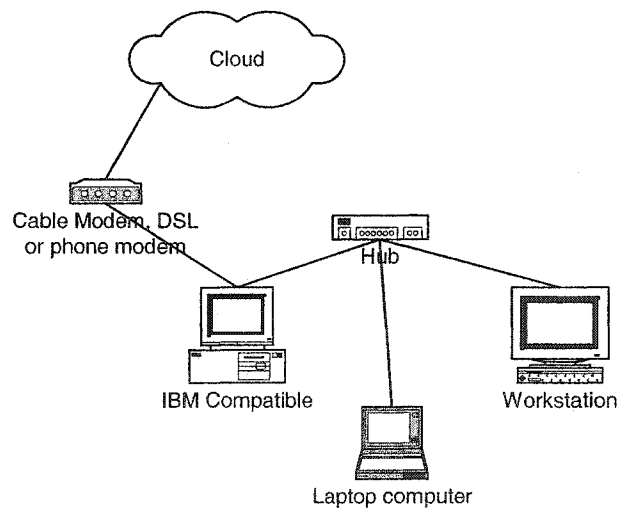


Figure 1.2: Computers on a Standard Ethernet LAN

1.1.3 Networking Protocols

A protocol is a set of rules that governs effective communication between computers in a network. Any networking device that performs a function or operation must have a protocol within itself. There are various guidelines in the design of a protocol. Some of them include physical topology or design, access method, data transfer rate, cabling method, traffic characteristics through the interfaces, etc. When designed, a protocol is seen that it contains all the above mentioned functionality criteria for effective performance.

The most common protocols are:

- Ethernet
- LocalTalk
- Token Ring
- FDDI

Ethernet:

The Ethernet protocol is the most widely used of all protocols. Ethernet uses CSMA/CD access method (Carrier Sense Multiple Access with Collision Detection). In this method, each device or machine listens to the channel or cable. If the line is free, then the device sends information or data. If the link is occupied, it waits for a while, and then transmits when the line is free again. With this access method, it is normal to have

collisions. The delay caused due to collisions and retransmissions is very small and so the data rate of the entire system is not changed. The Ethernet protocol allows for linear bus, star, or tree topologies. Data can be transmitted over twisted pair, coaxial, or fiber optic cable at a speed of 10 Mbps.

Fast Ethernet:

Ethernet with an enhanced speed of transmission of data is called Fast Ethernet. The speed is of the order of 100 Mbps or more. The network devices and interface cards connected in fast ethernet are more expensive. Also, a twisted pair of fiber optic cables is necessary for connection.

LocalTalk:

LocalTalk is a network protocol that was developed by Apple Computer, Inc. for Macintosh computers. The method used by LocalTalk is called CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance). It is similar to CSMA/CD except that a computer signals its intent to transmit before it actually transmits. LocalTalk adapters and special twisted pair cables can be used to connect a series of computers through the serial port. The Macintosh operating system allows the establishment of a peer-to-peer network without the need for additional software. With the addition of the server version of AppleShare software, a client/server network can be established. The LocalTalk

protocol allows for linear bus, star, or tree topologies using twisted pair cable. The primary disadvantage of LocalTalk is speed. Its speed of transmission is only 230 Kbps.

Token Ring:

IBM developed the Token Ring protocol. Computers are connected to form a ring-like design. Signal is transmitted from one computer to the other. There is a single electronic token that passes around the ring. When a computer wants to transmit data, it attaches a token with the data. It means that only if a computer acquires this token, it can transmit data. So the token reaches the destination computer with the data, the latter is received, and the former passes through the ring for the next transmission. This mechanism is called token-passing. The Token Ring protocol requires a star-wired ring using twisted pair or fiber optic cable. It can operate at transmission speeds of 4 Mbps or 16 Mbps.

FDDI:

Fiber Distributed Data Interface (FDDI) is a network protocol that is used to interconnect three or more LANs. The access method used in FDDI is token-passing. The physical topology applied is called dual-ring. Since fiber based, FDDI is known for its speed that is of the order of 100 Mbps. Even if a link in one ring fails, the network does not face too much havoc, since the other ring will compensate.

Table 1.1 is the summary of all the previously-mentioned network protocols:

Protocol	Cable	Speed	Topology
Ethernet	Twisted Pair, Coaxial, Fiber	10 Mbps	Linear Bus, Star, Tree
Fast Ethernet	Twisted Pair, Fiber	100 Mbps	Star
LocalTalk	Twisted Pair	.23 Mbps	Linear Bus or Star
Token Ring	Twisted Pair	4 Mbps - 16 Mbps	Star-Wired Ring
FDDI	Fiber	100 Mbps	Dual ring

Table 1.1: Network Protocols

1.1.4 Network Operating System (NOS)

The NOS has the most important components in a network. It acts like a director and manages the efficient operation of all devices and functionalities of a network. The two major types of network operating systems are:

- Peer-to-Peer
- Client/Server

Peer-to-Peer:

The Peer-to-Peer network operating system is used to share resources and files, and access them from other computers on the network. There is no file server or a centralized management scheme because as the name suggests, all computers are treated equal and have the same ability and functionality. Usually small to medium LANs are employed with this operating system. Windows 95 and Windows for workgroups, are some of the Peer-to-Peer operating systems used.

Advantages of a Peer-to-Peer network

- Less initial expense – A dedicated server is not required
- Setup – An OS like Windows 95, which is already installed in a network, needs only to be reconfigured for network Peer-to-Peer operations.

Disadvantages of a Peer-to-Peer network

- Decentralized – There is not a single place or system to store all files and folders.
- Security - Does not provide the security available on a client/server network.

Client/Server:

Client/server network operating system allows the network to centralize functions and applications in one or more dedicated file servers. The file server becomes the heart of the system, providing access to resources, and providing security. Individual workstations (clients) have access to the resources available on the file servers. The network operating system provides the mechanism to integrate all the components of the network, and allows multiple users to simultaneously share the same resources, irrespective of physical location. Novell NetWare and Windows NT server are examples of client/server network operating systems.

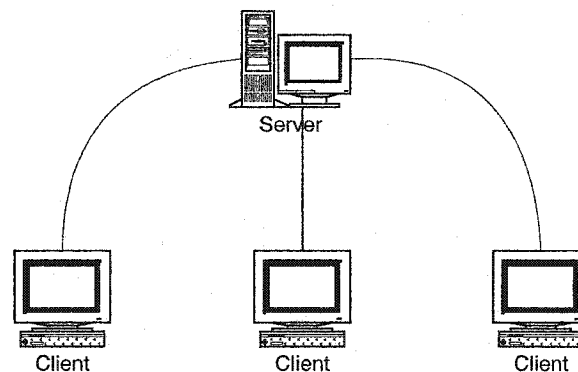


Figure 1.3: Client Server Network

Advantages of a client/server network

- Centralized - Resources and data security are controlled through the server.
- Scalability - Any or all elements can be replaced individually as the needs increase.
- Flexibility - New technology can be easily integrated into system.

- Interoperability - All components (client/network/server) work together.
- Accessibility - Server can be accessed remotely and across multiple platforms.

Disadvantages of a client/server network

- Expense - Requires initial investment in dedicated server.
- Maintenance - Large networks will require staff to ensure efficient operation.
- Dependence - When the server goes down, operations will cease across the network.

1.1.5 Networking Hardware

Networking hardware includes all computers, peripherals, interface cards, and other equipment needed to perform data-processing and communications within the network.

This section provides information on the following components:

- File Servers
- Workstations
- Network Interface Cards
- Concentrators/Hubs
- Repeaters
- Bridges
- Routers

File Servers:

A File Server stands at the heart of most networks. It is a very fast computer with a large amount of RAM and storage space, along with a fast network interface card. The network operating system software resides on this computer, along with any software applications and data files that need to be shared. The file server controls the communication of information between the nodes on a network. For example, it may be asked to send a word processor program to one workstation, receive a database file from another workstation, and store an e-mail message during the same time period. This requires a computer, like a file server, that can store a lot of information and share it very quickly.

Workstations:

All the computers connected to the file server on a network are called workstations. A typical workstation is a computer that is configured with a network interface card, networking software, and the appropriate cables. Workstations do not necessarily need floppy disk drives or hard drives, because files can be saved on the file server itself. Almost any computer can serve as a network workstation.

Network Interface Cards:

The Network Interface Card (NIC) provides the physical connection between the network and the computer workstation. Most NICs are internal, with the card fitting into an expansion slot inside the computer. Some computers, such as MAC-Classics, use external boxes which are attached to a serial port or a SCSI port. Laptop computers generally use external LAN adapters connected to the parallel port, or network cards that slip into a PCMCIA slot. Network interface card is a major factor in determining the speed and performance of a network. It is a good idea to use the fastest network card available for the type of workstation used. The three most common network interface connections are ethernet cards, LocalTalk connectors, and token ring cards. According to an international data corporation study, ethernet is the most popular card, followed by token-ring, and LocalTalk.

Ethernet Cards:

Ethernet cards are usually purchased separately from a computer, although many computers (such as the Macintosh) now include an option for a pre-installed ethernet card. Ethernet cards contain connections for either coaxial or twisted pair cables (or both). If it is designed for coaxial cable, the connection will be BNC. If it is designed for twisted pair, it will have a RJ-45 connection. Some ethernet cards also contain an AUI connector. This can be used to attach coaxial, twisted pair, or fiber optic cable to an ethernet card. When this method is used, there is always an external transceiver attached

to the workstation. (See *Section 1.1.6* for more information on connectors.)

LocalTalk Connectors:

LocalTalk is Apple's built-in solution for networking Macintosh computers. It utilizes a special adapter box and a cable that plugs into the printer port of a Macintosh. A major disadvantage of LocalTalk is that it is slow in comparison to ethernet. Most ethernet connections operate at 10 Mbps (megabits per second). In contrast, LocalTalk operates at only 230 Kbps (or .23 Mbps).

Ethernet	LocalTalk
Fast data transfer (10 Mbps)	Slow data transfer (.23 Mbps)
Purchased separately	Built into Macintosh computers
Requires computer slot	No computer slot necessary
Available for most computers	Works only on Macintosh computers

Table 1.2: Ethernet (vs.) LocalTalk

Token Ring Cards:

Token Ring network cards look similar to ethernet cards. One visible difference is the type of connector on the back end of the card. Token Ring cards generally have a nine pin DIN type connector to attach the card to the network cable.

Concentrators/Hubs:

A concentrator is a device that provides a central connection point for cables from workstations, servers, and peripherals. In a star topology, twisted-pair wire is run from each workstation to a central concentrator. Hubs are multi-slot concentrators into which, a number of multi-port cards can be plugged to provide additional access as the network grows in size. Some concentrators are passive, that is, they allow the signal to pass from one computer to another without any change. Most concentrators are active, that is, they electrically amplify the signal as it moves from one device to another. Active concentrators are used like repeaters to extend the length of a network.

Concentrators/Hubs are:

- Usually configured with 8, 12, or 24 RJ-45 ports
- Often used in a star or star-wired ring topology
- Sold with specialized software for port management

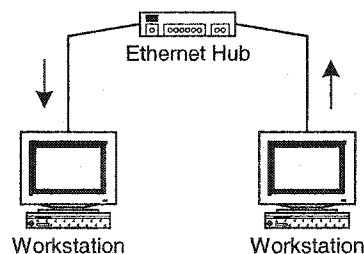


Figure 1.4: Ethernet Hub Network

Repeaters:

When a signal travels along a cable, it tends to lose strength. A repeater is a device that boosts a network's signal as it passes through. The repeater does this by electrically amplifying the signal it receives, and re-broadcasting it. Repeaters can be separate devices or they can be incorporated into a concentrator. They are used when the total length of the required network cable exceeds the standards set. A good example of the use of repeaters would be in a local area network using a star topology with unshielded twisted-pair cabling. The length limit for unshielded twisted-pair cable is 100 meters. The most common configuration is for each workstation to be connected by twisted-pair cable to a multi-port active concentrator. The concentrator regenerates all the signals that pass through it, allowing the total length of cable on the network to exceed the 100-meter limit.

Switches:

Switches avoid the congestion of a shared ethernet network by dividing it into individual subnets, and connecting them. The improvement in network performance can be dramatic. In the illustration shown in *Figure 1.5*, the switch is being fed a 100 Mbps signal. The switch is then creating four segmented networks each with its own 10 Mbps path. Net3 and Net4 are then connecting to a hub, creating two shared 10 Mbps networks. Switches come in a variety of configurations.

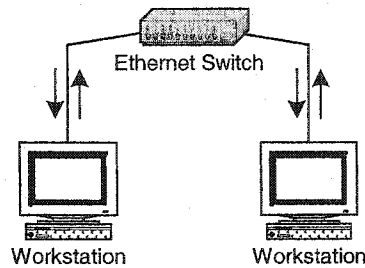


Figure 1.5: Simple Switch Network

Bridges:

A bridge is a device that allows segmenting of a large network into two smaller, more efficient networks. It can be used to improve an old wiring scheme, as well as maintain the current scheme up to date. A bridge monitors the information traffic on both sides of the network, so that it can pass packets of information to the correct location. Most bridges can "listen" to the network, and automatically figure out the address of each computer on both sides of the bridge. The bridge can inspect each message, and if necessary, broadcast it on the other side of the network. The bridge manages the traffic to maintain optimum performance on both sides of the network. It can be said that the bridge is like a traffic cop at a busy intersection during a rush hour. It keeps information flowing on both sides of the network, but at the same time, it does not allow unnecessary traffic to pass through. Bridges can be used to connect different types of cables or physical topologies. They must, however, be used between networks with the same protocol.

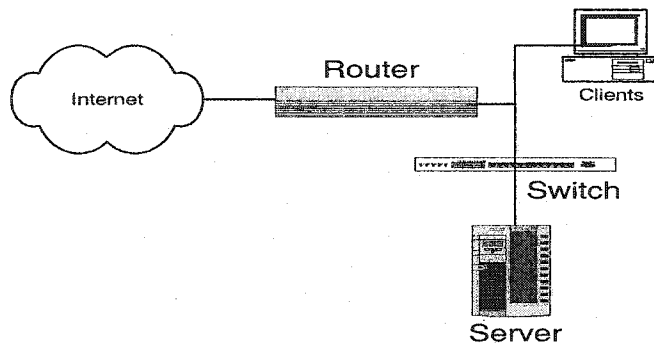


Figure 1.6: Connection Devices used to link Different Network Components

Routers:

A router translates information from one network to another; it is similar to a super-intelligent bridge. Routers select the best path to route a message, based on the destination address and origin. The router can direct traffic to prevent head-on collisions, and is smart enough to know when to direct traffic along back roads and shortcuts. While bridges know the addresses of all computers on each side of the network, routers know the addresses of computers, bridges, and other routers on the network. Routers can even "listen" to the entire network to determine which sections are busiest, and they can then redirect data around those sections until they clear up. If a school LAN needs to be connected to the internet, a router is needed. In this case, the router serves as the translator between the information on the LAN and the internet. It also determines the best route to send the data over the internet. Routers can:

- Direct signal traffic efficiently
- Route messages between any two protocols
- Route messages between linear bus, star, and star-wired ring topologies
- Route messages across fiber optic, coaxial, and twisted-pair cabling

1.1.6 Network Cabling

Cable is the medium through which information usually moves from one network device to another. There are several types of cables, which are commonly used with LANs. In some cases, a network will utilize only one type of cable, while other networks will use a variety of cable types. The type of cable chosen for a network is related to the network's topology, protocol, and size. Understanding the characteristics of different types of cable, and how they relate to other aspects of a network, is necessary for the development of a successful network. The following sections discuss the types of cables used in networks and other related topics.

- Unshielded Twisted Pair (UTP) cable
- Shielded Twisted Pair (STP) cable
- Coaxial cable
- Fiber Optic cable
- Wireless LANs
- Installing cables - Some guidelines

Unshielded Twisted Pair (UTP) cable:

Twisted pair cabling comes in two varieties: shielded and unshielded. Unshielded Twisted Pair (UTP) is the most popular, and is generally the best option for school networks. The quality of UTP may vary from telephone-grade wire to extremely high-speed cable. The cable has four pairs of wires inside the jacket. Each pair is twisted with a different number of twists per inch, to help eliminate interference from adjacent pairs and other electrical or electronic devices. The EIA-TIA (Electronic Industry Association-Telecommunication Industry Association) has established standards for UTP, and has rated five categories of wire.

Type	Use
Category-1	Voice Only (Telephone Wire)
Category-2	Data to 4 Mbps (LocalTalk)
Category-3	Data to 10 Mbps (Ethernet)
Category-4	Data to 20 Mbps (16 Mbps Token Ring)

Table 1.3: Categories of Unshielded Twisted Pair

One difference between the different categories of UTP is the tightness in the twisting of the copper pairs. The tighter the twisting, the higher is the supported transmission rate, and greater is the cost per foot. The best cable has to be used; most schools purchase category-3 or category-5. Category-5 cable is highly recommended. If a 10 Mbps ethernet network has to be designed, and if the cost savings of buying category-3 wire

instead of category-5 are considered, it has to be remembered that the category-5 cable will provide more "room to grow", as the transmission technologies increase. Both category-3 and category-5 UTP have a maximum segment length of 100 meters. 10BaseT refers to the specifications for unshielded twisted pair cable (category 3, 4, or 5) carrying ethernet signals.

Unshielded Twisted Pair Connector:

The standard connector for unshielded twisted pair cabling is an RJ-45 connector. This is a plastic connector that looks like a large telephone-style connector. A slot allows the RJ-45 to be inserted in only one way. RJ stands for Registered Jack, implying that the connector follows a standard borrowed from the telephone industry. This standard designates which wire goes with each pin inside the connector.

Shielded Twisted Pair (STP) Cable:

A disadvantage of UTP is that it may be susceptible to radio and electrical frequency interference. Shielded Twisted Pair (STP) is suitable for environments with electrical interference; however, the extra shielding can make the cables quite bulky. Shielded twisted pair is often used in networks using token ring topology.

Coaxial Cable:

Coaxial cabling has a single copper conductor at its center. A plastic layer provides insulation between the center conductor and a braided metal shield. The metal shield helps to block any outside interference from fluorescent lights, motors, and other computers.

Although coaxial cabling is difficult to install, it is highly resistant to signal interference. In addition, it can support greater cable lengths between network devices than twisted pair cable. The two types of coaxial cabling are: thick coaxial and thin coaxial. Thin coaxial cable is also referred to as thin-net. 10Base2 refers to the specifications for thin coaxial cable carrying ethernet signals. The 2 refers to the approximate maximum segment length being 200 meters. In fact, the maximum segment length is 185 meters. Thin coaxial cable is popular in school networks, especially linear bus networks. Thick coaxial cable is also referred to as thick-net. 10Base5 refers to the specifications for thick coaxial cable carrying ethernet signals. The 5 refers to the maximum segment length being 500 meters. Thick coaxial cable has an extra protective plastic cover that helps to keep moisture away from the center conductor. This makes thick coaxial a great choice when running longer lengths in a linear bus network. One disadvantage of thick coaxial is that it does not bend easily, and is difficult to install.

Coaxial Cable Connectors:

The most common type of connector used with coaxial cables is the Bayone-Neill-Concelman (BNC) connector. Different types of adapters are available for BNC connectors, including a T-connector, barrel connector, and terminator. Connectors on the cable are the weakest points in any network. To help avoid problems with the network, BNC connectors that crimp must always be used, rather than connectors that need to be screwed onto the cable.

Fiber Optic Cable:

Fiber optic cable consists of a center glass core surrounded by several layers of protective materials. It transmits light rather than electronic signals, eliminating the problem of electrical interference. This makes it ideal for certain environments that contain a large amount of electrical interference. It is also the standard for connecting networks between buildings, due to its immunity to the effects of moisture and lighting. Fiber optic cable has the ability to transmit signals over much longer distances than coaxial and twisted pair. It also has the capability to carry information at vastly greater speeds. This capacity broadens communication possibilities to include services such as video conferencing and interactive services. The cost of fiber optic cabling is comparable to copper cabling; however, it is more difficult to install and modify. 10BaseF refers to the specifications of fiber optic links carrying ethernet signals.

Facts about fiber optic cables

- Outer insulating jacket is made up of teflon or PVC.
- Kevlar fiber helps to strengthen the cable, and prevent breakage.
- A plastic coating is used to cushion the fiber center.
- Center (core) is made up of glass or plastic fibers.

Fiber Optic Connector:

The most common connector used with fiber optic cable is an ST connector. It is barrel shaped, similar to a BNC connector. A newer connector, the SC, is becoming more popular. It has a squared face, and is easier to connect in a confined space.

Specification	Cable Type
10BaseT	Unshielded Twisted Pair
10Base2	Thin Coaxial
10Base5	Thick Coaxial
10BaseF	Fiber Optic

Table 1.4: Ethernet Cable Summary

1.1.7 Wireless LANs

Not all networks are connected with cabling; some networks are wireless. Wireless LANs use high frequency radio signals or infrared light beams to communicate between the workstations and the file server. Each workstation and file server on a wireless network has some sort of transceiver/antenna to send and receive the data. Information is

relayed between transceivers as if they were physically connected. For longer distances, wireless communication can also take place through cellular telephone technology or by satellite. Wireless networks are great for allowing laptop computers or remote computers to connect to the LAN. Wireless networks are also beneficial in older buildings, where it may be difficult or impossible to install cables. They can either have Wireless Access Points (WAP) or not. Those without WAPs are generally called ad-hoc wireless networks. The wireless protocols are 802.11 that range from “a” until “g”.

Wireless LANs also have some disadvantages. They can be very expensive, provide poor security, and are susceptible to electrical interference from lights and radios.

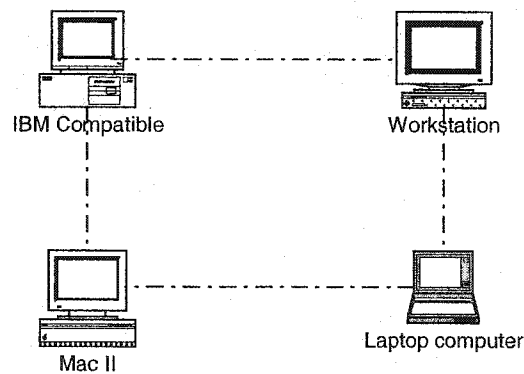


Figure 1.7: General 802.11b Wireless Ethernet Network without WAP

1.2 Latest (Current) Trends in Networking

In this section, the latest and cutting edge networking technologies that exist in the industry today will be discussed. It is necessary to go through the wired and wireless concepts, consumer networking, and some additional topics as well. They are as follow:

1.2.1 Wired Networks

Home networking involves connecting devices or peripherals and systems, and transferring voice, data or video within the home. This has to ensure that broadband access is provided to end users within the house. So devices like modems, phones, etc. which in turn are linked through wires, need to be connected. The latest in wired technologies are presented below:

IEEE 1394: This is an international hardware and software standard that is being used for digital interfacing of communication, computing, electronic, and entertainment devices. It provides a low-cost, and high efficiency digital interfacing. The speed on date has already exceeded the unit giga bits per second (Gbps) range while all other standards have been successful only until the Mbps ranges. VCRs, printers, scanners, cameras, etc. that operate digitally can be interfaced by this standard.

Analog Modems: These are modems that work with analog data, voice, fax, etc. They can be connected to a dial-up phone, and operate at the rate of 56 Kbps. The hardware does not need a separate micro-controller, as it is host-controlled. This modem provides error-correction, and data-compression, and maximizes data transfer.

Cable modems: Cable modems also modulate and demodulate data just like an analog modem. But the bandwidth allocated is more, and so the speed of transmission is way above analog modems. They are basically digital cables just like the ones being used for cable TVs. So the modem is connected to the internet using these cables that provide speedy transmission of data, voice or video. But since the bandwidth is shared by users, there might be latency due to its sharing and at heavy traffic times.

xDSL Modems: Digital Subscriber Line (DSL) modems convert the usually existing coaxial cables into access paths. They work on duplex schemes, and the data rates are usually 6 Mbps downstream (to the end-user) and 600 Kbps upstream, thus giving the rate of 1.1 Mbps on both sides (duplex). It segregates the voice, data or video information into blocks, and each block is attached to an error-correction code. This is today's cutting-edge technology modem.

DSLAM: Digital Subscriber Line Access Multiplexer (DSLAM) is a device that is connected to a DSL line. The sole function of DSLAM is to aggregate or multiplex data

traffic from DSL lines before they enter a switch or router to be transferred to other networks or devices.

Ethernet: This is a standard that adheres to the famous 802.3 CSMA/CD (Carrier Sense Multiple Access with Collision Detection) access methods and physical layer specifications. This has been adopted by International Standards Organization (ISO) as the best networking standard. This is secure over other networking standards and also provides speedy transmission.

ISDN Modem: Integrated Services Digital Network (ISDN) modem is used to integrate voice, video, audio, and data, and then transmit them over the network. This has greatly reduced or even avoided the cost and complexity, when all the different forms of information require separate methods or channels to be communicated.

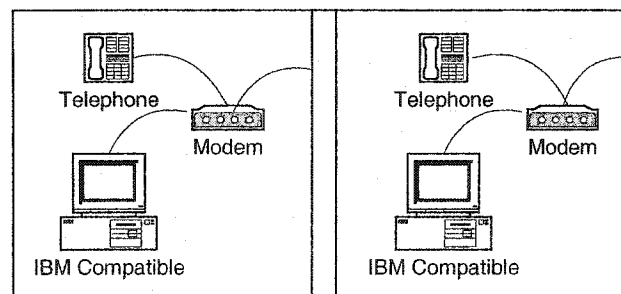


Figure 1.8: Phonetel Network between Computers

Phonetel: The use of phonetel based home networking is greatly felt when it connects PCs, peripherals, and devices with the existing phone wiring itself (without additional

wiring). This is a very user-friendly, high-speed, and low-cost technology.

Powerlines: Powerlines (residential) are the most pervasive network. It is a home networking solution that comes into play when the internet applications keep expanding, and broadband access becomes more desirable. When rooms or buildings need more outlets, the use of latest available powerlines with maximum power capacity and efficiency becomes very valuable.

SOHO routers: Small Office House Office (SOHO) routers connect the LAN to WAN/Internet. SOHO is similar to Virtual Private Network (VPN), where corporate or office data is accessed from home or an outside place for applications like email, directories, video files, etc. SOHO routers also manage the interconnections between the LAN devices.

USB 2.0: Universal Standard Bus (USB) 2.0 is the latest bus standard that is used to connect or interface PCs with peripherals. At present, all PCs are manufactured with many USB ports. The USB 2.0 standard can be used to interchange or swap peripherals without shutting down the system while in operation. Digital imaging, computer telephony, video gaming, etc. are some areas that are hugely benefited by this standard.

1.2.2 Wireless Networks

Wireless networks, as the name suggests, operate without any physical link. Information is transferred in the form of electromagnetic waves from the transmitter to receiver. Wireless systems that emerged in the 70s and 80s can now be divided into three generations. The first wireless generation was analog cellular, and the speed with which data and voice were transmitted was low. AMPS and CNET were the standards used at that time. The second wireless generation shifted towards digital cellular technologies, and the speed of data and voice transfer was considerably enhanced. GSM and CDMA were the standards during this time, and cordless phones were made. The third generation (3G) wireless era is the current one, and the most advanced of all generations. IMT-2000/UMTS are the standards used in 3G systems, which emphasize on two aspects namely: (1) Effective usage of the bandwidth allocated and the spectrum availability; and (2) The highest speed of data transfer, like high speed wireless networks. Some of the cutting-edge wireless technologies are mentioned in this section:

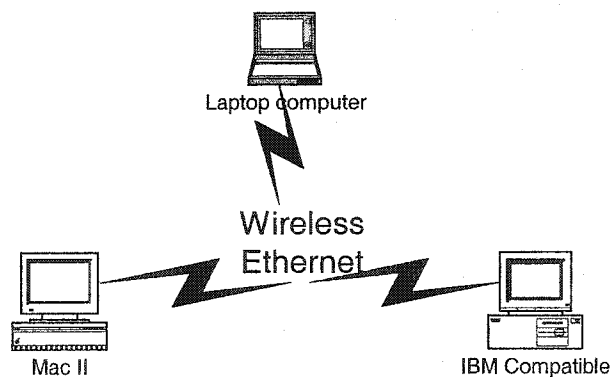


Figure 1.9: Wireless Connectivity

802.11: IEEE 802.11 is a wireless standard that uses CSMA multiple access method. Collision Detection (CD) cannot be put forth for wireless networks, as the signal transmitted is going to hear the same signal perfectly. Only in this way, 802.11 differs from 802.3, which was already explained under wired networks. This standard explains the interface that exists between a wireless client (mobile or laptop), with a base station of a single cell or coverage area.

Bluetooth: It can be called as a small-scale wireless system technology. It serves for a short range of distance, say 15 meters, and a mediocre performance in terms of data rate (600 Kbps). The major advantages are the dynamic configurability and low-power consumption. They can be used for hand held applications, and can transmit all forms of information. They are very advantageous when it comes to network a personal set-up of devices.

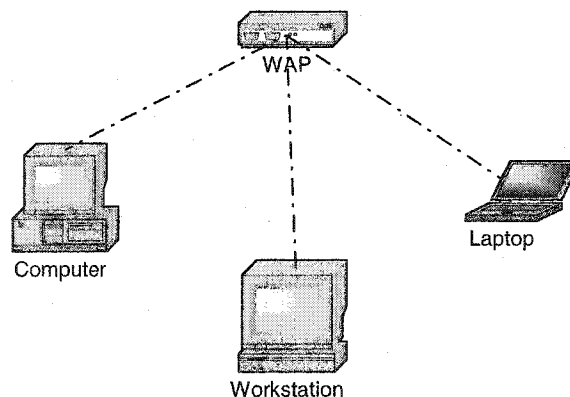


Figure 1.10: 802.11b Wireless Ethernet Network with WAP

HiperLAN2: This is a global broadband wireless technology that operates in the 5GB range (highest). It connects devices and systems in the industrial, organizational, educational as well as personal environments. Mobility and speed of transmission are the best to be pointed out metrics of this technology. The Quality of Service (QoS) is very high when compared to the other existing technologies.

HOME-RF: The idea of providing low cost, data, voice, and video availability made the emergence of HOME-RF technology possible. Voice is handled by DECT (Digital Enhanced Cordless Telecommunications) standards, and data is handled by SWAP (Shared Wireless Access Protocols) standards.

Satellite Modem: In Satellite modem technology, the transponder that transmits TV signals is also used for internet access (data). The satellite sends data digitally to the dish at a very high rate in a video broadcasting system. The PC is connected to the satellite through an interface card. The PC is also connected to the internet through a modem. The service provider and the satellite operators make this set-up function to the maximum efficiency by involving various interfacing techniques.

Wireless LANs: Local Area Networks (LANs) are designed lately to operate with wireless systems. This is made possible by the technologies already mentioned. But the two standards that are greatly responsible are the IEEE 802.11 and HiperLAN2. While

802.11 standards are used for reasonable speed and distances, HiperLAN2 is very effective, e.g., data rate of 60 Mbps over a distance of 200 meters.

1.2.3 Consumer Network Technologies

Digital technologies have become very advanced and produce high-featured and application-specific consumer appliances in networking. Low-cost, user-friendly, low-weight, and less complex devices are greatly being used in the market. Mobile applications that enhance email access, music access, calendar authentication, etc. are made easily. Some of the daily-life applications are given below:

Auto Entertainment: Auto entertainment is the computing or media applications inside a moving vehicle. Some of them are cell phones, music, traffic checks (periodical), remote vehicle maintenance, etc.

Automotive: Automotive is embedded with hi-fi applications like electric windows, central locks, safety systems, climate controls, remote controls, etc. Also traffic information applications, internet access, electronic game control, MPEG music download controls, etc. are employed in latest systems.

Computers: Computers produced in the recent times have all been very economical, very fast, with high memory and storage capacities, high performance processing, and throughput. This was actually made possible by the design and fabrication of high programmable FPGAs and ASICs. The networking aspects cover computer peripherals, storage devices, printers, and PC add-on cards like multimedia, network interface, peripheral controllers, etc.

Digital TV/HDTV: Digital TV transmits a signal that is broadcasted by encoding it into 0s and 1s. It enables multimedia creativity by allowing the transfer of voice, video, and data along the same medium. The output pictures have excellent quality (cinematic) and sound of true CD quality.

Gaming Devices: These devices have the latest hardware and software designs to enable and deliver electronic games to consumers. They run with AC power as input, and need a screen or monitor for output display. They also have modems to connect to the internet, and provide broadband access. Gaming consoles that are mobile, running with batteries are also existent.

Home Security: Controlling sensors, surveillance cameras, video cameras, alarm systems, intelligent circuitry, etc. are some components of a home security network. Powerlines, phonedlines, 1394 or wireless technologies can be used to interface all the

systems to the network.

Internet Audio– MP3: MP3 is the compressed digital audio format of the original music. This standard is superior to all available online commerce today. The quality of MP3 formats downloaded resembles the original track to perfection. This is stored in the hard disk, and easily extracted. The file size is small, and so the quality is sustained. Technologies are researched to make it more efficient, and enhance the performance.

Mobile Phones: Cell phone or mobile is a radio-telephone that has the base station transmitting and receiving the signals in the form of electromagnetic waves. The area of operation or service area is divided into clusters, and in turn cells, to maximize the quality of signal transmission, and enhance the performance. With the latest integrated circuit chips (ASICs), the functionality of today's mobiles has exponentially grown, at the same time becoming very economical too.

Pay Phones: Gone are the days when pay phones were used at public places to only deliver voice. Latest trends and innovations mention that pay phones are able to deliver fax messages, and can be used for internet operations like email access, checking traffic, climate, etc.

PC Appliances: As already mentioned, home networking has lately emerged as the most

recent advancement in the field. PCs are being connected to peripherals or devices accessing broadband, and are becoming active members of the internet cloud.

Personal Digital Assistant (PDA): A generally accepted definition of PDA comes from the PDA industry association. A PDA is primarily a productivity and communications tool that is lightweight, compact, durable, reliable, easy to use, and integrates into existing operations. Typically, it can be held in one hand leaving the other to input data with a pen type stylus or a reduced size keyboard. The latest trends in market say that PDAs have great performance in wireless connectivity, storage, speech memory, multimedia extensions, etc.

Printers: Printers, one of the most widely used output devices, had impact or thermal methods before. But now, every single printer is either of ink-jet type or laser type. This is going to be more innovative, cost-effective, and highly efficient soon.

Screen Phones: Internet screen phones are high-end desktop telephones that have LCD displays. They are used for email access, web browsing or any information service. They are structured with a base module, a communication module (like a handset), key pad, and a display. Right now, there is not much consumer awareness for this technology application, but the trend will change soon.

Set-Top Boxes: High quality digital entertainment and digital information can be witnessed from one's own house using Set-top boxes. Again, they can be used for web access for any purpose.

Smart-Card Readers: A card that is embedded with semiconductors, can be used to replace even small denominations of money, unlock any security door, boot a PC, as well as exchange or transfer information, is now existent. Such a card is called a Smart-Card.

VoIP Phones: Voice over Internet Protocol (VoIP) is the technology used for transmitting voice over packet switched networks. Usually, telephone lines are circuit switched, meaning that there would be a dedicated connection until a call gets over. VoIP is emerging fast, and uses H.323 protocols for operation. It also provides multimedia connectivity. It exists in various forms like Fax over IP (FoIP), Voice over DSL (VoDSL), and VoCABLE (Voice over Cable).

Web/Fridge Pads: This is a consumer information appliance that has a touch screen and a browser based interface for internet accesses. The display is an LCD screen. A web pad is also called a web tablet. The pad is made up of two parts, namely a base station and the LCD display. The base station sends and receives wireless transmissions from the pad. The station is connected to the internet by a modem or any broadband access method.

White Goods: These are nothing but the fridges, washing machines, ovens, dishwashers, microwaves, etc. that are connected to the home network.

1.2.4 Additional Topics

Following are some additional technologies and applications that have emerged most recently in networking.

Video Capturing/Editing: Video graphics are most employed and exploited by all technical persons in entertainment sector. It involves a lot of creativity in thought, which in turn has to be supported or complemented by the existing technologies. This has become reality now, and even a normal person outside the technical sector can create digital video movies, and share with others on the web, educate people, etc.

Video Imaging and Processing: Video and image processing applications such as video broadcasting, video conferencing, medical image processing, 3D graphics, digital photography, film scanning, editing, digital copying, and handling, etc. are some more technologies recently developed with the design and fabrication of cores and expertise for various Digital Signal Processing (DSP) needs and applications.

1.3 Packet Switching

Switching can be defined as the transfer of data from one point to another inside or between networks. Most of this thesis is concerned about this concept, and the design and implementation of one such network. There are two types of switching, namely circuit switching and packet switching. Circuit switching has physical links between the sender and receiver. So whenever there is data transfer, there is a real link established, or in other words, once a call is set up, a dedicated path exists between both ends, and exists until the call is finished. This section has more explanation of packet switching, because that is the switching mode for most of the switching and routing systems.

Packet switching, an alternative to circuit switching, can also be called Message switching, and does not have any dedicated path between sender and receiver. There is no physical circuit for packet transmission. Packet switching is also termed as store and forward switching, because incoming packets that need to be switched are buffered at the first device (like a router), and then forwarded. In this way, each block of data is authenticated, inspected, and then transmitted. It should be noted that in packet switching, there is no limit to the size of data. This also reduces the delay and increases throughput. It also conserves bandwidth because any unused bandwidth by a packet or a data block is utilized by other packets, as opposed to circuit switching where there are numerous dedicated channels. The only problem with this type is that, since it is non-

dedicated, an overwhelming stream of input data might completely crash the link. So this dynamic bandwidth allocation, and store and forward transmission, make this method a very superior and easy-to-use type. A very simple timing display for packet switching is given below:

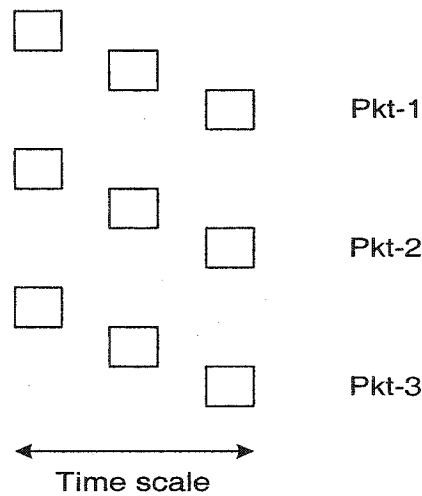


Figure 1.11: Packet Switching

1.3.1 Crossbar Switches

This section defines or delineates a simple crossbar switch that would help the reader in understanding how individual switches actually work inside. This is the simplest of all switches. In a configuration of “x” inputs and “x” outputs, the number of cross-points or intersections is x^2 . Whenever a connection between an input and output is needed, a semiconductor device acts like a switch, and connects both. Since the Spherical

Switching Network (SSN) is also made up of a crossbar, the following diagram might help better in understanding the same.

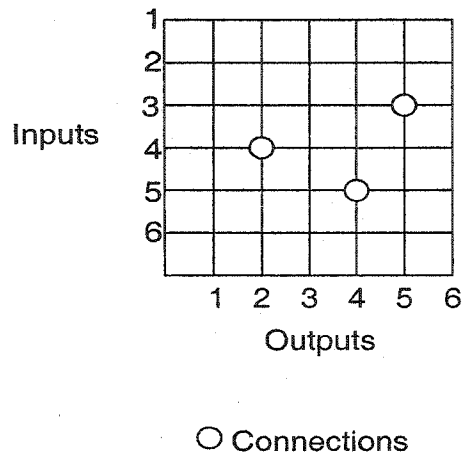


Figure 1.12: Crossbar

1.4 Organization of Thesis

The thesis has two sections which consist of the core technical parts. They are the design and analytical parts. These make up the majority of it. This is complemented by the summary of the entire system, and a brief idea about network interfaces and various concepts. This thesis is organized as follows:

Chapter 2 offers an overview of network interfacing, which is very important in any network hardware design. It explains the basic interfaces like port, transducer, IP and MAC addresses, etc. It explains the conventional switch fabric architecture, and discusses the bottlenecks and betterment methods. Various switch design approaches

including the crossbar are introduced. This chapter gives a feel of switch designs, and helps to get into the real design part.

Chapter 3 summarizes the publication titled, “*An efficient switching fabric for next-generation large-scale computer networks*” by Nader Mir [1]. The entire thesis, as such, is based on the high speed network called the Spherical Switching Network (SSN), which is made up of numerous switch elements. Each element has the input and output port processors, and a crossbar. Since design and analysis of the Input Port Processor (IPP) is carried over in this document, this chapter will help the reader acquire an insight towards the SSN and its functionality.

Chapter 4 portrays the hardware design of the Input Port Processor (IPP), which has three important functionalities, namely, multicasting, shaping, and scheduling. These are performed by designing encapsulations, First In First Out (FIFO) buffers, serial to parallel converter and vice-versa, memory, stack, leaky bucket, etc. The section has the datapath or block diagram with description, explanation of pins, and the RTL codes. The timing diagram will explain the system’s output.

Chapter 5 presents a performance evaluation of the Input Port Processor (IPP) design. A stochastic/probabilistic model that studies the random process or nature of IPP packets is assumed. The various blocks of IPP are assumed as different universal queuing

systems, and an estimation of the parametric involved is computed. Derivation of various values including delay, waiting times, etc. is done, and these parameters are plotted by changing different constant values, which would give an idea of how the system performs and simulates to changing input conditions. This chapter is a seminal innovation in the sense that this new design called IPP is evaluated and analyzed.

Chapter 6 summarizes the contribution of the thesis, and provides recommendations and references.

Chapter 7 discusses the future work that is possible to enhance both the design and its performance.

Chapter 2 Network Interfaces

A Network Interface (NI) is a component that connects any peripheral or device to the network. It can be a separate circuit board that is attached inside the device, or it might be a part of the device's complete internal circuitry board itself. Be it any method, the function is to connect the system or device physically and logically to the network, and also convert information into electrical signals for transmission.

2.1 *Basic Interfaces*

2.1.1 Port

For interfacing or connection purposes, a port is employed. In case of wired networks, the cables or links are plugged into these ports on one side, and to the system attached on the other. If the network is wireless, then the port can be a transmitter or receiver, capable of sending and accepting information.

2.1.2 Transducer Operation

Physical connectivity is one function of a network interface. The other function (as already mentioned) is to convert the information or data into electrical signals so that they move through the network. It has to be noted here that this depends on the standards or protocols followed by the network. Suppose the network is 10BaseT ethernet type, then

the network interface should support 10BaseT standards. There are interfaces like 10BaseT and 10BaseTX that can connect multiple network types.

2.1.3 Interface Drivers

To make the interfaces function, we need some software modules called Drivers. These drivers are very essential for interfacing. They are installed when the interfaces are connected. They are available in disks or they can be obtained from the interface manufacturing companies or downloaded from the internet. Updating the drivers as well as the interfaces from time to time is vital, because the research and advancements in this area are very fast. The instructions to install the drivers can be seen in the interface manual itself. Drivers that can eradicate bugs if any, enhance the performance of interfaces.

2.1.4 MAC Address

Every network interface is associated with a unique address called MAC (Media Access Control) address. This address is used to transfer information in a network through devices like routers or switches within the local network. The concept of MAC address varies according to the network type. For example, an ethernet network has MAC address unique for every single interface. This implies that when an interface is plugged into a slot, the unique MAC address becomes the identity of the system or device. There are networks where MAC addresses are assigned to interfaces on a dynamic basis until the system or device is connected to that network.

2.1.5 IP Address

There is also another type of address to every system or device in a network, called the IP address (Internet Protocol). This address is also unique to every machine in a network, and can be used as an identity in routing or switching data, even among different networks.

2.1.6 Interface Cards

Network interfaces are also called Network Interface Cards (NIC) or Adapters. These cards have to follow certain criteria, which are mentioned below:

- The cards should be compatible to the physical or data-link protocols. For example, if the network is 10baseT, the cards should also follow the same set of standards.
- The cards should be compatible to the slots. Every slot has specifications like speed (data rate), capacity (number of bits in data), number of wires, and connections in the inside motherboard. Slots can be of various types in systems. So before the NIC could be installed to a slot, the slot specifications should be very clear.

2.1.7 Programmable Network Interfaces

To meet the performance requirements and functionality of the designed network, programmable microprocessors on network interfaces (PNIs) have to be used, and they

can be customized with software that is domain-specific. So the cost/performance measurements are greatly favored.

2.1.8 Workload for PNIs

It was already mentioned that network interfaces are used for moving information or data with switches or routers. In this case, they just act as a channel or link for data or packets to move on. But PNIs have something more than this basic functionality. They actually have processing capability on the data packets that move through them. This is called as workload for PNIs. There are two categories that have to be mentioned here.

- Application specific packet processing routines
- Execution environment (Store-Process-Forward)

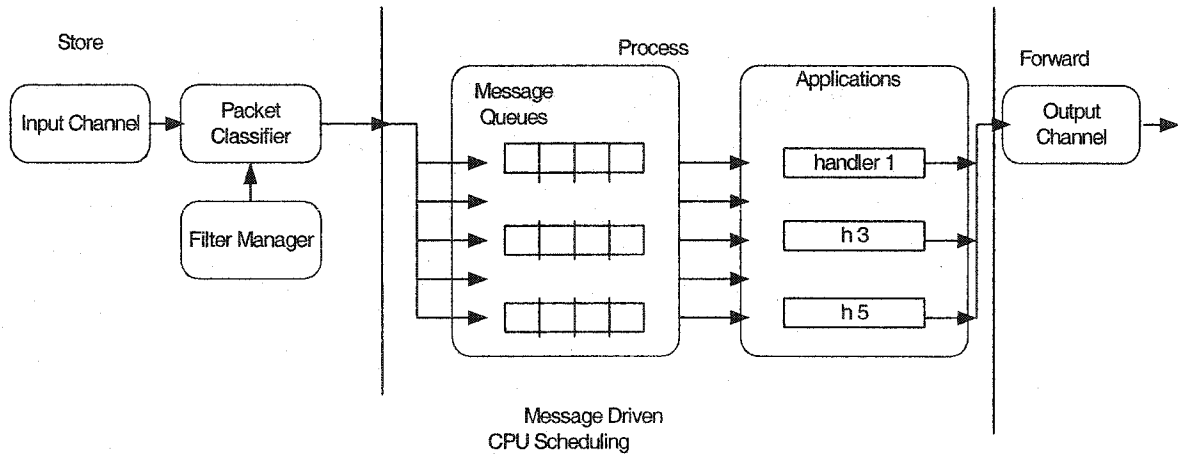


Figure 2.1: Network Interface Design Example

2.2 Switch Fabric Architecture

In this section, traditional switch designs are discussed, along with factors like sharing the load, and the methodology of connecting crossbars with interfaces (both input and output). Also, the concept of Multi-Stage Interconnection networks is defined in brief.

2.2.1 Traffic Bottlenecks and QoS: Main Concern in Switch Designs

Quality of Service (QoS) is a very important issue that needs to be taken care of in switching system design. As the number of packets transmitted as well as the speed of

transmission increase, quality becomes even more important so that service is not lost as a compromise to system betterment.

In order to solve or reduce the concerns of QoS, it is very important to understand the architecture of the switching system. A very conventional switch fabric model is presented in *Section 2.2.2*.

2.2.2 Traditional View of Switching Systems

As shown in *Figure 2.2*, a typical switching system consists of a switch fabric and a number of network processors connected to various ports and physical interfaces.

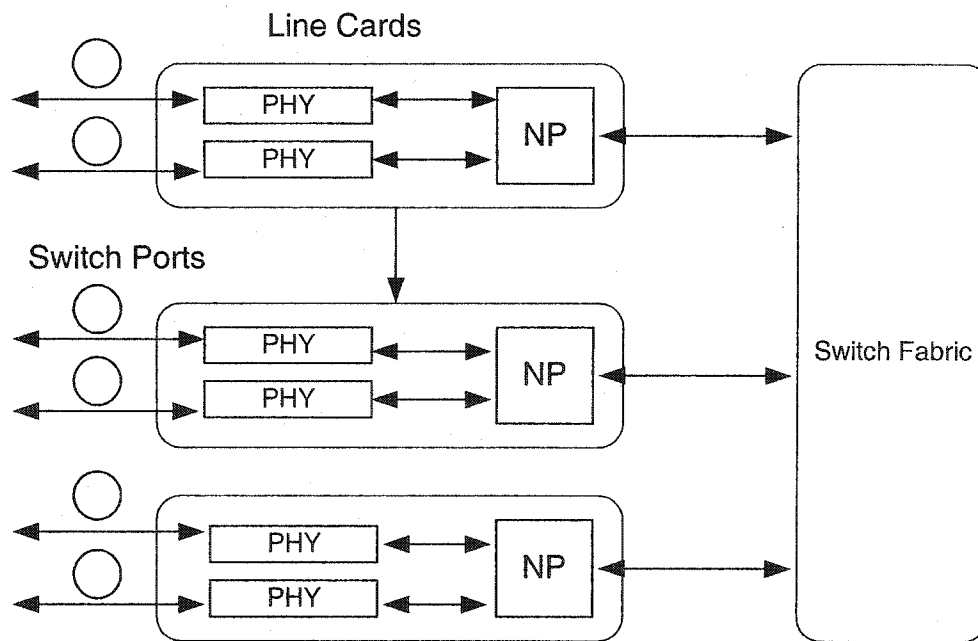


Figure 2.2: Conventional Switch Design

Packets or cells enter the system through the ingress ports, get transmitted through the switch fabric, and leave the system through the egress ports. When packets enter the system through the ingress ports, port processors perform some authentication operations, most important of which are:

- Destination address
- Specified Quality of Service (QoS) required
- Alterations, if necessary, like encapsulation, TTL (time to live) changes, encryption or decryption, etc.

The QoS specifications normally insist when the packets have to be transferred from the ingress to egress ports through the fabric. Systems are so designed that there is no incremental delay in this transfer. Packets move with a speed that is equal to the port-line rate. At the egress ports, there are different traffic shaping algorithms that insist when each packet needs to leave the switch, after being properly queued. This is called output port queuing or output queuing model. This can be seen in all conventional switches.

This exactly means that the ingress ports should send all packets without any delay, and no queuing should be done. The egress ports receive all the packets at the maximum line speed and perform the queuing operation, and it can be seen that there is no delay in any packet until they get out of the switch fabric.

2.2.3 Sharing the Load

The Shared Memory Architecture (SMA) is in force for sharing the load. It will suit particularly at the egress ports where the output queuing model is present. This means that there is a global memory available at the output ports, and those packets in the queue can take any of the egress ports, since memory is available globally or being shared.

The bandwidth of the memory should be high, for scaling the performance of this system. Hence the memory design of high bandwidth is very important. Care should also be taken that the IC pin numbers do not become very high, when the bus width increases with bandwidth.

This is one problem that exists in the SMA type. When more bandwidth is needed, one must make it passive by increasing the memory width as well. As days progress, the bandwidth requirements drastically increase, whereas memory widths are not that enhanced at such pace. So scaling the performance is not achieved to great heights. As a result, shared-memory switch fabrics currently would not scale beyond 20 Gbps of total line-end bandwidth.

2.2.4 Alternate Designs

The SMA memory design problems have led to the alteration of switch architecture, and lately, the Input-Queuing model is widely used. This model actually eliminates the

fact that packets can take any egress port after crossing the switch. Making the packets move at a pace greater than the port-line rate makes this possible.

2.2.5 Multi-stage Interconnection Network (MIN)

MIN is a structure that has multiple switches embedded in a single fabric. Since the number of inputs is high, the complexity becomes more. Also, more arbitration, queuing, and shaping makes the QoS come down drastically. In addition, since the packets take multiple routes before reaching the egress ports, latency is also increased. Hence this approach has not been very effective.

2.2.6 The Crossbar Approach

Crossbar fabrics, till now, have been considered to have the best architecture and performance in switching. They have a single-stage structure, but operate with high bandwidth. Space Division Multiplexing (SDM) is employed here, in contrast to TDM in the system explained before. Bandwidth needs to be maintained for only a single switch port, and so the aggregate bandwidth is way above the shared-memory switch fabrics as in MIN.

As latency is inversely proportional to bandwidth, crossbars have less latency. They can be scaled to operate in the tera-bit range lately, more than mega or even giga-bit ranges.

A good crossbar system must operate with different types of data forms, namely IP packets, ATM cells or multiplexed byte streams. The system might need fixed length fabric cells for this purpose. This requires segmentation and reassembly (SAR), which would cost a bit more, but useful.

In a crossbar fabric, cells or packets get queued at the input port, hence termed as input queuing model. Then, the arbiter (which can see the state of all input queues), decides the order of transmission of cells or packets. This is made possible with the QoS needed for each transmission, and the state of output queues (this knowledge is given by the egress ports to the arbiter). In this way, transmission or flow of packets takes place from ingress to egress ports through the fabric that consists of an arbiter.

For flexibility and high QoS, the cells are arranged or queued on the basis of destination address and class. In cases where cells need broadly similar QoS, they are placed in Virtual Output Queues (VOQs). In this way, care is taken such that the Output Port Processors (OPPs) are never devoid of cells, and keep transmitting cells that were earlier queued by the Input Port Processor (IPP), arbitrated and transmitted by the crossbar.

Latency and overhead can also be greatly diminished by enhancing the width of the switch fabric pipes. This makes the crossbar, a better switching system. The block diagram can be seen in *Figure 2.3*.

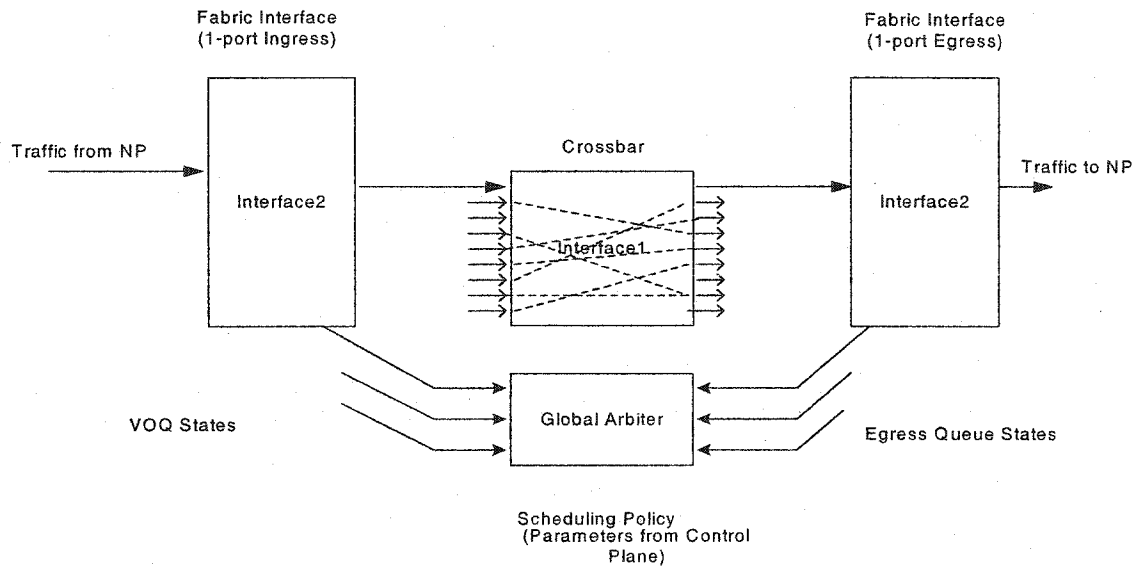


Figure 2.3: Crossbar and Network Interfaces

In crossbar architecture, the single (global) arbiter makes sure to balance the QoS specifications at the wire speed. Bandwidth is not wasted, simply because the arbiter has a global view of the traffic in the system. So the system efficiency is highly enhanced (even up to 98%).

Crossbar has always been designed to have output queuing and QoS associated with it. But input queuing can actually be combined with an arbiter. The arbiter can make switching decisions for every interval, say 20 or 30 ns. It is also supposed to transmit data with guaranteed bandwidth and bounded latency.

2.3 Functionalities of a Switch

As it has been discussed thus far, a switch is a networking device that is used to transfer or switch data from one point to another. In this thesis, a switch system with some unique functionality is developed. The design or the architecture of the switching network as well as a single switch element is presented. An analysis showing the performance as well as comparison of this network with some other types would be described. In order to design this switch of highest functionality and most efficient performance, various parameters for scheduling and shaping are studied and analyzed, to find the best one in each. These parameters are then incorporated to design the IPP.

Various networking architecture have been proposed in recent times. All of these would include transmission of data from one point to many (multicasting), as well as routing data in various lines or links in a network. Some of the factors that would greatly characterize switch systems are buffering, multipoint connectivity, speed, performance, cost, reliability, scalability, fault tolerance, etc.

The background information for this system stems from numerous research publications. Chao [2] gives an idea about the architecture and design for regulating the traffic mainly in terms of its scheduling. The advance product brief [3] of Lucent technologies' switch element explains the implementation of a shaper in the input port processor block. Certain other books [4, 5, 6, 7], written by eminent authors in the field, help in gaining insights in variegated technologies that are embedded in this system

design. The most recent one is that in which Mir [1] introduces a new switching fabric called the Spherical Switching Network (SSN) for high speed applications in next generation networking. This forms the premise of the thesis. This network is being designed and implemented here. The performance evaluation and comparison delineates that SSN outperforms all its peers currently employed in real-life switches and routers. This paper was very helpful in the conceptualization of the thesis itself. Some other papers [8, 9] give an idea about the phenomenon of queuing in input processor and control of congestion.

Some of the terms that need to be defined, and would form major topics of discussion in this thesis, are:

Multicasting: This is the process of sending data from one node to a group of nodes. The difference between multicasting and broadcasting is that in broadcasting, transmitting a packet from one node to all other nodes in a network takes place, whereas in multicasting, transmission of a packet to a subset of nodes in the entire network is done.

Scheduling: This is the process of sorting out packets at the appropriate time. For example, when multiple packets are destined for the same output or node, this phenomenon will determine which packet has to be sent at what time.

Shaping: This is the process of streamlining the flow of packets or traffic across the network. The main reason for congestion in any network is the capricious flow of traffic.

This is transformed to a more predictable rate, by conforming the flow to a specified behavior.

Encapsulation: This is the process of translating an Internet Protocol (IP) packet to ethernet. This is similar to envelope stuffing. This is more or less a segregation process of a packet into various fields like preamble, ethernet address of destination, ethernet address of sender, packet length, error checking information, etc.

Splitting: This is the process of splitting big packets into smaller ones for easy transmission. It adds the speed of transfer and avoids congestion at the input buffer. The split packets take different routes in the network, and so, the process of reassembly is vital at the output port processor.

The various questions that need to be asked for authenticating the validity of this thesis can be answered as follow. First of all, this is a new network, both structurally and functionally. One cannot find too many publications or information about this network and system. Secondly, it exhibits cutting edge technologies, and is in par with the latest technologies and trends. Also, it does provide scope for future work and development. To be more specific, the primitive design of the system is designed here. This can be expanded to numerous levels or planes. Statistical analysis and modeling of the system are other ventures. Last but not least, this innovative research can be applied in the commercial world as well, which adds to its validity. An analysis that delineates the study of various algorithms, determining the best ones, and implementing the same in the

design reveals the validity to the tertiary level. The following sections would deepen down into the actual analysis, design, and architecture.

2.4 Traffic Management

This section will introduce some basic concepts regarding traffic control that allows the sender to establish affinity with the traffic contract, and obtain the required Quality of Service (QoS). An overview of traffic and congestion control functions is as follows: it consists of terminal equipments and switches that shape the traffic flow to conform to traffic parameters. These traffic parameters are checked with usage parameter control functions by the ingress ports of this network. This is carried at the user network interface. Similarly, arriving packet flows are checked from the prior network by the network parameter controls. This takes place at the network-node interfaces. Functionalities like connection admission control, flow control, traffic shaping, resource management, etc. are performed. These functionalities tend to operate on various time scales including cell time, cell time level, round trip delay, etc.

Numerous algorithms exist, when it comes to traffic and congestion controls. Some of them are leaky bucket, sliding and jumping windows, generic cell rate, and guaranteed frame rate algorithms, etc. When it comes to flow control, there is available bit-rate closed loop flow control, conformance checking, point-to-multipoint bit rate algorithms, etc.

2.4.1 Traffic Control

In order to achieve maximum benefit from guaranteed QoS, traffic shaping was already mentioned. The need for shaping occurs when any receiving device or application accepts a certain number of packets, and starts discarding later. Some of the shaping implementations widely used are buffering, spacing, scheduling, leaky and token buckets, framing, policing, packet rate reductions, etc.

Buffering is done to ensure that packets do not violate traffic parameters by storing them to the leaky bucket's capacity by violating the traffic parameters. *Spacing* is the process of delaying cells by holding them from various connections, minimizing their variations, and scheduling the departures to conformed parameters. *Peak packet rate reduction* is the phenomenon of making the sending terminal work at a peak rate just less than that of the traffic contract, and reducing any violation to conformance. *Burst length limiting* is done to restrain or decrease the transmitted burst length than the maximum burst size to shape data. *Source rate limitation* is said to be performed, when the actual source rate is limited in some other method. This is more an implicit way of traffic shaping. *Framing* is a widely used shaping method that super-imposes a multiplexed structure onto the packet sequences, and employs the frames of data to schedule packets that need delay variation in a controlled way. In addition to these, the generic flow control mechanisms include Connection Admission Control (CAC), resource management using virtual paths, etc.

Traffic contract, in essence, can be written as the network establishing certain rules or protocols like a set of QoS parameters, a set of traffic parameters, conformance checking rule, and network definition of compliant connections. It is an agreement between a user and a network with regards to the quality of service that is assured. There are various QoS parameters like packet transfer delay, packet delay variations, packet loss ratio, etc. The conformance rule also specifies and associates a tolerance parameter with the peak-rate. The leaky bucket algorithm checks the flow of incoming packets at rates corresponding to peak rates of packets. It discards excess incoming packets making sure there is no overflow.

2.4.2 Traffic Contract

As mentioned in the previous section, the traffic and congestion control schemes operate on the basis of various time scales. Also, the network usage parameters are obtained by various policing methodologies including bit tagging, packet discarding, as well as leaky and token buckets. Adaptive traffic control, which is like closed loop flow control, provides high-performance and fairly allocated service to all switching and routing applications. Methods like priority queuing and discard thresholds support a range of QoS parameters that suit the desired applications.

2.4.3 Congestion Control

Congestion can be defined as the condition when the demand for resources exceeds the available resources in a given period of time. In terms of networking, it is the condition where the offered load to the network exceeds the network design limits for guaranteeing the QoS specified in the traffic contract. The impact of congestion in a network has already been studied with the congestion control performance in parallel. Congestion management has various methods including resource allocation, connection admission control, and usage parameter control, with each method having numerous parameters, and methods of implementation. Congestion avoidance, in turn, attempts to avoid severity in congestion while keeping track of the offered load as well. Some of the methods to achieve this are explicit forward congestion indication, usage parameter control, call blocking, and flow control mechanisms like window-based flow control, rate-based flow control, credit-based flow control, etc.

2.5 Traffic Engineering

This section will discuss the aspects of traffic engineering in brief. Source modeling, performance evaluations, and switch performance modeling are various aspects that need to be discussed. As the model gets more and more complex, evaluations such as these become more complicated. Since this thesis consists of modeling the switch fabric by a probabilistic (stochastic) model, some basic queuing theory concepts are presented in this section.

2.5.1 Queuing Theory Overview

Queuing theory is the most widely used modeling method in a mathematical or real-time system. This theory is utilized in almost every research venture and publication, to analyze and find out the various parameters, and plot them. This thesis will have the same methodology too.

Throughout the design and analysis of scheduling and shaping algorithms, a number of queues are seen. For example, the input buffer, the scheduler (all algorithms), the shaper (both leaky and token buckets) contain queues. Any network transmission will have a number of packets lined one after the other to reach its destination. Therefore, a general type of queuing model is assumed for all cases, thus making the calculations less complicated. A very basic and most commonly used queuing model is briefly enumerated below:

Some of the common parameters when it comes to networking and queuing of packets are: *Burstiness*, the ratio of peak rate to the average rate of packets, and *Source activity probability*, the reciprocal of burstiness.

The model has Poisson arrivals and Markov process when it comes to dealing with packets in a network. Poisson arrivals of packets imply that, for every increase in time interval "t", the probability of a new packet arrival is independent of the previous ones, and also independent of the packet's size. This probability is called as *inter-arrival time*

probability. When it comes to random processes, Markov process is important, which again states that the past history of packets does not affect the nature of future arrivals.

Markov process is of two types, namely:

- Discrete time Markov process model
- Continuous time Markov process model

Chapter 3 Spherical Switching Network (SSN)

3.1 Introduction

A switch is a networking device that is used to transfer or switch data from one point to another. In the paper, "*An efficient switching fabric for next-generation large-scale computer networks*," Mir discusses a switching network with some unique functionality [1]. The design or the architecture of the switching network as well as a single switch element is presented. An analysis showing the performance as well as comparison of this network with some other types is described. A summary of the paper is presented in this section. Since this thesis is all about the research and study of this network, the advantages that serve as the reasons for our selecting this network are also discussed.

Various networking architecture have been proposed in recent times. All of these would include transmission of data from one point to many (multicasting) as well as routing data in various lines or links in a network. Some of the factors that would greatly characterize switching systems are buffering, multipoint connectivity, speed, performance, cost, reliability, scalability, fault tolerance, etc.

Understanding the difference between single-path and multi-path networks is vital before getting into the design of any network device. Single-path network, as the name suggests, consists of only a single path between an input and an output port in any network, whereas multi-path network has many routes for a single input and output port.

Single path network leads to congestion and high probability of blocking. Also, in a single path network there can be single-stage and multi-stage networks. The number of stages will influence the switching characteristics of the network like speed, delay, traffic, etc.

Multicasting is a subset of broadcasting that employs one-to-many transmissions. It is employed to increase the throughput, and reduce blocking and congestion problems. Recent research focus mainly on this concept, to be implemented in any switch, router or gateway. For this purpose, buffers are included in every switch element in the network, thereby making the hardware design more complicated. This also leads to flow-control implementations because of buffer overflows. At every switching stage, the data packet has to either be discarded or deflected whenever buffer overflow takes place. The former is more widely used in industries. The author focuses on the latter in this paper.

Since focus is more on packet deflection (and not discards), this network possesses some additional features, which are as follow:

Randomizing traffic: This is used to distribute traffic in an even manner, to avoid congestion.

Re-circulation of traffic: When a packet does not find itself destined, it is re-circulated by the switch element. Every time it happens, the priority is increased, so that it reaches the destination sooner than later.

Increasing Speed-up factor: Speed up factor can be defined as the ratio of the internal link speed to the external link speed. The design also takes into consideration, the various methods to enhance this factor.

The network proposed here helps in reducing the number of deflections due to a colossal number of interconnections, and thus increasing the speed up factor. The organization of the paper consists of the architecture (design) of the switching network and a single switch element, along with an analysis of the performance using stochastic (probabilistic) models for queuing-deflection processes. Also the superiority of this network with its peers or family is also mentioned.

3.2 SSN Architecture

SSN architecture is basically a regular mesh, which is a collection of horizontal, vertical, and diagonal rings, in a bi-directional fashion. It employs the self-routing scheme, that consists of cyclic connections physically, thus reducing the number of deflections.

The design consists of switch elements, each of which is a 9 X 9 crossbar with a local controller. The bi-directional links are composed into nine pairs for each crossbar. Of these, eight pairs are used internally, and the ninth pair is used externally. The word internal refers to data and connections within the switch element; external refers to links

and data through the entire network. These links can also be used to carry internal and external traffic respectively.

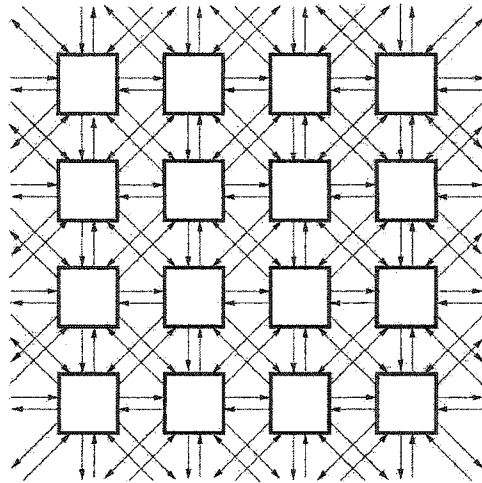


Figure 3.1: A 16-Port Proposed Spherical Switching Network

The main feature of SSN is that, it does not possess any internal buffer. In other words, there is no buffering within the network. This feature will greatly enhance contention resolution. So it is primarily concerned with deflection mechanism. When a packet tries to reach the destination, the switch element performs arbitration of all incoming packets. The arbiter selects one packet out of all, and then transmits it to the destination. All the remaining packets that lost the contention are deflected through various other links throughout the network. Every time a single packet is deflected, its priority field gets incremented by one. This means that when the same packet contends at a different time, it is going to be preferred by the arbiter. The advantage here is that no packet is discarded, which is contrary to most switching elements today. Even when

there is congestion, packet discarding is strictly prohibited. On the other hand, they are just misrouted temporarily. This directly implies that there are multiple equally desired outputs possible, viz. more than one shortest path from same source to destination.

3.2.1 Self-Routing Feature

The self-routing scheme that is introduced in SSN is very simple. An index number is assigned to every switch element in the network. For example, in a 16-port network, the indices range from [00-00] through [11-11]. Decremental or incremental addressing, depending on whether the flowing traffic or data is directed towards decreasing or increasing indices respectively, performs this routing scheme. There are four different cases:

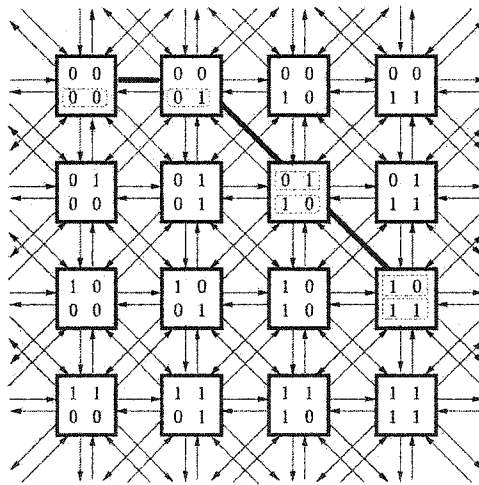


Figure 3.2: Routing Scheme in Spherical Switching Network

- Straight Horizontal Routing
- Straight Vertical Routing
- Straight Diagonal Routing
- Combination of Diagonal/Horizontal or Diagonal/Vertical Routing

The premise of all these schemes is that a data packet takes a certain route (shortest preferably), and reaches its destination. In the first scheme, the second two index-bits of the source address is increased or decreased hop-by-hop, until the actual addresses of source and destination become the same, where routing gets completed. This mainly depends on whether the index number of the source is greater or smaller than the index number of the destination. In the second scheme, the increment or decrement is done on the first two index-bits of the source address, but the process is just identical. In the third scheme, accomplishment of routing is attained when both values of the first as well as second two index-bits get incremented or decremented. The final scheme is a combination of either first and third or second and third schemes. Obviously in the fourth scheme, there could be more than one preferred route. As mentioned earlier, these routing schemes apply to transfer packets to their destinations by an advantageous route. But when congestion occurs, deflection comes to the picture, and incrementing the priority of the packets becomes inevitable.

3.2.2 Proposed Switch Element

The diagram of the proposed switch element is shown in *Figure 3.3*. It consists of a 9 X 9 crossbar and a controller (with deflection and priority features), Input Port Processor (IPP) with input buffers, and Output Port Processor (OPP).

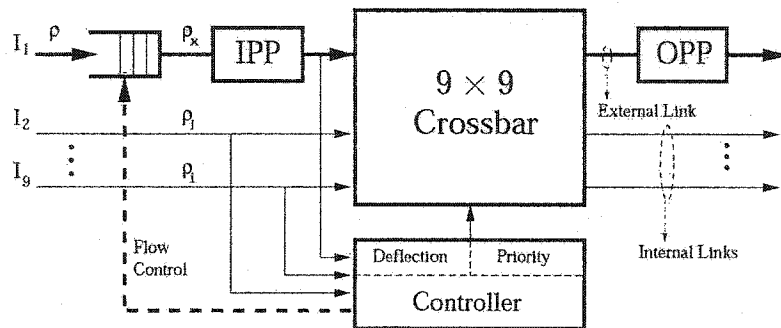


Figure 3.3: The Proposed Structure of the Switch Element

Buffering takes place at the external input port. Routing takes place within the network. For this purpose, the eight incoming and outgoing internal links inside the switch element are present, and one incoming and outgoing external link joins each switch element to the IPP and OPP. This odd link can also interface the switch with an external transmission link. This is the input port buffering structure in which, each switch element is a 9 X 9 crossbar.

Each incoming packet is temporarily stored in an input buffer slot. They are then arbitrated by the switch element, after which the crossbar switches them to the destinations. The operation of the IPP also lies in converting the input packets into the

local network format. The nodes also have IPP look-up tables for routing purposes. The IPP accepts input packets only when there are empty slots. Otherwise, it just discards them. Once the packets are accepted, they undergo the queuing and routing operations, and reach the OPP. The output port processor plays a vital role in re-sequencing packets. This is performed only for mis-ordered packets, which get deflected and prioritized once they lose contention. This leads to variation in delay.

The routing decision is made in this way. The packets arrive, and the control information is stored in control circuits of each switch element. Also, flow control is not done to every switch element, but only to the local processor. This means that elements must receive multiple packets every cycle, or at least should have the capability to do so.

3.3 Performance Evaluations

The performance of the Spherical Switching Network (SSN) was evaluated by considering a stochastic model. Each input port generates a packet independently with a fixed probability “ p ”. The arriving packets are assigned a random destination address. At the external buffer of a switch element, “ b ” slots are assumed. After the packet is generated, the input queue is incremented. But if the number of input buffers is less than the number of incoming packets, the packets are discarded. Only when the arbiter accepts the packet, they enter the switch fabric and get routed. This contention of packets has two distinct cases: contention for an external link, and contention for internal outgoing links. If a packet loses contention, it is deflected, which in turn depends on two

parameters. First, a packet is authenticated to see whether it is destined to traverse the switch element. Second, the location of the packet is checked to see if it has already reached the destination.

3.3.1 Performance Results and Comparison

In this section, selected results from simulated outputs are extracted and compared with four other types of networks. Actually, the author has considered five different networks, namely: The SSN (with no internal buffers), the Clos network (two internal buffers per port), the Shuffle network (two internal buffers per port), the Banyan network (two internal buffers per port), and the Manhattan Street network with two internal buffers per port.

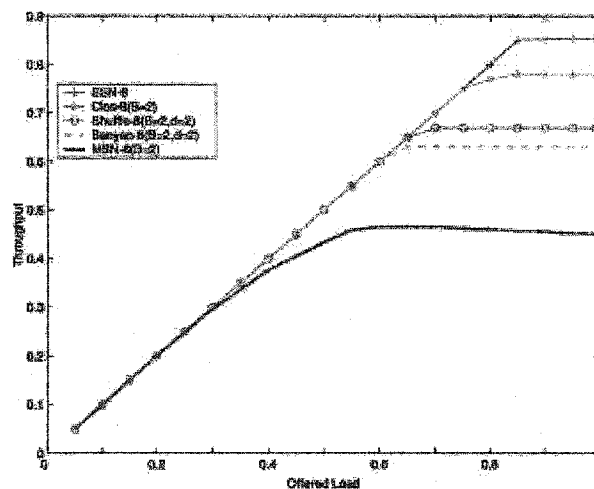


Figure 3.4: Comparison of Throughput (vs.) Offered Load for the Five Networks

The plot in *Figure 3.4* clearly portrays the direct proportionality of throughput with the offered load, but after a threshold ($\rho=0.4$ in this case), it starts to flatten out due to the system attaining its full capacity. Beyond this point, it can be noted that there is no more increase in throughput. Clearly, the SSN has better throughput performance than the other systems mentioned here.

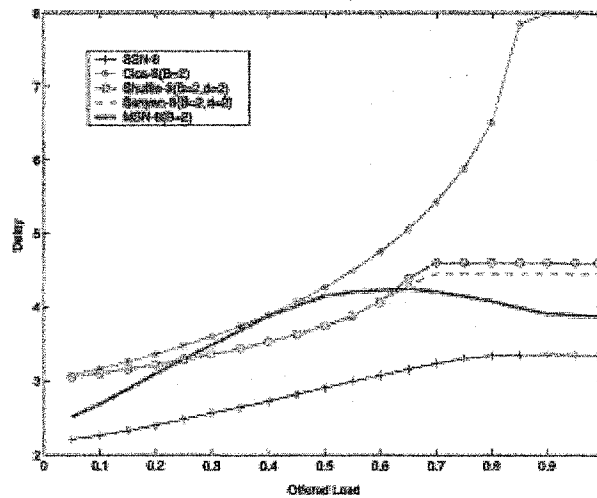


Figure 3.5: Comparison of Delay (vs.) Offered Load for the Five Networks

The plot in *Figure 3.5* shows the relation between the offered load and delay. As always, the delay reaches a peak value whenever the network capacity is reached. Here again, SSN shows a better performance of delay when compared to the other networks.

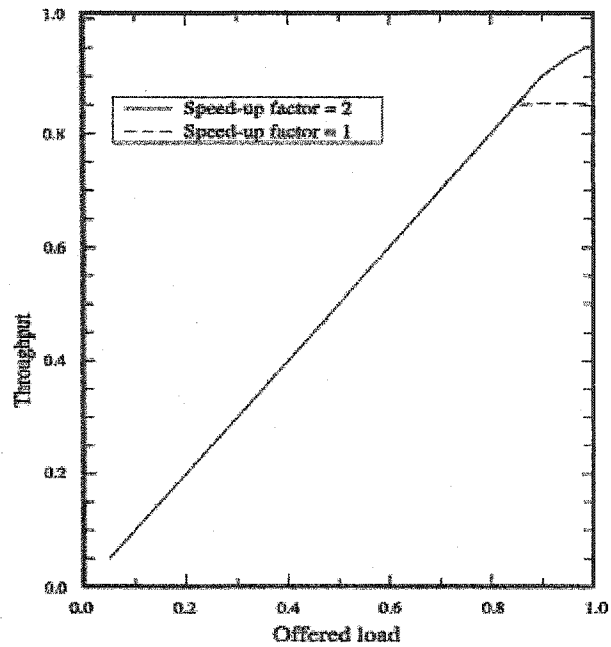


Figure 3.6: Effect of Doubling the Speed-up Factor (Speed Advantage) on the Throughput (vs.) Offered Load

The plot in *Figure 3.6* shows the effect of doubling the speed-up factor on the throughput versus offered load. This gives a speed advantage, in the sense that the speed of the network operation being influenced is given.

The results of the performance evaluation reveal that this network clearly surpasses the other four in performance. It also shows low packet delays, with the average number of deflections being reduced. Another simulation result portrays the quicker impact of the speed-up factor on the system throughput.

The scalability and complexity of 256-switch fabrics is also simulated. This output delineates clearly that the spherical switch has highly scalable architecture due to the nature of expandable architectures.

3.4 *Reasons for Choosing SSN for Thesis*

The spherical switching network topology proposal is summarized in [1]. It is actually a toroidal mesh network in which a request is deflected on alternate output links, if the desired link is already utilized. This network implements a simple self-routing feature, and is constructed with fixed 9 X 9 switch elements, independent from the network size. One of the nine pairs of links carries external traffic, while the rest of them carry internal traffic only. Arbitrating all the incoming packets performs the contention resolution between packets, and deflecting them through other links is done, if and once they lose the contention. Only one packet is thus transmitted through that route. This is called deflection plus incremental priority of packets. Since there is no internal buffer in this spherical network, the system complexity is greatly reduced. The results of the performance evaluation clearly shows that this network outperforms buffered Banyan and Clos networks as well as buffered MSN and Shuffle networks. Since the number of deflections is diminished, the system portrays a mediocre delay or latency of packets. The average number of deflections, and probability of deflection plots with the network size gives a feel of the network's superiority. Also, the speed-up factor impact on the system throughput is analyzed and graphed. It is explained that with a minimum speed ratio of two, this system operates with great efficiency, while for all other systems, to

show the same percentage in efficiency, the minimum speed ratio should always be greater than two. The convenience and low-cost expandability of this network is beneficial for resizing the network according to the application specified for any future development.

The performance evaluation and comparison plots clearly lineate that this network is superior to its peers, which are currently implemented in switches and routers. Important parameters like throughput and delay are seen to have the best performances with respect to offered load. The speed-up factor getting doubled on the throughput with respect to the offered load is one of the primary advantages of SSN. The packet losses incurred are also not to a maximum. These are the major reasons why this spherical switching network is studied, implemented, and analyzed in this thesis.

Chapter 4 Hardware Design of Input Port Processor (IPP)

The following section is the first of the two most important sections of this thesis, the design part. The architecture of the Spherical Switching Network (SSN) was described in the previous section, and it was mentioned that the three most important blocks that make up the architecture of a switch element are the Input Port Processor (IPP), crossbar, and Output Port Processor (OPP). This thesis is associated with just the IPP part of it. The basic functionality of IPP is to accept packets from the network; queue and buffer them temporarily, and once the ports become free, transmit them into the network. The IPP also performs some major functions like multicasting, scheduling, and shaping of the incoming packets. The design of this system has to be done with these three functionalities in mind, with separate blocks for each. Each of them will be considered, and the technology and individual protocols associated with them will be explained. Also, a summary of the various other hardware blocks that make up the design of the IPP will be mentioned in brief.

4.1 *Multicasting*

Multicasting on a buffered switched network can be implemented in many ways. The simplest approach from a hardware point of view is the flooding technique. A switch will send all multicast packets as if it were a broadcast packet, and allow the end nodes to

make a decision if the packet is for them or not. At each node, the multicast packet gets replicated, and is sent to every link directly connected to it, except the link from which the packet was received. This technique does not scale very well, so there are several algorithms that improve network efficiency for multicasting. The one which will be of interest is the Reverse Path Multicast (RPM). By having nodes announce to the network that they want to receive multicast packets, switches and routers will know where to send multicast packets by looking at a link-state table or routing table for each switch. For this method, only the nodes or switches that need the multicast packets will receive them.

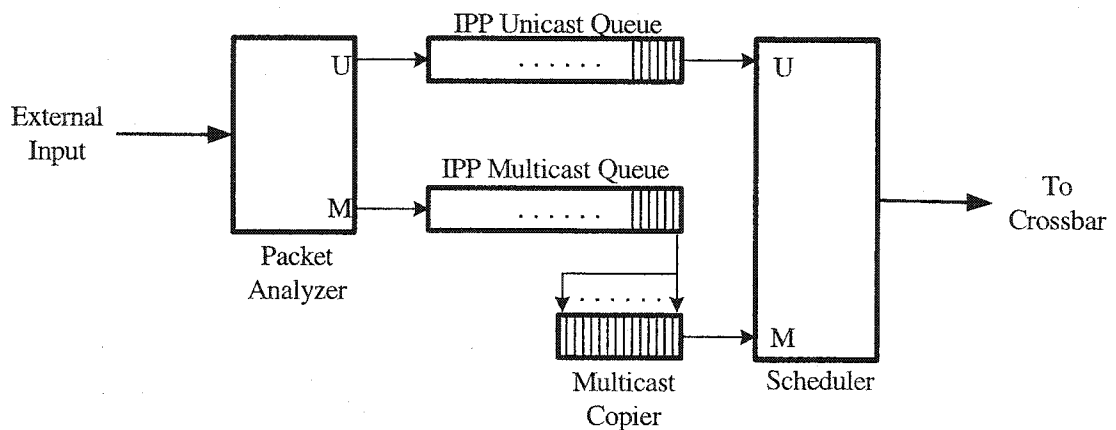


Figure 4.1: IPP Block Diagram for Multicasting Queues with Relationship to External Inputs and Scheduler.

Multicasting in SSN architecture is not a trivial task. The lack of buffers at the crossbars makes packet duplication on the fly an impossible task. This eliminates the possibility of flooding the network with broadcast packets as an option. Also the deflection scheme of SSN causes packets to take a variety of different routes to reach a

node, which interferes with the flooding approach. There is no guarantee that packets will reach all points in the network because deflections are random. Modifying RPM allows multicasting to be done on a SSN network efficiently, but with more complex hardware. In order to solve the packet copying, the IPP will have to copy packets when they enter the network from an external source. Only one packet may enter the network at a time, so there will be competition between the copied multicast packets and the incoming external packets. To facilitate competition, a multi-level queuing system will be needed to assure quality of service for unicast and multicast packets, as shown in *Figure 4.1*.

Once multicast packets have been replicated and are ready to enter the network, the scheduler must prioritize traffic. Scheduling is required to handle multiple types of packets, broadly classified as Real Time Packets (RTP) and non-Real Time Packets (non-RTP). RTP include video and audio data, which have a low delay requirement, and non-RTP include normal data packets, which do not have a delay requirement. Scheduling ensures fairness to the different types of packets, and provides Quality of Service (QoS).

4.2 Scheduler

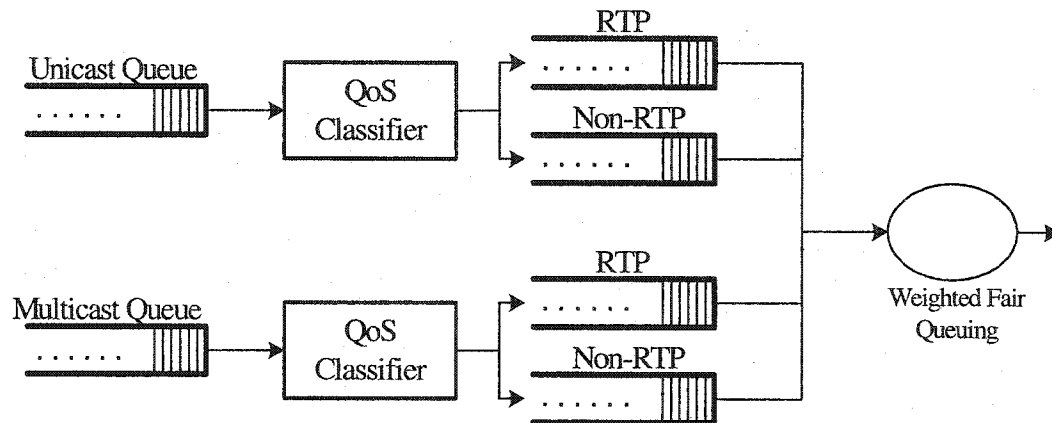


Figure 4.2: Block Diagram of the Scheduler

The QoS classifier, shown in *Figure 4.2*, classifies the packets into real time and non-real time packets. The packet type is deduced from the 4-tuple <source IP address, source port, destination IP address, destination port> in the packet header for unicast packets. For multicast packets, the 2-tuple <source IP address, source port> is used to identify the type of the packet. The QoS classifier contains a table of mappings between each 4-tuple (and 2-tuple in case of multicast packet), and its packet type as shown in *Table 4.1*. Reservation protocols like RSVP can be used to populate the QoS classifier table. Packets that do not have a matching entry in the classifier table will be assigned to the non-real time queue.

<i>4-tuple < source IP address, source port, destination IP address, destination port ></i>	<i>Packet type</i>
192.138.10.128, 23, 172.197.189.1,45	Real Time
187.67.37.45, 180, 172.38.45.12, 8000	Non-Real Time

Table 4.1: QoS Classifier Table

Separate QoS classifiers are used for unicast and multicast packets. The inputs to these two classifiers are from the unicast and multicast queues. After the classification, the packets are sorted into four different queues: unicast RTP, unicast non-RTP, multicast RTP and multicast non-RTP.

Once the QoS classifier populates the queues, different techniques can be used to schedule the packets. The Round-Robin technique is the simplest; however, real-time guarantees cannot be provided by this technique. Priority queues can be used, but it is still not an optimal solution, since it can cause starvation in low priority packets. A well-balanced technique, Weighted Fair Queuing (WFQ), is used in order to ensure fairness and to avoid starvation.

4.3 Traffic Shaping

The primary reason for network congestion is the capricious behavior of traffic. Most network problems would not exist if the traffic was more predictable. This becomes the

reason to convert a busty stream of data into a more uniform one, process known as Traffic Shaping. It becomes easier to allocate resources in a switching network like bandwidth availability and maximum delay, so that the objective of delivering the Quality of Service (QoS) is guaranteed. This also assures that all the resource management strategies are withheld. It controls the rate of data going through an interface, and can be adjusted in such a way that the transmitter and receiver work with data at a same rate, thereby avoiding speed mismatch. The most important terminologies or parameters related to the shaping process are:

Mean bit rate: The mean rate specifies how much data can be sent or forwarded per unit time on average.

Peak bit rate: Burst size -- Also called the committed burst (Bc) size. The burst size specifies in bits (or bytes) per burst, the extent of traffic that can be sent within a given unit of time, without creating scheduling concerns. For a shaper, burst size specifies bits per burst; for a policer, burst size specifies bytes per burst.

Burst length and burst frequency: Length and frequency of a burst, which is the set of bits, bytes or characters, grouped together for transmission.

Burst Tolerance (BT): Traffic parameter used in VBR services; limit parameter of the GCRA.

PCR (Peak Cell Rate): The cell rate that the source may never exceed, expressed in cells/second.

VBRrt (Variable Bit Rate real time): Category characterized by tightly constrained delay and delay variation, but not necessarily a fixed transmission rate, typically used for packet voice and video applications.

Cell-loss rate: Rate at which a fixed-size packet of data is lost.

Cell Loss Ratio (CLR): A QoS parameter: the ratio of lost cells to total transmitted cells.

Cell-loss priority (CLP): A 1-bit field in the cell header that determines whether or not a given cell should be dropped by network equipment during periods of congestion. The source node or the network can set this explicit loss priority.

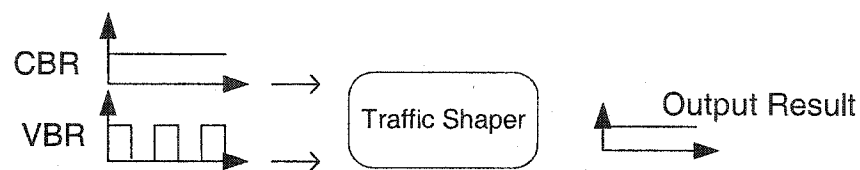


Figure 4.3: Constant and Variable Bit Rate Inputs to a Shaper

4.3.1 Reasons for Traffic Shaping in the IPP

The primary reason to shape traffic inside any network is to control access to available bandwidth, and regulate data in such a way to avoid congestion. When there is a policy that says the rate of packets cannot exceed a specified rate even though access rates might be higher, there has to be a mechanism that would control the packet flow. Also if there is a network with different access rates, the flow has to be regulated. Circumstances like these and many more, find shaping as their best solution. Shaping also prevents packet loss, and attains a specific configured rate or a derived rate based on the level of congestion.

Traffic Shaping is the conversion of turbulent flow of incoming data into a streamlined one. This process obtains conformance to a specified behavior of flow of data. Since there exists a specified behavior defined in the network, the network will be aware if it can expect packets or not.

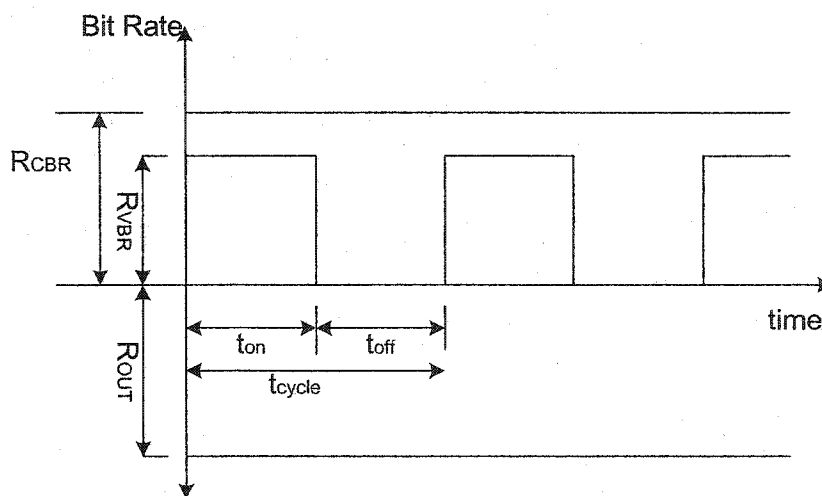


Figure 4.4: Traffic Analysis Diagram

Shaping can determine if a flow can be allowed or packets can be transmitted. The network can also monitor the flow of traffic constantly, and verify if this flow still obeys the expected intensity. This is all about regulating the burstiness and average rate of data transmission. The primary reason for employing shaping in any network is to avoid congestion, and to control access to the available bandwidth. Some of the very basic terminologies associated in this aspect are as follows:

- **Mean Bit rate:** This determines the average amount of data (or packets) that can be sent or received in unit time.
- **Peak Bit rate:** This is also called Burst size. It is the amount of data (in bits), or traffic that can be transmitted in unit time without creating any scheduling concern. This is also called Committed Burst size. For a shaper, burst size is measured in bits, whereas for a policer, it is measured in bytes.

- **Policing:** It is the regulation process of a unique data rate at which packets are allowed to enter the network.
- **Burst size and frequency:** Length or size of the burst (bits, bytes or characters integrated for transmission), and its frequency.

The two most widely used algorithms are explained in brief below. They will be analyzed thoroughly and one of them will be used in the design section in this thesis.

4.3.2 Leaky Bucket

This algorithm provides a means for converting busy traffic into a regular stream of packets, and was proposed by Turner (1986). It consists of a single server that operates with constant service time. *Figure 4.5* depicts a leaky bucket. The way this algorithm works, can be explained as follows: A leaky bucket interface is connected between the host that transmits packets, and the network. Packets from the host enter the bucket at an irregular rate. As any normal leaky bucket in real life, there is a virtual aperture at the rear side, and packets get out from the bucket through this aperture. The number of packets that leave this bucket depends on the aperture size. This size fixes the specified behavior of traffic, and makes the incoming burst to conform to this behavior.

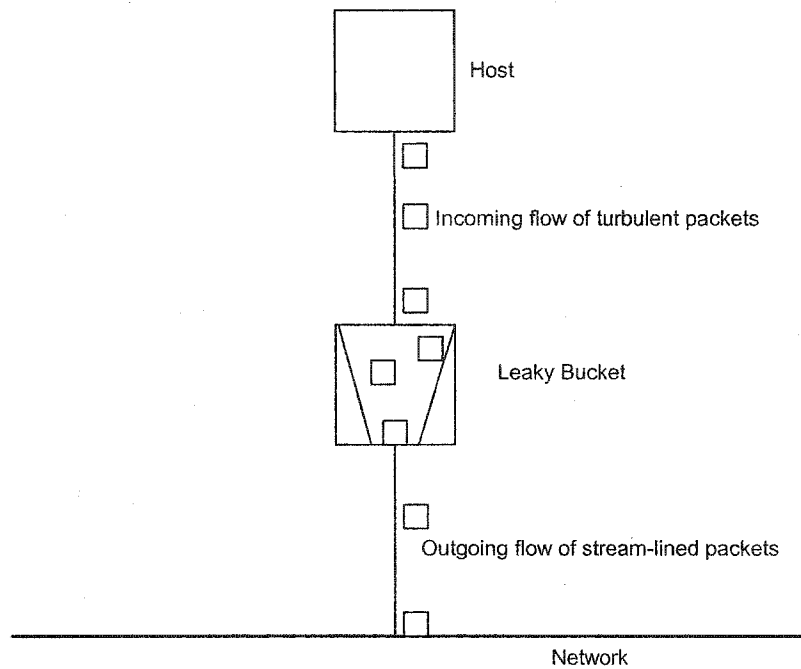


Figure 4.5: Leaky Bucket Algorithm

Thus traffic that goes out from the bucket's interface is regular and uniform, and packets are equally spaced. If the rate of outgoing packets is assumed to be " r ", then the space between each of them is " $1/r$ " time units, i.e., packets are delivered at this interval one after the other. It was already mentioned that the bucket size plays a significant role, as packets get relieved out from its aperture. Incoming packets are discarded once the bucket becomes full. This implies that once the queue (FIFO in this case) becomes full, no more packets can enter the bucket, which directly binds the maximum burst size that comes in from the host. It also controls the delay incurred by each packet that comes in, and goes out of it.

4.3.3 Token Bucket

This mechanism also provides a way to limit the input traffic into a fixed data rate and burst size. This can be better understood with *Figure 4.6*. The operation of the token bucket can be explained as follows: It consists of a token bucket interface that gets filled up with tokens at a constant rate, or every clock cycle through a token generator. The unpredictable rate packets from a host are made to pass through this token bucket. According to this protocol, each incoming packet has to acquire sufficient number of tokens that depends on its packet in order to enter the network. One token associates itself to a fixed size of data. Since the token size is constant, appropriate number of tokens gets attached to any packet size that traverses through it. If the bucket is full of tokens, only the excess tokens are discarded, but not the packets. If it is devoid of tokens, incoming packets are delayed (or buffered) until sufficient number of packets is generated. If the packet size is too big, such that there are not enough tokens to accommodate with it, then again a delay of input packets is carried over. The ideal operation of this algorithm is when there is more number of tokens than the incoming packet sizes. This is not a big flaw as tokens are periodically generated. The output rate of packets attached with tokens obeys the clock, and so it becomes regular and non-busy. In this way, by changing the clock frequency and varying the token generation rate, the input traffic can be controlled or transformed to the expected rate.

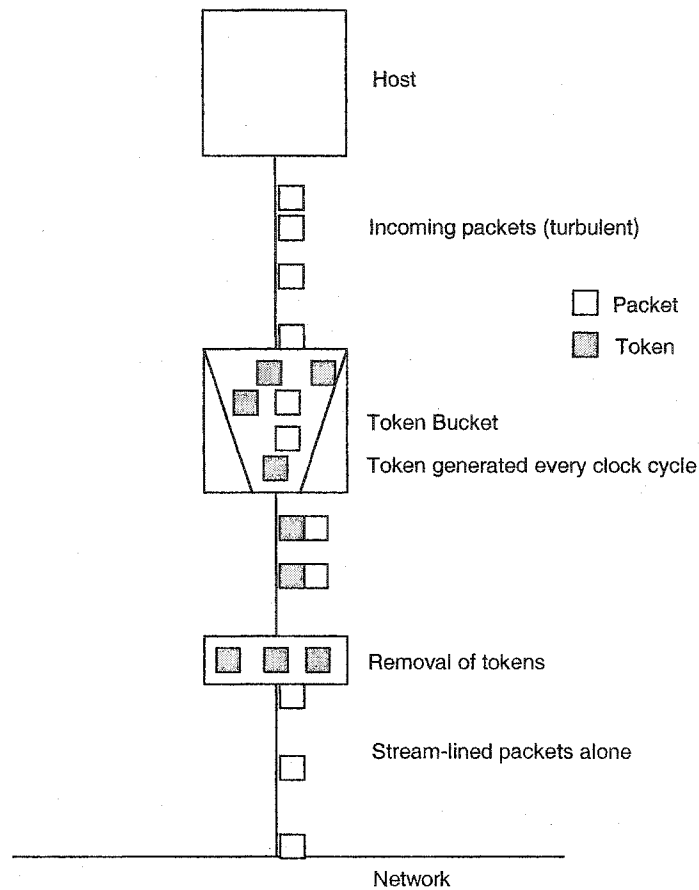


Figure 4.6: Token Bucket Algorithm

4.3.4 Reasons for Choosing Leaky Bucket over Token Bucket

Having explained both algorithms and operations, it is important to find out the most efficient one in order to implement the same to the hardware design. Both algorithms have their own pros and cons. First of all, token bucket algorithm enforces a more flexible output pattern at the average rate, no matter how busy the incoming traffic is, as opposed to leaky bucket that enforces a more rigid pattern. So it could be written that the token bucket is a more flexible one. But most importantly, when it comes to complexity

of design and analysis, the token bucket has a separate block called the token generator that operates with a clock, and a queuing regulator that attaches incoming packets with tokens, and then another block that detaches the two and discards the tokens alone. This complexity is not seen in leaky bucket as no virtual tokens are seen. Thus the leaky bucket greatly enhances the speed of operation and performance of design in turn. The primary scope of coming up with a high-speed switching network can thus be easily attained.

The entire design of the IPP can be given by the following block diagram. This design has three functionalities involved, including multicasting, shaping, and scheduling

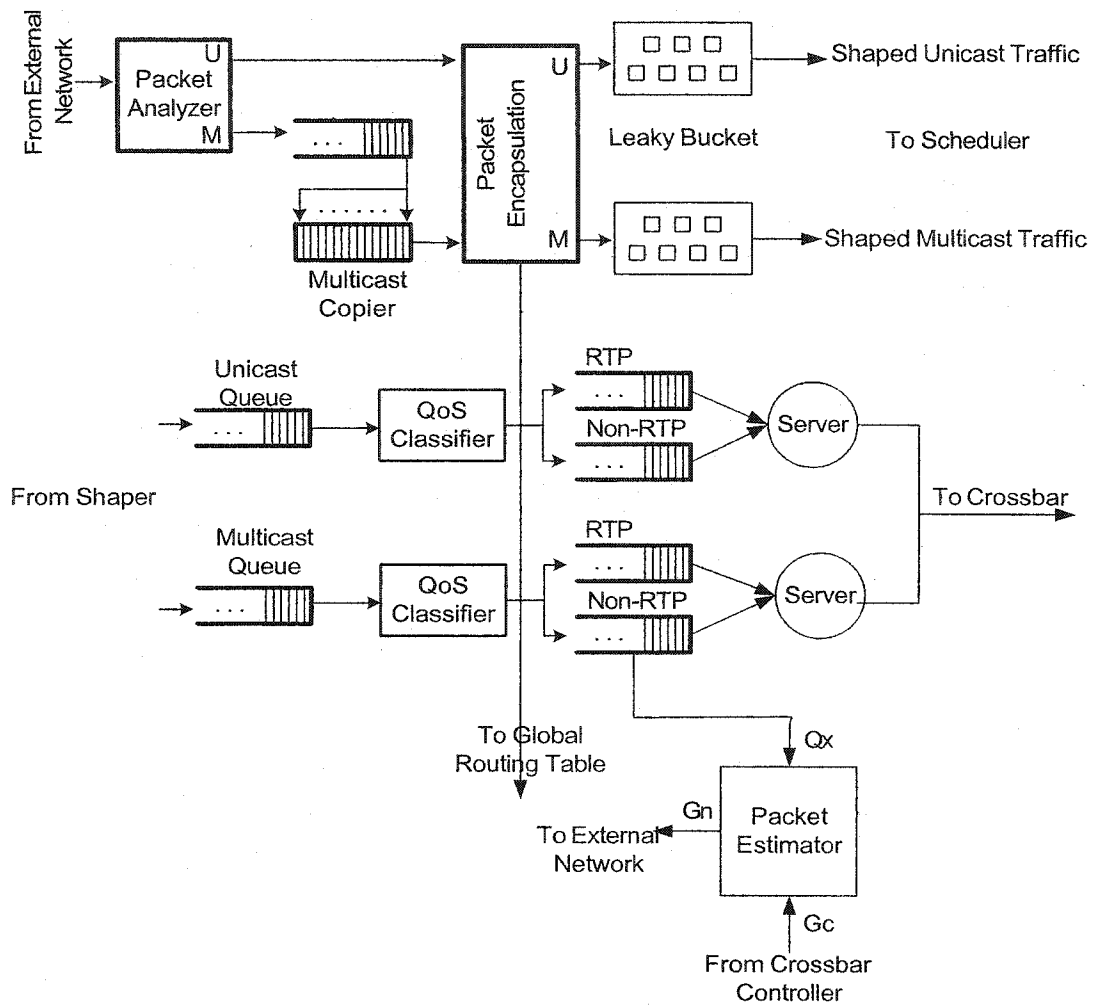


Figure 4.7: IPP Block Diagram

4.3.5 Datapath of Leaky Bucket Design

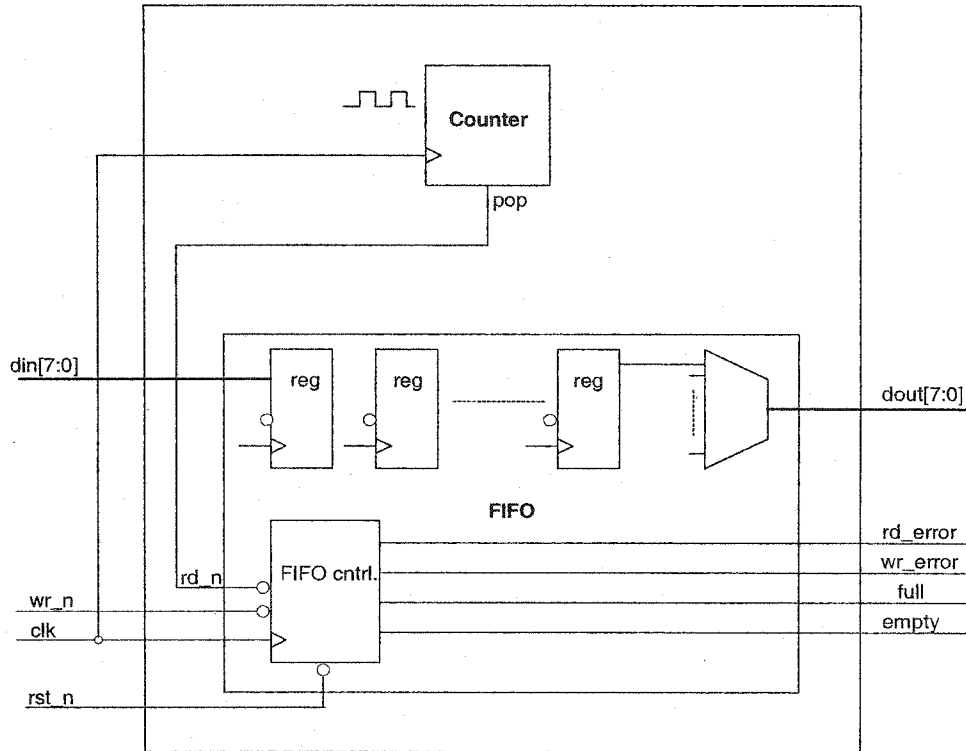


Figure 4.8: Leaky Bucket Datapath

4.3.6 Description of the Blocks and Pins

Implementation of Leaky Bucket algorithm into hardware has two major blocks or modules. First of all, it is made up of a First in First out (FIFO) buffer, which acts as the leaky bucket itself. Secondly, it has a counter that acts as the aperture or hole of the bucket. This counter sets a specific time interval (delay), which is analogous to the aperture size. Whatever be the packet rate of the input traffic, the packet gets out of the bucket at a constant rate that is decided by the hole-size. Likewise, the input packets are

buffered into the FIFO (bucket), and the output packets from the FIFO are sent at fixed intervals of time, which is the delay fed by the counter. It is assumed that all incoming packets are of uniform size. So data is sent out in the form of packets. If they were of different sizes, it would be better to send out data in terms of bits. Also, the aperture-size or counter-delay is set as parameter, and can be varied. Once the FIFO becomes full, the bucket discards all incoming packets. The design has flags that indicate when the FIFO becomes full or empty, and also flags that indicate whenever there are read or write errors. Various control signals are connected to the controller that enable reading and writing to the FIFO block. In this way, a turbulent flow of incoming traffic is converted to a more streamlined one, thus satisfying the whole phenomenon of traffic shaping. Following is the pin description of the design:

clk: Positive edge clock for all sequential logic.

din[7:0]: 8-bit input to the FIFO.

dout[7:0]: 8-bit FIFO output.

empty: Flag indicating that there is no valid data in the FIFO. There can be no further reads from the FIFO until after at least one write to the FIFO occurs.

full: Flag indicating that the FIFO is completely full. There can be no further writes into the FIFO until after at least one read of the FIFO occurs.

wr_n (low true): Synchronous (sync.) signal to write din into the FIFO. If FIFO is full, design must ignore this signal, and generate a wr_error output as described below.

wr_error (high true): Synchronous signal indicating that a write is trying to occur when the FIFO is full.

rd_n (low true): Synchronous signal to remove dout from the FIFO. If FIFO is empty, the design ignores this signal, and generates an rd_error output. This is not a request signal. Generally, this signal is also used to load the output of the FIFO into the next stage of the pipeline. This is controlled by the pop signal of the counter.

pop: Output of the counter that enables rd_n every seven clock pulses (can be varied).

rd_error (high true): Synchronous signal to indicate that a read occurred while FIFO was empty.

rst_n (low true): Asynchronous signal that initializes the FIFO to empty state.

The entire design of the traffic shaper supports both read and write operations simultaneously.

4.3.7 RTL Codes

```
//-----
```

```
// Name : Alfred Jude
```

```
// Class : EE299
```

```
// Thesis : FIFO module of Traffic Shaper
```

```
//-----
```

```
module fifo (
```

```
    clk,          //clk for sequential logic
```

```
    din,          //8 bit input to FIFO
```

```
    dout,         //8 bit FIFO output
```

```

empty,      //flag indicates no valid data in FIFO
full,      //flag indicates FIFO is full
wr_n,      //low true - write enable din to FIFO
wr_error,  //high true - write while FIFO full
rd_n,      //low true - remove dout from FIFO
rd_error,  //high true - read while FIFO empty
rst_n     //low true - async initialize FIFO to
          empty state

```

```
);
```

```
//-----
```

```
// Declaration , and assignment
```

```
//-----
```

```
parameter width = 8;           //FIFO bit width
```

```
parameter depth = 32;        //FIFO bit depth
```

```
parameter shift_in_, and_out = 2'b00, //state machine cases
```

```
    shift_in           = 2'b01,
```

```
    shift_out          = 2'b10,
```

```
    no_action          = 2'b11;
```

```

integer i; //for loop counter , and upper limit

reg [4:0] p; //array pointer max value is 5d'31

input clk, wr_n, rd_n, rst_n;

input [(width-1):0] din;

output [(width-1):0] dout;

output empty, full, wr_error, rd_error;

reg [1:0] currentstate;

reg [(width-1):0] fifo_data [0:(depth-1)]; //Array for data in FIFO

// instantiating the counter module for the pop signal

counter count(.clk(clk),.pop(rd_n),.rst_n(rst_n)); // Connection of ports

assign empty = (p == 0) ? 1'b1 : 1'b0;

assign full = (p == (depth-1)) ? 1'b1 : 1'b0;

assign rd_error = &{empty,!rd_n}; //rd_error true

assign wr_error = &{full,!wr_n,rd_n}; //wr_error true

```



```

assign dout = fifo_data[0];

//-----

// Procedural block

//-----

always @ (wr_n or rd_n)

    currentstate = {wr_n,rd_n};

always @ (posedge clk or negedge rst_n)
    begin

        if (rst_n == 1'b0) //async reset to fifo at the negative edge of rst_n
            //to initialize flags , and array pointer
            p<=0;
        else //sync FIFO operation
            begin
                case (currentstate)

                    shift_out: //remove dout from FIFO
                        begin
                            if (!rd_error) //no error condition

```

```

begin
    if (p != 1)
        begin
            for (i=1; i<depth; i=i+1)
                fifo_data [i-1] <= fifo_data[i]; //shift fifo data toward out
            end
            p <= p-1;          //Decrement array pointer
        end
    end

shift_in:    //write din into FIFO
begin
    if (!wr_error)    //no error condition
        begin
            fifo_data [p] <= din;          //shift data into FIFO array
            p <= p+1;          //increment array pointer
        end
    end

shift_in , and_out:
begin

```

```

        if (!rd_error)
            begin
                for (i=1; i<depth; i=i+1)
                    fifo_data [i-1] <= fifo_data [i];

                    fifo_data [p-1] <= din;    //p does not need to be updated
                                                // since data is shifted in/out of FIFO
            end
        end
    endcase
end
end

endmodule

//-----
// Name : Alfred Jude
// Class : EE299
// Thesis : Counter module of Traffic Shaper
//-----

//`timescale 1ns/1ns

module counter(clk,rst_n,pop);

```

```

parameter DELAY = 2;

input clk,rst_n;

output pop;

wire pop;

reg [DELAY:0]cnt;

assign pop = (cnt ==3'b111) ? 1'b0 : 1'b1;

initial
begin
    cnt = 3'b0;
end

always @(posedge clk or negedge rst_n)
begin
    if(!rst_n)
        cnt <= 3'b0;
    else
        cnt <= cnt + 1;

        $display("pop%b",pop);
end

endmodule

```

4.4 Other Design Blocks of IPP – An Overview

The input port processor consists of eight modules: serial to parallel conversion, packet encapsulation, routing table, FIFOs, free memory stack, memory, leaky bucket, scheduler, and parallel to serial conversion. These modules, shown in *Figure 4.9*, provide multicasting capabilities, traffic shaping, and scheduling. The previous sections have already gone through an in-depth design and description of the traffic shaper. The following section explains in brief, all other blocks that complement the entire IPP design.

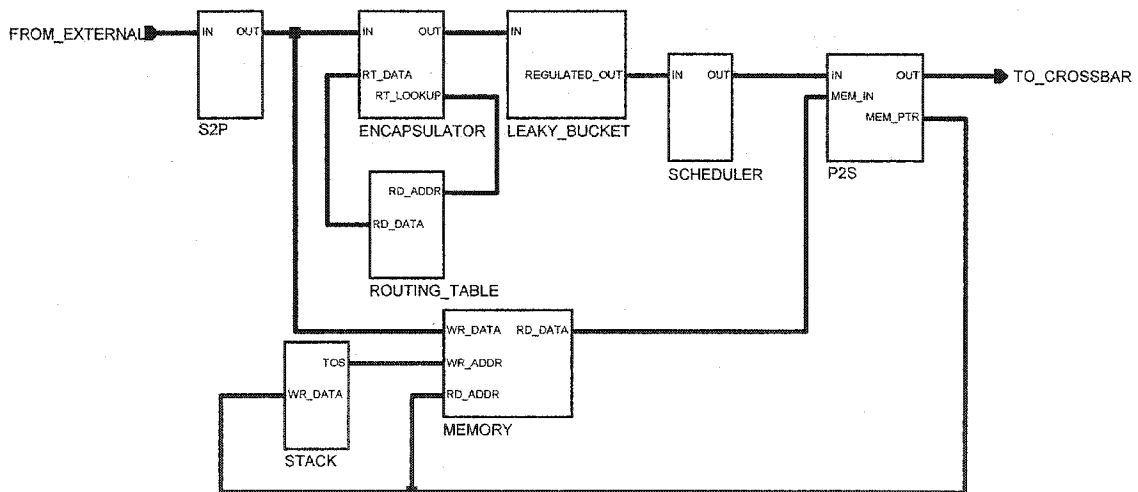


Figure 4.9: IPP System Block Diagram

4.4.1 Serial to Parallel Converter

The serial to parallel unit converts an incoming serial byte stream into a fully parallel data stream. This unit also performs communication handshaking with the external connection. Upon a pulse on the SYNC line, the unit begins to shift in bytes of data into its registers. When a second SYNC pulse arrives, the last byte of the packet arrives, and the data is ready to be passed to the encapsulator. A valid bit line is also available to delay the serial stream in mid-transmission.

4.4.2 Packet Encapsulator

This module encapsulates the incoming packet with an SSN header before forwarding further in the IPP. The format in the SSN header is shown in *Figure 4.10*. A routing table lookup (explained in *Section 4.4.3*) with the destination ethernet address, provides the SSN destination port. The Type-of-Service field in the IP header is copied onto the SSN header. The memory pointer field contains the pointer to the stack where the data corresponding to this header is stored.

SSN Destination	SSN Source	SSN Priority	IP TOS	Memory Pointer
4 bit	4 bit	4 bit	4 bit	7bit

Figure 4.10: SSN Header Format

4.4.3 Routing Table

When a packet enters the IPP, the destination port of the SSN should be chosen, based on the destination ethernet address of the incoming packet. This destination port needs to be appended to the incoming packet as part of the SSN header. The routing table is basically a lookup table containing the possible destination addresses and the corresponding SSN output port. An external algorithm (running in software; like the Address Resolution Protocol - ARP) fills this routing lookup table. The purpose of the routing table module is to look up an entry, corresponding to the destination ethernet address of the input ethernet packet, and provide the output SSN port. The packet encapsulation module populates the routing table module; performs the routing table lookups, and includes the SSN output port (i.e., the output of the routing table module) into the SSN header. An example of the routing table is shown in *Table 4.2*.

Destination Ethernet Address (48 bits)	Output SSN port (16 bits)
00:30:71:f8:00:ad	10
00:30:71:f8:01:74	12
....	
00:30:71:f9:01:37	9

Table 4.2: Example of the Routing Table

4.4.4 Routing Table Memory

The routing table memory module implements a 64 X 16 memory block used to store a table of ethernet destination address, and the corresponding output SSN port number. The ethernet destination address is 48 bits wide. The remaining 16 bits are used for the SSN port number (4 bits are used for the SSN port number, and the remaining 12 bits can be used for future expansion).

4.4.5 Routing Table Lookup

The routing table lookup module finds a matching entry for an input ethernet address with the contents of the routing table memory module. In every clock cycle, it reads the routing table memory, and checks if the ethernet address matches the address in the memory. If there is a match, the done flag is set high, and the output SSN port is made available at the output port (dout pin of the lookup table). If there is no match, it increments the rd_ptr, and continues to look up the next memory location.

4.4.6 First In First Out (FIFO)

FIFO is implemented using a cyclic buffer with read and write pointers. When data is written into the FIFO, data is actually written into the memory location pointed by the write pointer, and data is read from the FIFO pointed by the read pointer. A counter counts the number of memory locations that are currently occupied in the FIFO. At every positive edge of the clock signal, if the reset signal is low, then the FIFO is reset, i.e., the

FIFO memory is flushed out, and the `wr_ptr` and `rd_ptr` pointers are initialized to memory location zero; the counter is reset to zero; empty flag is set, and the full flag is reset. If the `wr_n` signal is low, data in `d_in` is written to the memory location pointed by `wr_ptr`. Then, the `wr_ptr` field is incremented by one, taking care of the wrap-around case (i.e., if `wr_ptr` exceeds 32, it is initialized back to 0). The counter field is also incremented by 1. If counter exceeds 32, the full flag and the `wr_failed` flag are set. If the `rd_en` signal is low, and if the counter value is zero, the `rd_failed` and empty signals are then set to high. Else, data in the memory location pointed to by `rd_ptr` is copied on to `dout`; the `rd_ptr` is incremented by 1 taking care of wrap-around case; the count is decremented by 1, and the full signal is reset.

4.4.7 Free Memory Stack

The stack is a first in, last out memory device (FILO). The stack module consists of two main parts: a state machine based controller and a memory device. The stack has three input operators: push, pop, and reset. On a push, data is written into the top of the stack, and a pop signal will remove the data at the top of stack. A reset signal will set the stack in a predetermined state, filling the registers with sequential numbers.

Design Considerations: The free memory stack is used as a memory management device. IPP memory management strategy takes advantage of the speed of FIFO queues and the robustness of memories such as RAM. To increase FIFO performance, queue sizes are fixed to reduce control logic. Since network packets can be of various lengths, a

memory device is needed to store packet payloads, while a fixed length header travels through the system. Since packets can leave the network in a different order from the one in which they arrived, a memory monitor is necessary to keep track of which locations in the memory are free for use. Borrowing a concept from operating systems principles, a free memory list serves as a memory manager implemented by a stack of pointers. Another optimization is achieved in the packet duplication process. By using a memory module for storage, copying is done cheaply and easily by appending a counter field to the memory locations, to signify the number of copies of that location needed.

4.4.8 Memory

The memory module is used to store packets and duplicate multicast packets by holding the memory, until all instances of the multicast packets exit the IPP. Writing to the memory takes two passes for a multicast packet and only one pass for a unicast packet. In order to keep track of how many copies a multicast packet needs, the packet counter in the memory module must be augmented after the multicast packet has been written to memory. The basic functionality of the module works just like an ordinary memory device, with a write enable signal.

4.4.9 Scheduler

The scheduler consists of the following components:

- Unicast and multicast input queues

- QoS classifier
- RTP and non-RTP unicast and multicast output queues.
- Weighted Fair Queuing (WFQ) to service the above queues

The unicast and multicast input queues are implemented using a FIFO buffer. The QoS classifier identifies the packet type as RTP or non-RTP, based on the Type-Of-Service (ToS) field of the SSN header. The RTP and non-RTP unicast and multicast output queues are implemented using a version of FIFO buffer that is modified to support WFQ. A simplified form of weighted fair queuing is implemented to service the queues.

QoS Classifier: The QoS classifier classifies the incoming packets into RTP and non-RTP, depending on the Type of Service (ToS) field of the SSN header. Since there are separate unicast and multicast input queues, there are four queues to be serviced by WFQ, namely unicast RTP, unicast non-RTP, multicast RTP, and multicast non-RTP.

Weighted Fair Queuing: The FIFO is modified to implement the WFQ algorithm. There are three additional ports in this FIFO namely, finishtime (32-bit wide), virtualtime (32-bit wide), and LW-ratio (8 bit wide).

Once the QoS classifier populates the queues, packets are scheduled using the Weighted Fair Queuing (WFQ). In practice, WFQ is implemented in hardware using the concept of finishtime and virtualtime. The first packet in each weighted fair queued

FIFO is assigned a finishtime that is a measure of when the packet should be serviced. Additionally, a global virtualtime (common to all queues) is maintained, which is equivalent to the finishtime of the last packet that was dequeued from the WFQ queues.

When a packet is inserted into a WFQ queue, and if the corresponding WFQ queue is empty, the finishtime of the packet is calculated as:

$$\textit{finishtime} = \textit{virtualtime} + \textit{length of packet} / \textit{weight of the queue}.$$

When a packet is dequeued from the WFQ queue, the finishtime of the next packet (which is now the first packet in the queue) is calculated as:

$$\textit{finishtime} = \textit{finishtime} + \textit{length of packet} / \textit{weight of the queue}.$$

Thus the IPP can be designed by the above mentioned hardware blocks in order to perform the three most important functionalities of any switch, namely multicasting, shaping, and scheduling. The following section deals with the analysis of this hardware, in order to come up with various parameter computations.

Chapter 5 Performance Analysis of Input Port

Processor (IPP)

This section could be written as the gist of the thesis. Thus far, the design part in terms of hardware of the switch fabric and input port processor was given. This section will explain the analytical part that helps in coming up with a performance evaluation. With this statistical analysis and evaluation, it is easy to estimate numerous parameters of the design including delay, waiting time, number of packets, total time of packets inside the IPP, load, total time, service time, traffic intensity, and also model the various queues including the multicast packets, shaped packets, scheduler real time and non-real time packets, and analyze the outputs. This would pave the way to estimate the best possible design for maximum performance. The preliminary block diagram of the IPP was already presented in the previous section. A copy of the same is given in *Figure 5.1*.

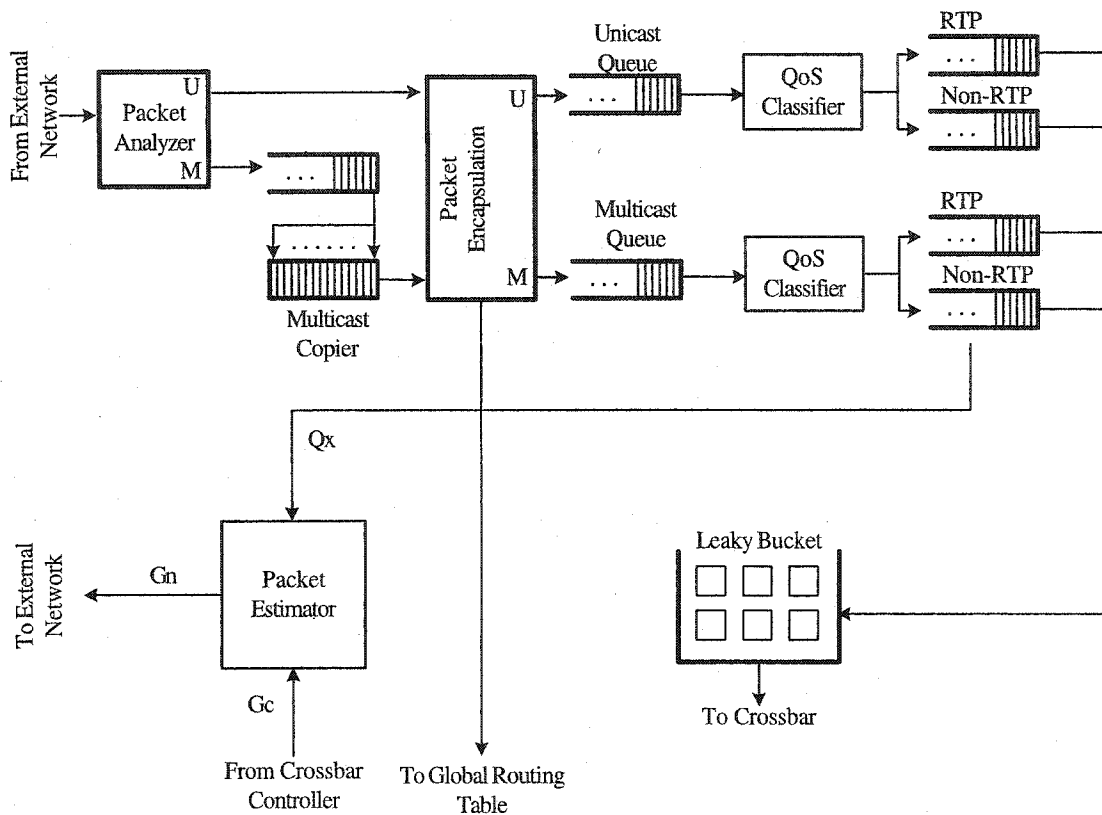


Figure 5.1: Preliminary IPP Design

The paper “*General Methodology for Designing Efficient Traffic Scheduling and Shaping Algorithms*” by Dimitrios Stiliadis and Anujan Varma [46], describes a general methodology for designing integrated shaping and scheduling algorithms for packet networks that provide fairness, low-end delay, and low burstness. According to it, the shaping mechanism is integrated with scheduling, and the resulting algorithms provide an identical end-to-end delay to that of Weighted Fair Queuing (WFQ). This kind of structure is proved to achieve a level of fairness in the distribution of free bandwidth, which is better than any other combination. It also provides optimal output burstness,

and is easily used in network adapters, routers, and switches that support traffic re-shaping. The logical structure for the shaper-scheduler structure in any design is shown in *Figure 5.2*.

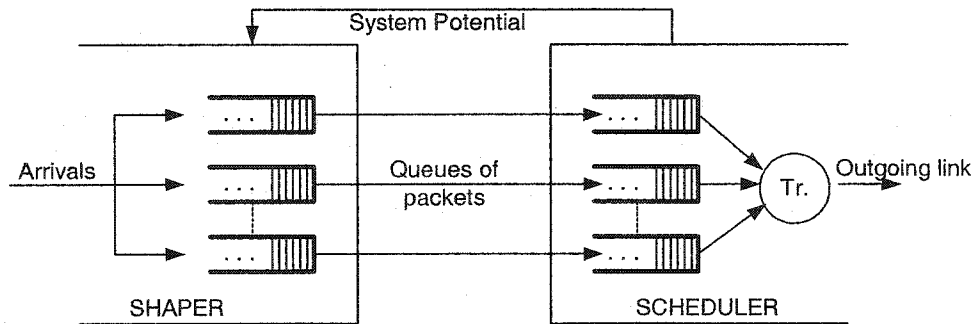


Figure 5.2: Shaper-Scheduler General Design

5.1 IPP Design Modification

By modifying the preliminary IPP design in accordance with the design shown in *Figure 5.2*, we obtain *Figure 5.3*. It should be noted that the shaper precedes the scheduler for reasons already mentioned.

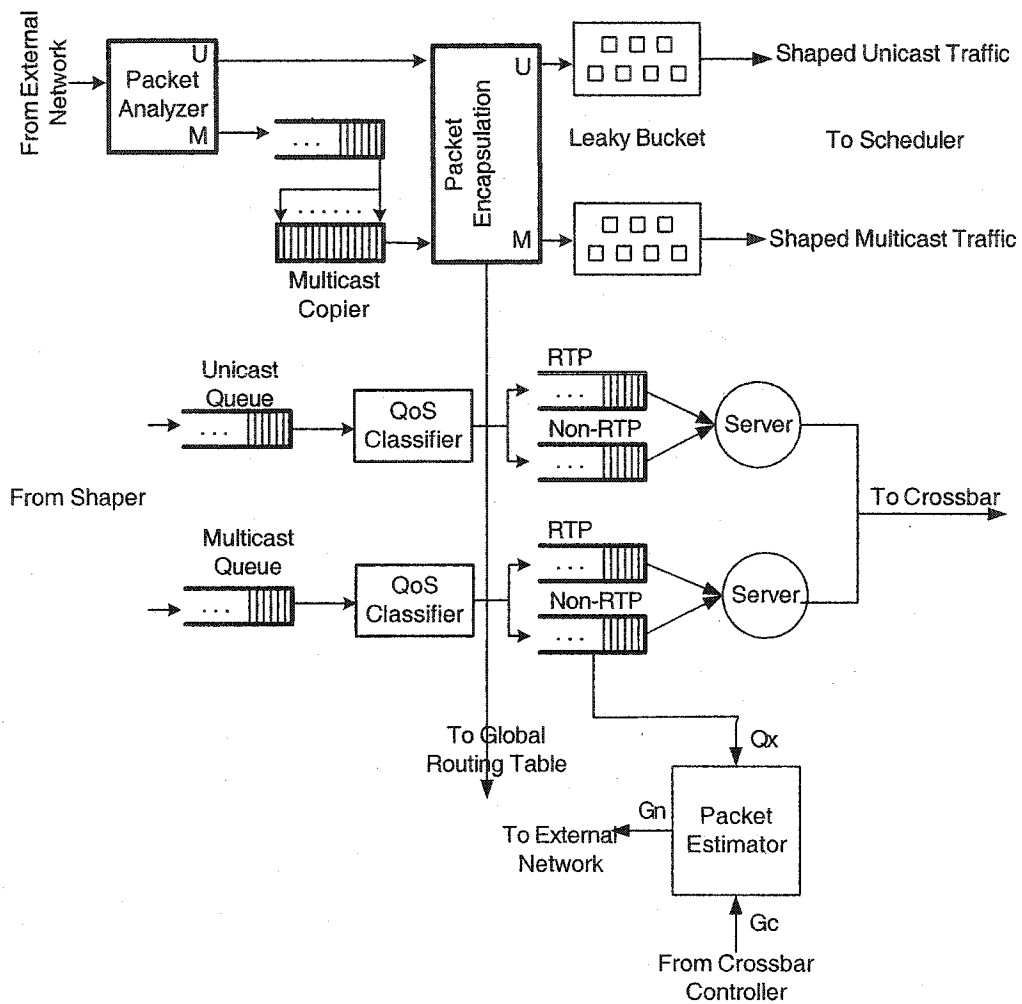


Figure 5.3: Modified IPP Design

5.2 Statistical Analysis of the IPP

Consider an $M/M/1$ queue where packets arrive according to a Poisson process of rate “ λ ”. To make a performance evaluation of any system, it is necessary to build a model that simulates the system’s functionality. In the case of IPP, a probabilistic (stochastic) model that would emulate the random nature of packets in a mathematical representation

is assumed. The various probabilistic distributions that vary according to the nature and number of arriving (input) packets are written below. Some of the obvious parameters to note are:

Mean = $1/\lambda$, for inter-arrival times

Mean = $1/\mu$, for service times that are independent

Number of packets is infinite, and μ (service rate) and λ (arrival rate) are exponential random variables.

5.2.1 System Models

The queuing models can be represented by a general notation $A / B / s (/w) (/p)$, where “A” denotes the arrival process that could be Markovian (M), General (G) or Deterministic (D); “B” denotes the departure process that could again be one of the above three; “s” denotes the number of servers; “w” the buffer positions or the capacity, and “p” denotes the population (number of packets) in the system.

Now that an idea about the various interfacing concepts and traffic modeling is obtained, the following sections will be an elucidation of the Spherical Switching Network (SSN), following more into the actual design, and then the necessary computations.

5.3 Modeling Real Time Packets (RTP)

Let us assume the number of packets = $N(t)$.

Since M/M/1 stands for continuous chain Markov process, the time until the next packet arrives is independent of service times of existing packets (memory less property). The real-time packets are bigger in size and are random in periodicity. This will require that these packets need infinite capacity buffers that makes any designer visualize them as large capacity single server queues. Inter-arrival time is independent of present and past history of $N(t)$.

To express $N(t)$, let us consider the various probabilistic ways:

(i) Probability of one arrival in an interval of length “ δ ”:

$$\begin{aligned} P [A(\delta)=1] &= \frac{\lambda\delta \exp(-\lambda\delta)}{1!} \text{ (Poisson)} \\ &= \lambda\delta \left\{ 1 - \lambda\delta + \frac{(\lambda\delta)^2}{2!} - \dots \right\} \\ &= \lambda\delta + o(\delta) \end{aligned}$$

(ii) Probability of more than one arrival

$$P [A(\delta) \geq 2] = o(\delta)$$

(iii) $P [\tau \leq \delta] = 1 - \exp(-\mu\delta) = \mu\delta + o(d)$

(iv) Probability of one arrival , and one departure in an interval of length “ δ ” is

$$P [A(\delta)=1, \tau \leq \delta] = P [A(\delta)=1] P [\tau \leq \delta] = o(\delta)$$

Hence the transition rate diagram for the real-time packet queues can be shown as:

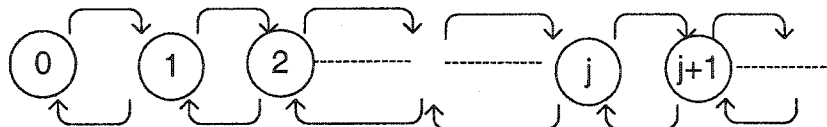


Figure 5.4: RTP Queues Transition Rate Diagram

It is stated that a steady state solution exists when $\rho < \lambda/\mu$

This condition can be met only for a stable system. Hence:

- Arrival rate = λ/μ = Maximum service rate
- Mean number of packets in the real-time packet queues:

$$E[N] = \sum_{j=0}^{\infty} j P[N(t)=j] = \frac{\rho}{1-\rho}, \text{ where } \rho \text{ stands for utilization.}$$

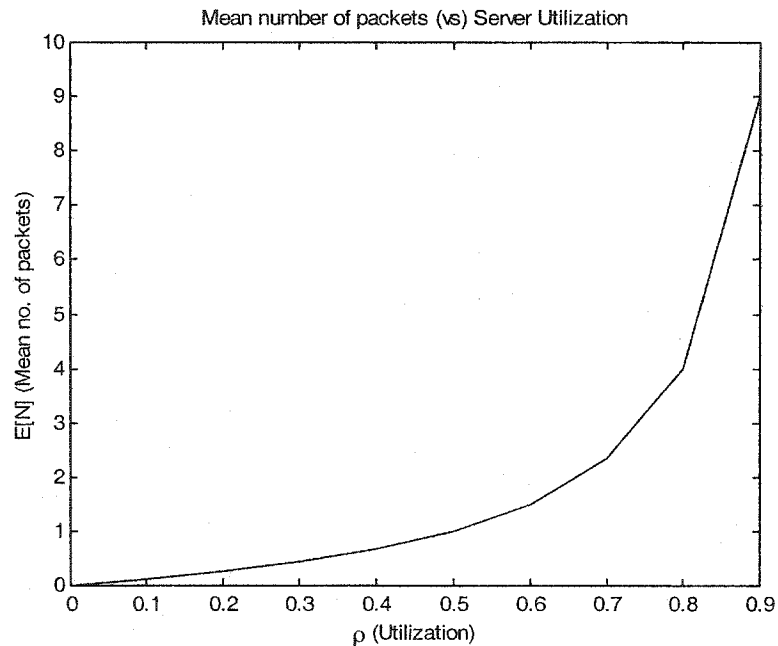


Figure 5.5: Mean Number of RTP Packets (vs.) Utilization

It can be inferred from the plot shown in Figure 5.5 that the mean number of packets increases exponentially with the server utilization.

5.3.1 Mean Total RTP Packet Delay

The mean total packet delay can be estimated as follows:

$$\begin{aligned} E[T] &= \frac{E[N]}{\lambda} = \frac{\rho}{\lambda(1-\rho)} \\ &= \frac{1/\mu}{1-\rho} = \frac{E[\tau]}{1-\rho} = \frac{1}{\mu-\lambda} \end{aligned}$$

where, μ – Mean service rate

λ – Mean arrival rate

$E[N]$ – No. of packets

➤ $\frac{E[T]}{1/\mu} = \frac{1}{1-\rho}$

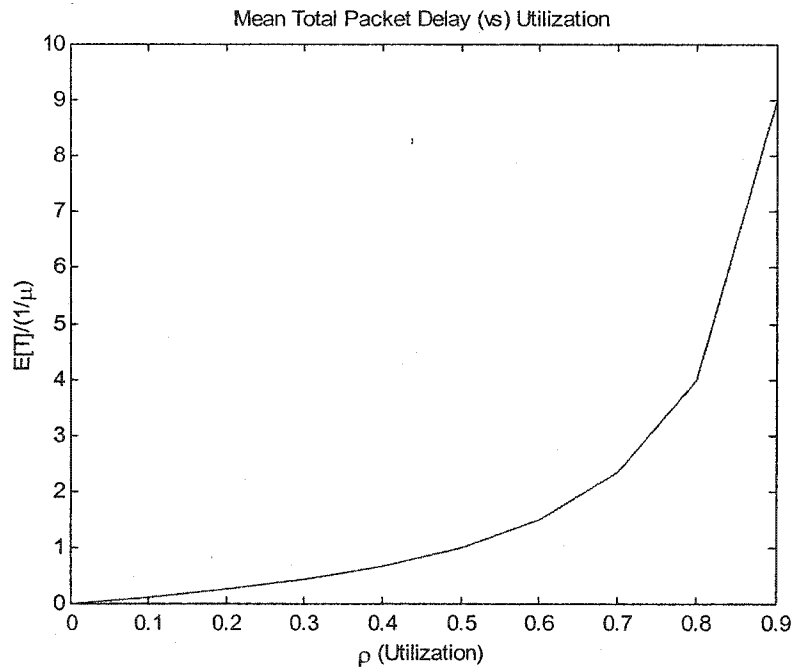


Figure 5.6: Mean Total RTP Packet Delay

It can be noted that as the number of packets increases, the mean total packet delay increases in accordance with the utilization.

$$\triangleright E[T] = \frac{1}{\mu - \lambda}$$

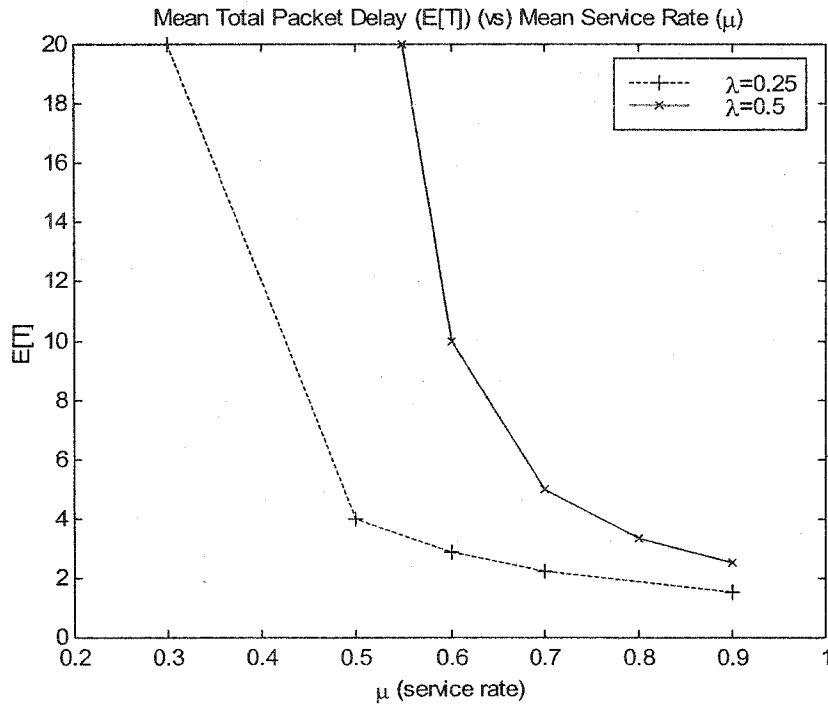


Figure 5.7: Mean Delay (vs.) Service Rate of RTP

The above plot tells that the mean total packet delay is decreased with rise in service rate.

Mean Waiting time in queue = Total time – Service time

$$\begin{aligned} E[W] &= E[T] - E[\tau] \\ &= \frac{E[\tau]}{1-\rho} - E[\tau] \\ &= E[\tau] \frac{\rho}{1-\rho} \end{aligned}$$

Mean number of packets in queue:

$$E[Nq] = \lambda E[w]$$

$$\begin{aligned}
E[N_q] &= \lambda E[w] \\
&= \frac{\rho^2}{1-\rho}
\end{aligned}$$

Server Utilization: $1-p_0 = 1-(1-\rho)$, where $\rho = \lambda/\mu$

Delay and arriving packets distribution:

Let us assume that N_a is the number of packets due to an arrival in the IPP

$P[N_a=k]$ – Arriving packets distribution

Statement: If packet arrival is Poisson, then the packet distribution is equal to the steady state distribution for the total number of packets in the IPP. Also, a packet arriving at time $t+\delta$ finds k in the Shaper-Scheduler if $N(t)=k$.

$$\begin{aligned}
\text{Hence, } P[N_a(t)=k] &= \lim_{\delta \rightarrow 0} P[(N(t)=k)/(A(t+\delta) - A(t)=1)] \\
&= \lim_{\delta \rightarrow 0} \frac{P[N(t)=k, A(t+\delta)-A(t)=1]}{P[A(t+\delta)-A(t)=1]} \\
&= \lim_{\delta \rightarrow 0} \frac{P[A(t+\delta)-A(t)=1/N(t)=k] P[N(t)=k]}{P[A(t+\delta)-A(t)=1]}
\end{aligned}$$

(Applying Conditional Probability)

Since probability of packet arrival at interval $(t, t+\delta)$ does not depend on $N(t)$,

$$\begin{aligned}
P[N_a(t)=k] &= \lim_{\delta \rightarrow 0} \frac{P[A(t+\delta)-A(t)=1] P[N(t)=k]}{P[A(t+\delta)-A(t)=1]} \\
&= P[N(t)=k]
\end{aligned}$$

Therefore, the arriving packets distribution is equal to the proportion of time when the Shaper-Scheduler queue has “ k ” packets.

For M/M/1 system of the design,

$$P[N_a=k] = P[N(t)=k] = (1-\rho)\rho^k$$

5.4 Total Time (T) of a Packet inside the Shaper-Scheduler

The previous section already explained the arriving packets distribution. It is vital to evaluate the total time spent by a packet inside the IPP, particularly at the shaper-scheduler combination. If FIFO algorithm for scheduling is employed, then:

Total time (T) = Residual service time of packet being served

= Service times of k-1 packets in queues

= Service time of incoming packet

$$f_T(x/N_a=k) = \frac{(\mu x)^k}{k!} \mu e^{-\mu x}, \quad x > 0$$

Probability density function = $f_T(x)$

$$= \sum_{k=0}^{\infty} \frac{(\mu x)^k}{k!} \mu e^{-\mu x} P[N(t)=k]$$

$$= \sum_{k=0}^{\infty} \frac{(\mu x)^k}{k!} \mu e^{-\mu x} (1-\rho) \rho^k$$

$$= (1-\rho) \mu e^{-\mu x} e^{-\mu \rho x}$$

$$= (\mu - \lambda) e^{-(\mu - \lambda)x}, \quad x > 0$$

$$\text{Mean} = \frac{1}{\mu - \lambda}$$

pdf for waiting time:

$$f_w(x) = (1-\rho) \delta(x) + \lambda(1-\rho) e^{-\mu(1-\rho)x} ; x > 0$$

5.5 Modeling Non-RTP Queues

Let us consider an input port processor with maximum capacity of “k”. The non real-time packets do not require a large capacity buffer, so it can be visualized as an M/M/1/k system.

$N(t)$ – Continuous time Markov chain process taking on values from $\{0, 1, \dots, k\}$

When $N(t) = k$, the system is full. Hence there is no more arrival.

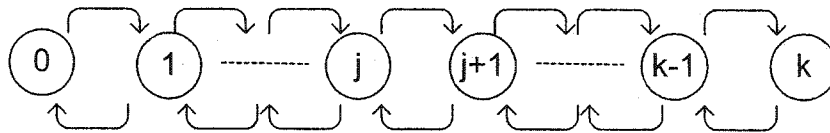


Figure 5.8: Transition Rate Diagram for Finite-Capacity Non-RTP System

Balance equations are $\lambda p_0 = \mu p_1$

$$(\lambda + \mu)p_j = \lambda p_{j-1} + \mu p_{j+1} \quad , \text{ where } j = 1, 2, \dots, (k-1)$$

$$\mu p_k = \lambda p_{k-1}$$

We already know that $\rho = \lambda/\mu$

$$\text{Steady State Probabilities: } P[N=j] = \frac{(1-\rho) \rho^j}{1-\rho^{k+1}} ; \quad j=0, 1, 2, \dots, k$$

We have three cases: $\rho < 1$, $\rho > 1$, $\rho = 1$

Mean number of non-RTP packets:

$$E[N] = \sum_{j=0}^k j P[N(t)=j]$$

$$= \frac{\rho}{1-\rho} - \frac{(k+1)\rho}{k+1} \quad ; \text{ for } \rho \neq 1$$

$$= k/2 \quad ; \text{ for } \rho = 1$$

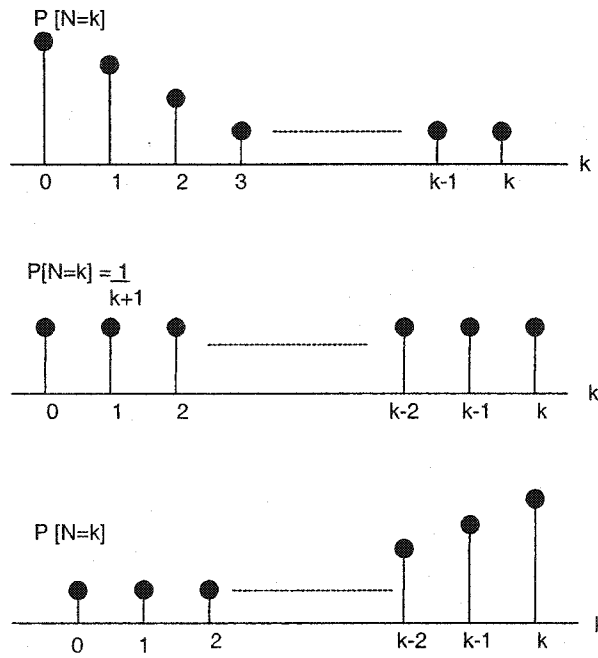


Figure 5.9: Probability Mass Function for $N(t)$ of Non-RTP Packets

(i) $\rho < 1$
 $P[N=k]$

(ii) $\rho = 1$
 $P[N=k] = \frac{1}{k+1}$

(iii) $\rho > 1$
 $P[N=k]$

Mean total time spent by non real-time packets in the system can be estimated by the proportion of time when the system turns away packets, which is $P[N(t)=k] = p_k$

Rate at which IPP discards packets is $\lambda_b = \lambda p_k$

Actual arrival rate into the system is $\lambda_a = \lambda (1-p_k)$

$$E[T] = \frac{E[N]}{\lambda_a} = \frac{E[N]}{\lambda(1-p_k)} \quad (\text{By Little's formula})$$

Since this is a finite capacity system, the difference between offered traffic load and actual load carried by IPP is given by these equations.

Offered Load = Traffic Intensity = λ (packets/sec) * $E[\tau]$ (secs of service/packet)

Carried Load = λ_a (packets/sec) * $E[\tau]$ (secs of service/packet)

5.5.1 Carried Load (vs.) Offered Load for Non-RTP Queues

Carried load is the actual requirement or demand met by the scheduler in terms of the number of packets. Offered load is the traffic intensity or the demand made to the system with the same metric. We know that $\rho = \lambda/\mu$ (Mean Arrival Rate/Mean Service Rate)

Here $\lambda_a =$ Carried load and $\rho =$ Utilization

Hence for a fixed number of packets “k”, the plot can be shown as in *Figure 5.10*.

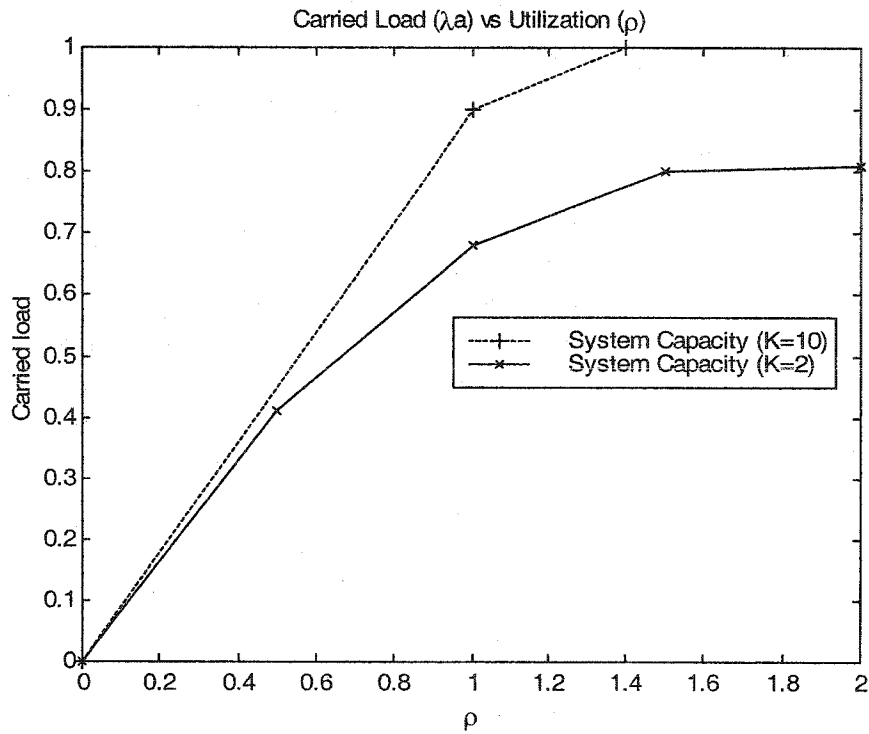


Figure 5.10: Carried Load (vs.) Utilization of Non-RTP

When system capacity increases, the capacity of the IPP increases, hence the increase in carried load implies that more packets enter the IPP.

5.5.2 Mean Packet Delay (vs.) Offered Load in Non-RTP Queues

Here again, when the number of packets (offered load) is increased from 2 to say 10, the mean delay increases.

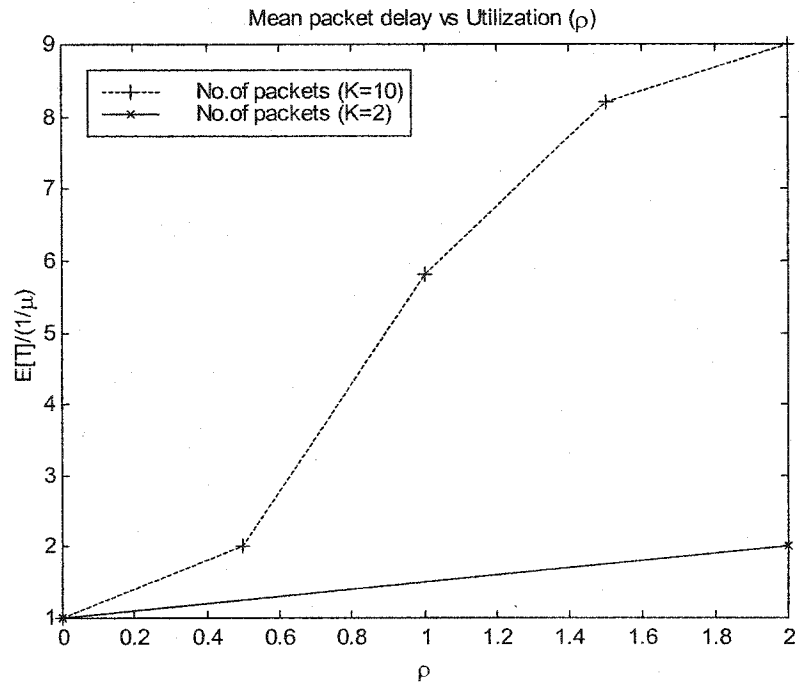


Figure 5.11: Mean Packet Delay (vs.) Utilization of Non-RTP

5.6 Modeling the IPP as a Multi-Server System (M/M/c, M/M/c/c)

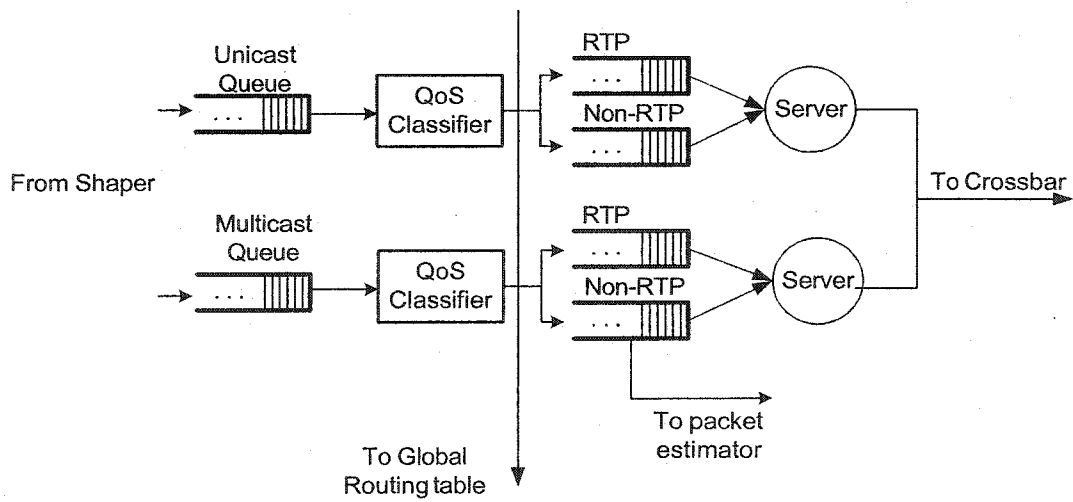


Figure 5.12: IPP as a Multi-Server System

In this design, the IPP is visualized as having two servers, both RTP and non-RTP for unicast as well as multicast packets. It consists of independent, identically distributed (iid) exponential random variables. The servers have different capacities, based on the nature of the incoming packet classes. The following section contains some important parameter estimations for this multi-server IPP design.

5.6.1 Important Estimations

(i) *Mean Number of Packets in Queue:*

$$\begin{aligned}
 E[N_q] &= \sum_{j=c}^{\infty} (j-c) \rho p_c = p_c \sum_{j=0}^{\infty} j' \rho \\
 &= \frac{\rho p_c}{(1-\rho)^2} \\
 &= \frac{\rho}{1-\rho} C(c,a)
 \end{aligned}$$

where, ρ – Servers' utilization

$C(c,a)$ – Probability that an arriving packet arriving finds all servers busy, and needs to wait in queue

By Erlang-C formula, $C(c,a) = \frac{p_c}{1-\rho} = P[W>0]$

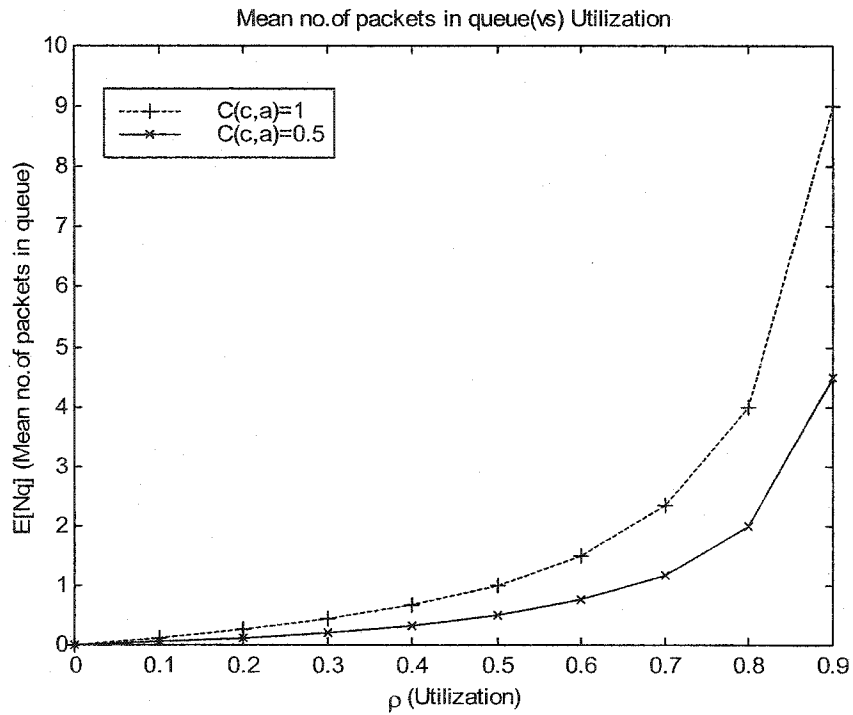


Figure 5.13: Mean Number of Packets (vs.) Utilization in a Multi-Server IPP

(ii) **Mean Waiting Time of Packets:**

This can be estimated by Little's formula. It is the ratio of the mean number of packets to their arrival rates.

$$\begin{aligned}
 E[W] &= \frac{E[N_q]}{\lambda} = \frac{\text{Mean no. of packets}}{\text{Mean arrival rate}} \\
 &= \frac{1/\mu \cdot C(c,a)}{c(1-\rho)}
 \end{aligned}$$

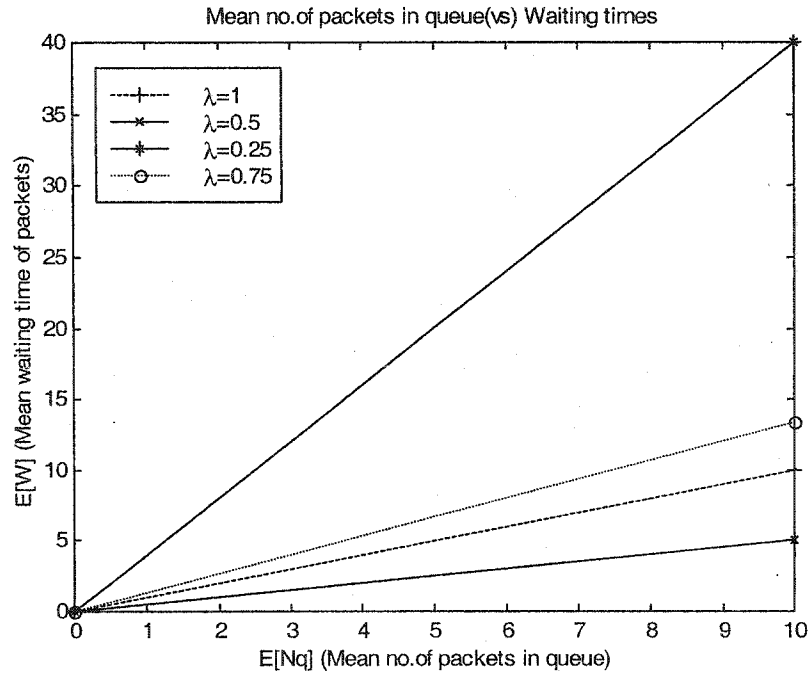


Figure 5.14: Mean Number of Packets (vs.) Waiting Times in a Multi-Server IPP

From these plots, it can be inferred that the mean waiting time of packets in the queue is directly proportional to the mean number of packets. This is also shown for varying arrival-rates of packets.

(iii) Mean Total Time in the System:

Mean total time in the system can be written as the sum of waiting time and the service time per packet.

$$\begin{aligned}
 E[T] &= E[W] + E[\tau] = E[W] + \frac{1}{\mu} \\
 &= \frac{E[N_q]}{\lambda} + \frac{1}{\mu} \\
 &= \frac{\mu E[W] + 1}{\mu} \quad (\mu - \text{Mean service rate})
 \end{aligned}$$

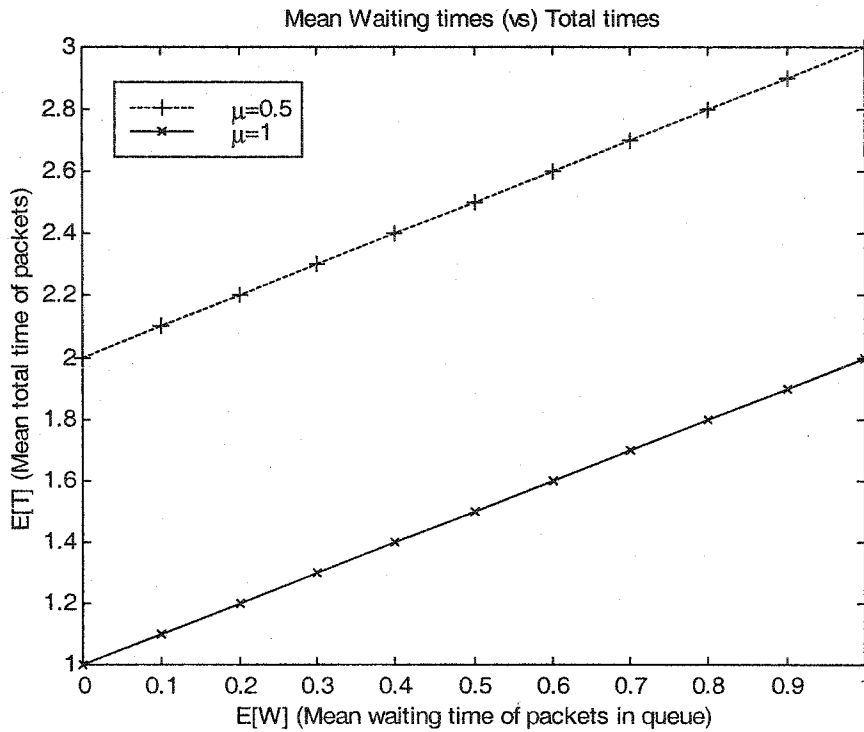


Figure 5.15: Mean Waiting Times (vs.) Total Times in a Multi-Server IPP

It can be noticed that the mean total time of packets in the IPP increases with the mean waiting time. The scale varies with the mean service rate as it is inversely proportional to mean waiting time.

(iv) Mean Number in the System:

This is evaluated by Little's formula. Mean number of packets in the system is directly proportional to the mean waiting time, the arrival rate being the constant of proportionality.

$$E[N] = \lambda E[T] = E[N_q] + a$$

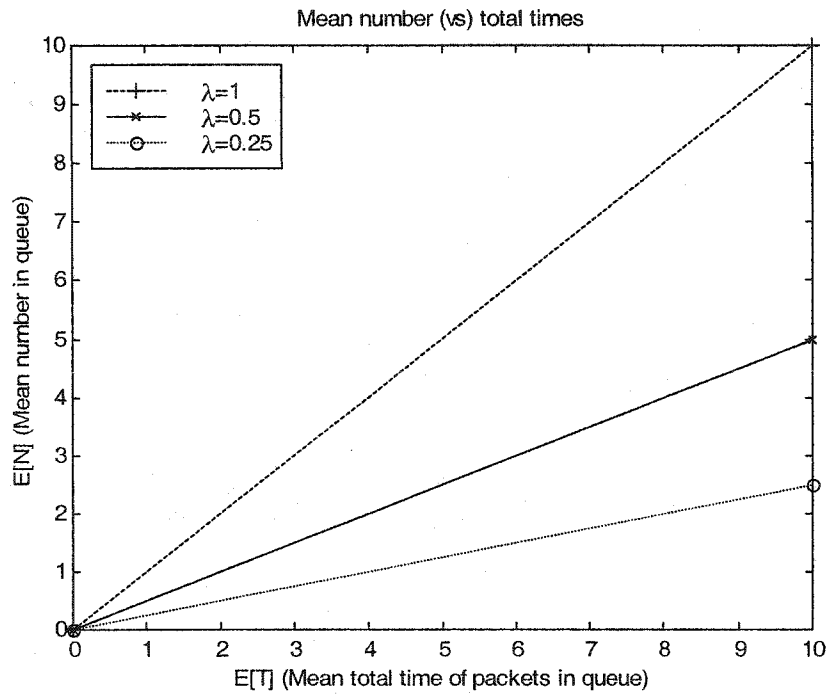


Figure 5.16: Mean Number (vs.) Total Times in a Multi-Server IPP

Mean number of packets increases with the mean total time.

5.7 Comparison of Scheduler with Different Server Powers

Let us assume the scheduler in two Queuing system forms, one with a single server, and other with “c” number of servers. Both of them have the same arrival rate and maximum processing rate. By modifying the scheduler into these two designs, the parameters like delay, waiting time, etc. can be calculated, and the better ones can be determined.

5.7.1 Scheduler with a Single Server

As already mentioned, the scheduler is now visualized as a single-server system by the block diagram shown in *Figure 5.17*.

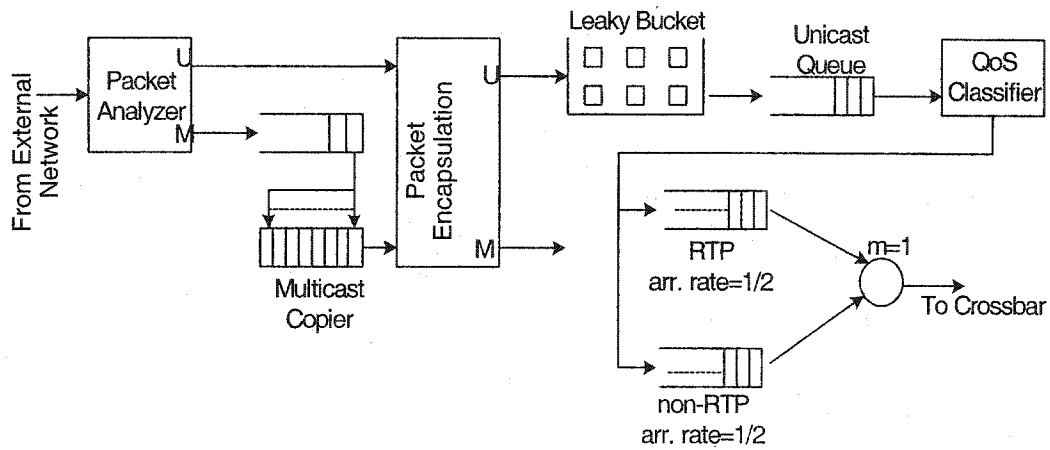


Figure 5.17: Scheduler with a Single Server

We know that $\rho = \frac{\lambda}{\mu} = \frac{1/2}{1} = 1/2$

Hence, the Mean waiting time is $E[W] = \frac{\rho}{\mu(1-\rho)} = 1\text{sec}$

Mean total delay is $E[T] = \frac{1}{\mu(1-\rho)} = 2\text{sec}$

5.7.2 Scheduler with Multi-Servers

Now the IPP scheduler is imagined to have two servers in order to compute the same parameters.

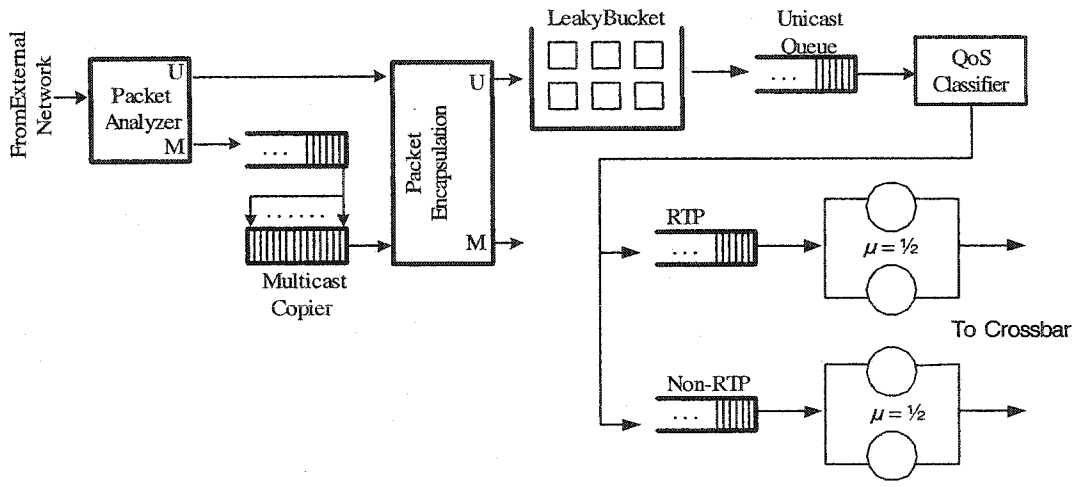


Figure 5.18: Scheduler with Multi-Servers

$$a = \lambda/\mu' = 1$$

$$\rho = \lambda/c\mu' = 1/c$$

Probability that arriving packets wait in queue due to busy servers (by Erlang-C formula)

can be written as:

$$C(c,a) = \frac{p_c}{1-\rho} = P[W>0]$$

$$\text{Here } C(c, 1) = \frac{a^2/c}{1-\rho}$$

Mean waiting time is:

$$E[W] = \frac{E[N_q]}{\lambda} = \frac{1/\mu}{c(1-\rho)} C(c,a)$$

$$= \frac{1/\mu'}{c(1-\rho)} C(c,a) = c/(c+1)'$$

$$\text{Mean Delay} = E[T] = 2/3 + 1/\mu' = 4c/3'$$

Therefore, the single server queuing system implies smaller total delay but larger waiting time, whereas multi-server system implies larger delay but smaller waiting time. Hence

by increasing the number of servers, the waiting time is decreased, but the total delay is increased.

5.8 Waiting Time Distribution of Multi-Server IPP

Let us assume that the IPP has “c” servers at the Shaper-Scheduler section. The density function of waiting time can be calculated by considering conditional probability that there are $(j-c) > 0$ packets in queue, given that all servers are busy $[N(t) \geq c]$.

$$\begin{aligned} P[N(t)=j/N(t) \geq c] &= \frac{P[N(t)=j, N(t) \geq c]}{P[N(t) \geq c]} \\ &= \frac{P[N(t)=j]}{P[N(t) \geq c]}, \quad j \geq c \\ &= \frac{\rho^j p_c}{\rho^c / (1-\rho)} = (1-\rho) \rho^{j-c} \end{aligned}$$

Thus when all servers are busy, M/M/c system becomes M/M/1 type.

Cumulative density function of W is derived as:

$$\begin{aligned} P[W \leq x] &= P[W=0] + F_w(x/W > 0) P[W > 0], \quad \text{where } x > 0 \\ &= (1 - C(c, a)) + (1 - \exp(-c\mu(1-\rho)x)) C(c, a) \\ &= 1 - C(c, a) \exp(-c\mu(1-\rho)x) \end{aligned}$$

5.8.1 Multi-Server Scheduler with Variable Capacity

This type has “c” servers and buffer capacity “c”. All packets that arrive when the queue is full, are discarded. Hence, when number of packets $N(t)=c$, arrival rate is zero.

Thus the Erlang-B formula can be derived.

$$B(c, a) = P[N=c] = p_c = \frac{a^c / c!}{1 + a + a^2/2! + \dots + a^c / c!}$$

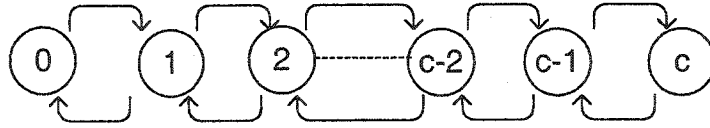


Figure 5.19: Variable Capacity Multi-Server Scheduler Transition Rate Diagram

Arrival rate of packets into the scheduler queue system is:

$$\lambda_a = \lambda (1 - B(c, a))$$

Average number in the system (Little's formula) is:

$$E[N] = \lambda_a E[\tau] = \frac{\lambda_a (1 - B(c, a))}{\mu}$$

where $E[N]$ = Carried load

λ = Arrival rate

$E[\tau]$ = $1/\mu$ - Mean service time

c = No. of servers

5.9 Finite Source Queuing System IPP Design

In this case, since there are two shapers, one for unicast and one for multicast queues, it is assumed that the entire shaper is a single-server queuing system that serves "k" sources, having two possible states for a source: (1) Source preparing a request for service from the server, and (2) Source's request being either in queue or being served.

The preliminary IPP design can be thus modified design shown in Figure 5.20.

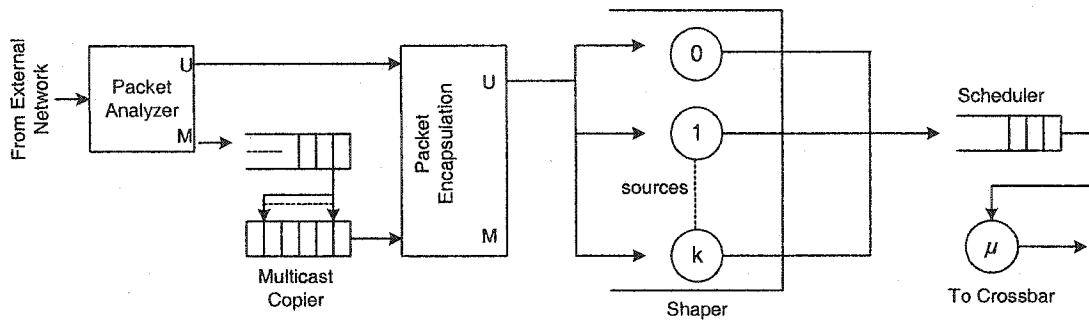


Figure 5.20: Finite Source Single Server Shaper-Scheduler

The sources that were previously mentioned can be “k” machines or computers (for example) or a server (a troubleshooter that fixes bugs).

Let $N(t)$ – Number of requests in the system. Each source spends an amount of time (exponential distribution) with mean $1/\alpha$ for each service request. The source generates a request for service in interval $(t, t+\delta)$ with probability $\alpha\delta + o(\delta)$.

If $N(t)=k$, then the number of idle sources = $K-k$.

Hence the rate at which service requests are generated = $(K-k)\alpha$

Thus $N(t)$ is a continuous time Markov chain with the transition time diagram shown in

Figure 5.21.

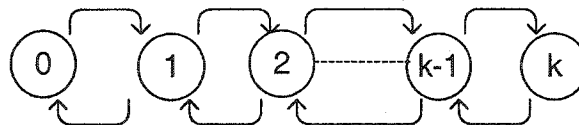


Figure 5.21: Markov Chain Transition Rate Diagram

Steady-state probability is written as:

$$p_k = \frac{K!}{(K-k)!} (\alpha/\mu)^k p_0; k=0,1,\dots,K$$

$$\text{where } p_0 = \sum_{k=0}^K \left\{ \frac{K!}{(K-k)!} (\alpha/\mu)^k \right\}^{-1}$$

It is important to compute “ λ ” (mean arrival rate) and mean delay $E[T]$. It is already known that server utilization “ ρ ” is the proportion of time when the system is busy:

$$\rho = 1 - p_0$$

Mean arrival rate is seen from:

$$\lambda E[T] = \rho = 1 - p_0$$

$$\lambda = \rho / E[\tau] = \mu \rho = \mu(1 - p_0)$$

Each source generates a request at the rate of $\{1/\alpha + E[T]\}^{-1}$ requests per second.

Actual arrival rate must be equal to the rate at which “ k ” sources generate requests,

$$\text{i.e., } \lambda = \frac{K}{1/\alpha + E[T]}$$

Mean delay in IPP for each request is $E[T] = K/\lambda - 1/\alpha$

Applying Little’s formula, mean number in IPP = $E[N] = \lambda E[T] = K - \lambda/\alpha$

Mean number of idle sources = λ/α

Mean waiting time = $E[T] - \text{Mean service time}$

$$E[W] = E[T] - 1/\mu$$

Time that a source waits for completion of service request, $P[\text{source busy}] = \frac{E[T]}{E[T] + 1/\alpha}$

5.9.1 Finite-Source Queuing System Example

Input port processor is assumed to have a scheduler with a server. There are different FIFO buffers (K number of queues) each one sending packets and waiting for the server to transmit them. "Throughput" of the server is the rate at which it computes transactions (schedules packets). "Repeat time" is the total time for a transaction to get over. The expressions for throughput and response time for two cases, K small and large, can be written as:

When K is small, the packets do not wait for a long time.

$$E[T] \cong 1/\mu$$

$$\lambda = \frac{K}{1/\alpha + 1/\mu} \quad \left[\lambda = \frac{K}{1/\alpha + E[T]} \right]$$

When K is large, $\lambda = \mu$ (μ - transactions per second)

$$E[T] = K/\mu - 1/\alpha$$

These two are asymptotic expressions. The point where the two E[T] asymptotes meet is called the system saturation point.

$$K^* = \frac{1/\mu + 1/\lambda}{1/\mu}$$

Delay and throughput for IPP as a finite source system as a function of number of sources, can thus be drawn.

5.9.2 Arriving Packets Distribution

There are FIFO service disciplines inside the shapers. This implies that $N_a=k$ requests in the system. A packet spends total time equal to the sum of one residual service time plus $(k-1)$ service times plus its own service time. These are exponential random variables.

$$\text{Mean} = 1/\mu$$

Mean time in the system for a request: $E[T/N_a=k] = \frac{k+1}{\mu}$

$$E[T] = 1/\mu \sum_{k=0}^{K-1} (k+1) P[N_a=k]$$

The only problem is that arrivals are not Poisson. $P[N_a=k]$ has to be found, for which the long-term proportion of time that arriving packets find “k” packets in the IPP.

Arrival rate when $N(t)=k = (K-k)\alpha$ requests per sec.

\therefore Number of arrivals that find “k” requests $\equiv (K-k)\alpha$ packets/sec * $p_k T$ sec in state “k”.

Total number of arrivals in time ‘T’ is the sum $\sum_{j=0}^k (K-j) \alpha p_j T$

Proportion of arrivals that find “k” requests in the IPP is given by:

$$P[N_a=k] = \frac{[(K-1)! / (K-k-1)!] (\alpha/\mu)^k}{\sum_{j=0}^{K-1} [(K-1)! / (K-j-1)!] (\alpha/\mu)^j} \quad 0 \leq k \leq K-1$$

This is the steady state probability of having “k” packets in the IPP with $K-1$ sources, i.e. a source when placing a request views a queuing system that behaves as if the source is absent.

5.10 Analysis of Traffic Shaper Output

Thus far, the IPP was assumed and analyzed with all the scheduler queuing systems; and computation and comparison of different parameters was performed. In this section, the traffic shaper output is considered. The IPP preliminary design can be shown as in *Figure 5.22*.

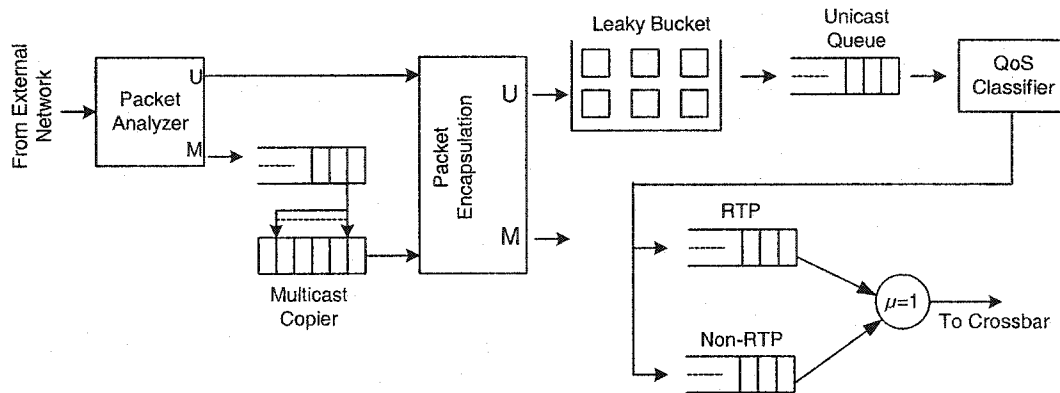


Figure 5.22: Shaper Output Analysis in IPP

The incoming packets to the traffic shaper are randomly spaced and paced. Hence it is a continuous-time random process. The state of the system is the data about the past behavior of packets that help predict the future behavior. The scheduler queuing systems had exponential distributions of packet inter-arrival and service times. So the number of packets $N(t)$ explained the state of the system. But here in the shaper, since the service times are constant, i.e., the size of the leaky bucket or the number of clock cycles for

every packet to get out of the system is made a constant, in a more hardware oriented language, the knowledge about when a packet started service specifies the packet's future departure time. Thus for the shaper output, the state of the system is explained by the number of packets along with the remaining time (residual) of the packet being shaped at a given time. By this elucidation, the output of the traffic shaper resembles an M/G/1 system nevertheless.

5.10.1 Residual Service Time

Assume a sequence of service times as shown in *Figure 5.23*.

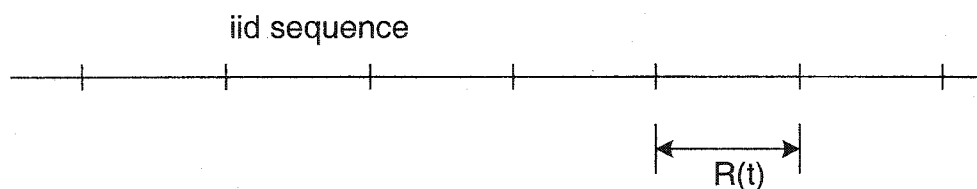


Figure 5.23: Sequence of Service Times

The arriving packets find the server busy. τ_1, τ_2, \dots iid sequence (independent, identically distributed). So when packets find the server busy, they pick a time at random in the time axis (scheduling). Residual service time is the remainder of time in the segment, which is intercepted.

After derivation, $E[R] = \frac{E[\tau]^2}{2E[\tau]}$

If one compares the residual service times of two systems, one with exponential service times, and the other with constant service times of mean “m”, the residual service time of constant random variable will be half that of the exponential one. Thus the shaper output has just half the residual service time of what the other systems in the IPP have, because it is not an exponential.

5.10.2 Mean Delay of Shaper Traffic

Let us assume a constant service time queuing system. The time “W” is spent by a packet waiting for service. Service discipline is FIFO.

$W = \text{Residual service time } R' + (k-1) \text{ service times of packets}$

$$E[W] = E[R'] + E[N_q(t)] E[\tau]$$

By Little’s formula, $E[N_q(t)] = \lambda E[W]$

$$\begin{aligned} E[W] &= E[R'] + \lambda E[\tau] E[W] \\ &= E[R'] + \lambda E[W] E[\tau] \\ &= E[R'] + \rho E[W] \text{ ----- (1)} \end{aligned}$$

where R' – Residual service time faced by an arriving packet, $R'=0$ (when IPP queues are empty)

R – Residual service time when a packet is in service.

$$E[R'] = \frac{\lambda E[\tau^2]}{2} \text{ ----- (2)}$$

Mean waiting time $E[W]$ of a packet = $E[W] = \frac{\lambda E[\tau^2]}{2(1-\rho)}$ Sub. (2) in (1)

But $E[\tau^2] = \sigma^2 + E[\tau]^2$

$$\triangleright E[W] = \frac{\rho (1+C_\tau^2) E[\tau]}{2(1-\rho)} \quad ; \quad \text{where } C_\tau^2 = \frac{\sigma^2}{E[\tau]^2}$$

Mean Delay $E[T]$ is given by:

$$\begin{aligned} E[T] &= E[\tau] + E[W] \\ &= E[\tau] + \frac{\rho (1+C_\tau^2) E[\tau]}{2(1-\rho)} \end{aligned}$$

5.11 Comparison of Mean Waiting Times of IPP Queues with Exponential and Constant Residual Service Times

IPP has been analyzed with exponential as well as constant residual service systems thus far. The only difference between these two systems, as is clear from the notations themselves, is that the scheduler system will have random length bursts (exponential), whereas the shaper system output has constant length bursts. Here the respective performances will be compared to come up with the better one to match the design requirements.

$$\text{Since } E[W] = \frac{\rho (1+C_\tau^2) E[\tau]}{2(1-\rho)}$$

$$E[W_{MM/1}] = \frac{\rho}{1-\rho} E[\tau] \quad ; \quad C_\tau \text{ (Coefficient of variation)} = 1$$

$$E[W_{MD/1}] = \frac{\rho}{2(1-\rho)} E[\tau] \quad ; \quad (C_\tau = 0)$$

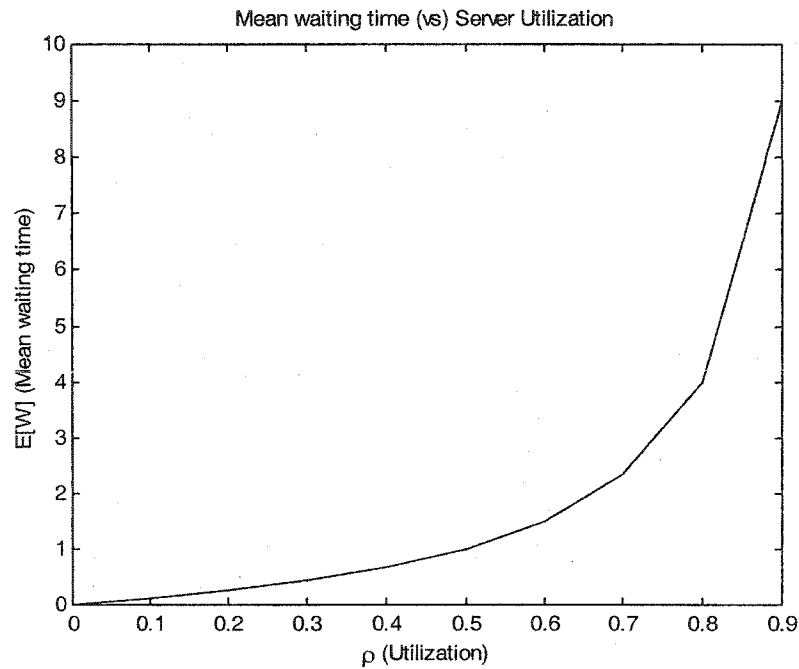


Figure 5.24: Plot of Mean Waiting Time (vs.) Server Utilization

$$E[W_1] = E[R^{**}] + E[N_{q1}] E[\tau_1]$$

$$\text{Mean waiting time for type 1 packets: } E[W_1] = \frac{E[R^{**}]}{1-\rho_1}$$

Similarly, since an arriving type-2 packet finds $N_{q1}(t)=k_1$ to be type-1, and $N_{q2}(t)=k_2$ to be type-2 packets waiting in queue, W_2 is given by:

$$E[W_2] = E[R^{**}] + E[N_{q1}] E[\tau_1] + E[N_{q2}] E[\tau_2] + E[M_1] E[\tau_1]$$

= Residual service time of packets + k_1 service times + k_2 service times + higher priority type 1 packets' service times

$$\text{Since } E[N_{q1}] = \lambda_1 E[W_1], E[N_{q2}] = \lambda_2 E[W_2], E[M_1] = \lambda_1 E[W_2],$$

$$E[W_2] = E[R''] + \rho_1 E[W_1] + \rho_2 E[W_2] + \rho_1 E[W_2]$$

$$\text{Solving, } E[W_2] = \frac{E[R'']b}{(1-\rho_1)(1-\rho_1-\rho_2)}$$

$$\text{Hence, } E[W_k] = \frac{E[R'']}{(1-\rho_1-\dots-\rho_{k-1})(1-\rho_1-\dots-\rho_k)}$$

$$E[R''] = \lambda \frac{E[\tau^2]}{2}$$

where $\lambda = \lambda_1 + \lambda_2 + \lambda_3 + \dots + \lambda_k$ (arrival rate)

$$E[\tau^2] = \frac{\lambda_1}{\lambda} E[\tau_1^2] + \dots + \frac{\lambda_k}{\lambda} E[\tau_k^2]$$

Mean waiting time for “k” type packets can hence be written as:

$$E[W_k] = \sum_{j=1}^K \frac{\lambda_j E[\tau_j^2]}{2 (1-\rho_1-\dots-\rho_{k-1}) (1-\rho_1-\dots-\rho_k)}$$

5.12 Priority Service: Mean Delay Calculations

The shaper-scheduler section that has been analyzed thus far, had all queues assigned with the same chance. This section has various queues having different priority levels and termed as high to medium to low priority queues. Priority scheduling is one of the most widely implemented scheduling algorithms in any design. The primary structure of priority service is shown in *Figure 5.25*. In this design, there are nearly four types of packets, which are unicast real time packets, unicast non-real time packets, multicast real time packets, and multicast non-real time packets.

There are “k” priority classes of packets. “k” type packets arrive according to Poisson process of rate λ_k , pdf $f_{\tau k}(x)$, and mean $E[\tau_k]$. Each priority class has a separate queue,

and whenever the server becomes free, it selects or transmits a packet from the highest priority queue. This is called “head-of-line priority service”.

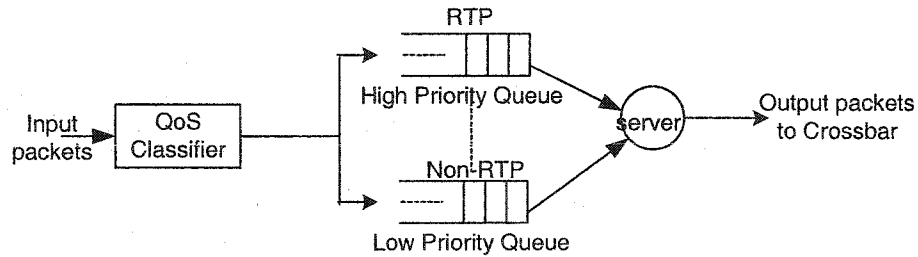


Figure 5.25: Priority Scheduling

Server utilization from “k” type packets is $\rho_k = \lambda_k E[\tau_k]$

Total server utilization < 1 is $\rho = \rho_1 + \dots + \rho_k < 1$. Assume mean waiting time W_1 (of highest priority packet).

$W_1 =$ Sum of residual service time R of packet found in service and $N_{q1}(t)=k$, the service times of type-1 packets.

Mean delay for type “k” packets is $E[T_k] = E[W_k] + E[\tau_k]$

5.12.1 Practical Methods to Compute Mean Waiting Time

This section has the computation of mean waiting time by assuming two systems of IPP design:

- (1) Ordinary queuing system
- (2) Two-priority queuing systems (priority to type-1 packets)

Arrival rates of two classes are Poisson (same rate).

Type-1 packets: Service time = 1ms.

Type-2 packets: Exponentially distributed amount of time with mean 20 ms.

First two moments:

$$E[\tau] = \frac{1}{2} E[\tau_1] + \frac{1}{2} E[\tau_2] = \frac{1}{2} + 20/2 = 10.5$$

$$E[\tau^2] = \frac{1}{2} E[\tau_1^2] + \frac{1}{2} E[\tau_2^2] = \frac{1}{2} (1^2 + 2(20^2)) = 400.5$$

Traffic intensity for each class:

$$\rho_1 = 1\lambda/2 ; \rho_2 = 20\lambda/2$$

$$\text{Total: } \rho = \lambda E[\tau] = 10.5\lambda \quad ; \quad \lambda - \text{total arrival rate}$$

$$E[R] = \frac{\lambda E[\tau^2]}{2} = 200.25\lambda$$

$$\text{Mean waiting time for IPP (as M/G/1 system), } E[W] = \frac{E[R]}{1-\rho} = \frac{200.25\lambda}{1-10.5\lambda}$$

$$\text{For the priority system, we have } E[W_1] = \frac{E[R]}{1-\rho_1} = \frac{200.25\lambda}{1-0.5\lambda}$$

$$E[W_2] = \frac{E[R]}{(1-\rho_1)(1-\rho)} = \frac{200.25\lambda}{(1-0.5\lambda)(1-10.5\lambda)}$$

We see that the waiting time of type 1 packets is improved by a factor of $(1-\rho)/(1-\rho_1)$, and type-2 gets worse by $1/(1-\rho_1)$.

$$\text{Mean waiting time, } E[W_p] = \frac{1}{2} E[W_1] + \frac{1}{2} E[W_2] = \frac{1}{2} \left(\frac{E[R]}{(1-\rho_1)} \right) \left(1 + \frac{1}{(1-\rho)} \right)$$

$$= \frac{1-2.75\lambda}{1-0.5\lambda} E[W]$$

where $E[W]$ – mean waiting time without priorities.

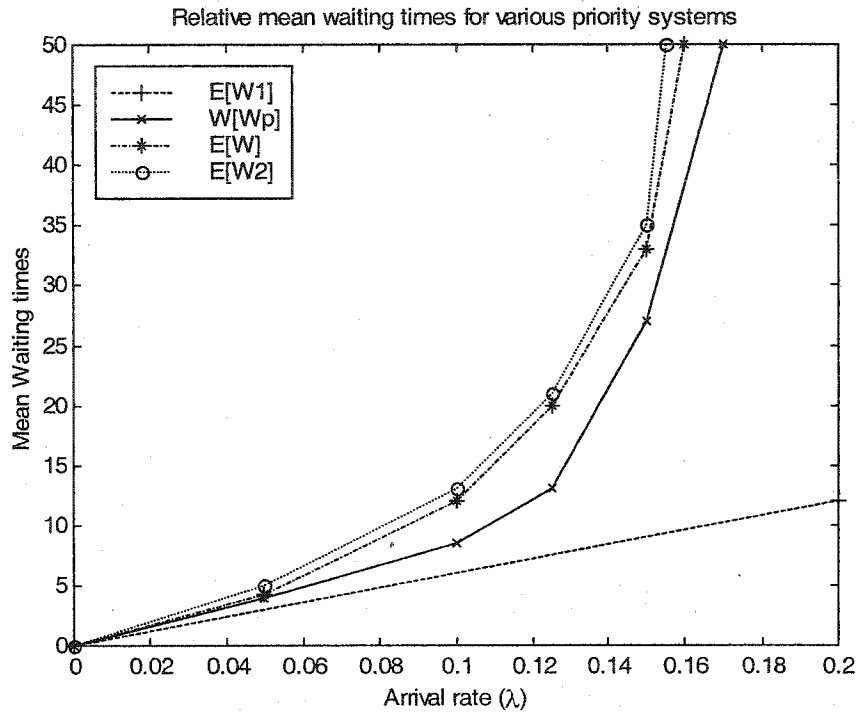


Figure 5.26: Mean Waiting Times for Priority Systems

5.12.2 Shaper with Embedded Markov Chains

The shaper output is a constant residual service time queuing system. It was found that the state of this system is decided by the number of packets in the IPP ($N[t]$), and the residual service time of packet in service. When residual time=0, the number of packets left behind by the j th departing packet, $N_j=N(D_j)$; where D_j is the instant when j th service completion occurs. The steady state pmf of $N(t)$ must be found by calculating for chain N_j first. In this section, the shaper data will be analyzed as embedded Markov chains, in order to compute parameters like delay, number of packets, etc.

5.12.3 Embedded Markov Chain

Sequence $N_j = N(D_j)$ is Markov chain. If $N_{j-1} \geq 1$, packet enters the shaper (for service immediately at time D_j).

$N_j = N_{j-1} - 1 + M_j$ (1 – packet served in between, and M_j – number of packets that arrive during service time of j th packet)

Packet (j-1) leaving the shaper non-empty at time D_{j-1} . If $N_{j-1} = 0$, no packet leaves until j th packet arrives, and completes service. Number of packets that enter the queue at this time, $N_j = M_j$, if $N_{j-1} = 0$. Packet (j-1) leaving the shaper empty at this time D_{j-1}

Two postulates:

- (1) Rate at which packet arrivals find “n” in the system equals the rate at which departures leave “n” in the system.
- (2) Distribution of states seen by arriving packets is the same as the steady state distribution.

5.13 Number of Packets in the Shaper Buffers

The gist of the derivation to determine the number of packets in an M/G/1 system is as follows:

Generating function for the steady state pmf of N_j . For that, the transition probabilities for N_j ,

$$p_{ik} = P[N_j=k/N_{j-1}=i] = P[M_j=k-i+1] \quad ; \quad i>0$$

$$p_{0k} = P[N_j=k/N_{j-1}=0] = P[M_j=k]$$

Probability that there are $N_j=k$ packets in the system at time j is

$$P[N_j=k] = P[N_{j-1}=0] P[M_j=k] + \sum_{i=1}^{\infty} P[N_{j-1}=i] P[M_j=k+1-i]$$

$$P[N_d=k] = P[N_d=0] P[M=k] + \sum_{i=1}^{\infty} P[N_d=i] P[M=k+1-i]$$

N_d – Number of packets left behind by a departing packet.

Generating functions for N and M :

$$G_N(z) = \sum_{k=0}^{\infty} P[N=k] z^k$$

$$G_M(z) = \sum_{k=0}^{\infty} P[M=k] z^k$$

After some manipulations, $G_N(z) = P[N=0] G_M(z) + (1/z) \sum_{i=1}^{\infty} P[N=i] z^i$

$$= \sum_{k=0}^{\infty} P[M=k+1-i] z^{k+1-i}$$

$$G_N(z) = \frac{(1-E[M]) (z-1)G_M(z)}{z-G_M(z)}$$

Similarly, $G_M(z)$, the generating function for the number of arrivals during a service time

$$G_M(z) = \tau'(\lambda(1-z)); \quad \tau'(s) - \text{Laplace transform of pdf of } \tau$$

$$\tau'(s) = \int_0^{\infty} e^{-st} f_{\tau}(t) dt$$

$$G_N(z) = \frac{(1-\rho)(z-1)\tau'(\lambda(1-z))}{z - \tau'(\lambda(1-z))} \quad \rho - \text{Server utilization}$$

This Pollaczek – Khinchin formula can be used to find for $N(t)$ of an M/M/1 system.

Laplace transform for pdf of an exponential service of mean $(1/\mu) = \tau'(s) = \frac{\mu}{s+\mu}$

$$G_N(z) = \frac{(1-\rho)(z-1)[\mu/(\lambda(1-z)+\mu)]}{z - [\mu/(\lambda(1-z)+\mu)]}$$

$$= \frac{1-\rho}{1-\rho z}; \quad \rho = \lambda/\mu$$

$$G_N(z) = \sum_{k=0}^{\infty} (1-\rho) \rho^k z^k$$

$$= \sum_{k=0}^{\infty} P[N=k] z^k$$

steady state pmf = $P[N=k] = (1-\rho) \rho^k$; $k=0,1,2,\dots$

pmf for number of packets can also be found in a similar way.

5.14 Delay and Waiting Time Distributions in Shaper FIFOs

The number of packets was computed in the previous section. The delay and waiting time distributions is calculated by assuming the FIFO queuing system inside the shaper.

Packet spends T_j seconds in queue.

No. of packets N_d left behind in queue = No. of packets that arrive during these T secs.

$$\begin{aligned} \text{Generating function for } N_d: G_{Nd}(z) &= \sum_{k=0}^{\infty} \int_0^{\infty} P[N_d=k/T=t] f_T(t) dt z^k \\ &= T'(\lambda(1-z)) ; T\text{-Total delay in the IPP} \\ &\quad T'(s) \text{ - Laplace transform of pdf } T \\ T'(\lambda(1-z)) &= \frac{(1-\rho)(z-1) \tau'(\lambda(1-z))}{z - \tau'(\lambda(1-z))} \end{aligned}$$

$$\text{If } s = \lambda(1-z), \text{ then } T'(s) = \frac{(1-\rho)s \tau'(s)}{s - \lambda + \lambda \tau'(s)} ;$$

where $T = W + \tau$, W and τ being independent random variables

$$\therefore T'(s) = W(s) \tau'(s)$$

$$\text{Solving using Laplace transforms, } W'(s) = \frac{(1-\rho)s}{s - \lambda + \lambda \tau'(s)}$$

The above Pollaczek-Khinchin transform equations can be utilized to calculate pdfs of “ W ” and “ T ” for an $M/M/1$ system.

5.15 Departure from Shaper-Scheduler Systems

When a packet needs service from many servers, a network of queues is required. Queues follow one after the other. In the case of IPP design, the same set of packets traverses through numerous queues at each stage of the hardware. First of all, the input queues to the multicasting unit, and secondly the two different queues as the output are present, the types being unicast queue as opposed to multicast queues. The second stage has another set of queues related to the shaper. There is an input queue, a set of queues

inside the shaper itself that act as FIFO buffers and the output queue from the shaper. The next set of queuing systems is seen at the scheduler section, the real time ones and the non-real time ones. Thus it can be noted that the entire IPP design has innumerable queues, and the set of traversing packets are being served at different times. This is termed as the 'Network of Queues'. The statistical properties of departure process from a queue are found.

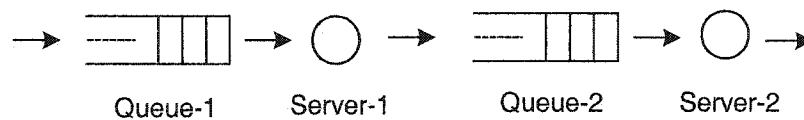


Figure 5.27: Network of Queues

Network of queues is shown in *Figure 5.27*. Departure from one queue becomes arrivals to the next. Arrivals to first queue are Poisson with rate " λ ". Service time at queue-1 is exponentially distributed with rate being $\mu_1 > \lambda$.

Service time in queue-2 is also exponentially distributed, with rate being $\mu_2 > \lambda$. It can be derived that the number of packets at queue-1, and those in queue-2 at the same time instant are independent random variables. Also, the steady state pmf at second queue is M/M/1 system with Poisson arrival rate " λ " and exponential service time " μ_2 ". The network of queues has a product-form solution.

5.15.1 Application of Burke's Theorem on IPP

Let us consider all queuing systems of IPP, at steady state with arrival rate λ , then:

- Departure process is Poisson with rate " λ ".
- At each time " t ", number of packets in system $N(t)$ is independent of sequence of departure times prior to t .

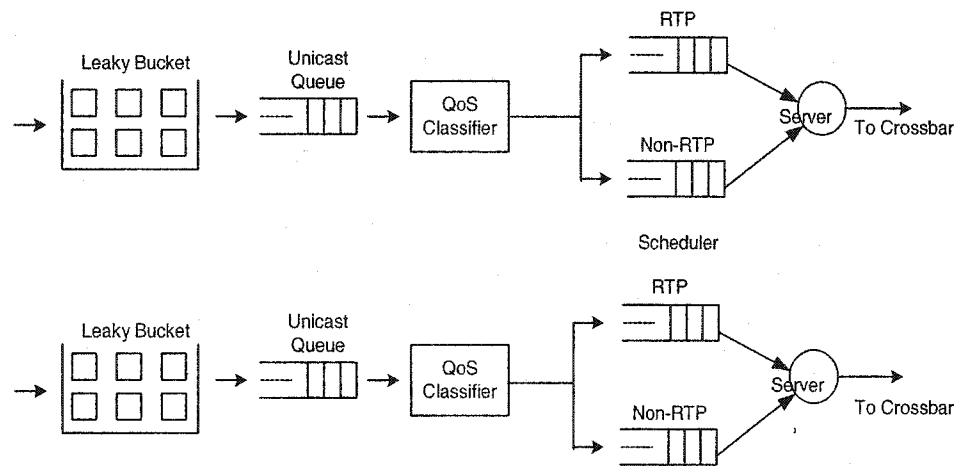


Figure 5.28: IPP according to Burke's theorem

The various types of packet queues, including the shaper output, real time and non-real time packets were already explained. It could be deciphered that the number of packets of different queues at the same time instant is an independent random variable. For instance, the real time packets need infinite capacity buffers because of their colossal nature of packets, whereas the non-real time packets could be well buffered with a finite number of servers of known finite capacity. As opposed to these two types being exponential and unpredictable, the output packets from the shaper can be expected, and

are constant queues. Thus the entire IPP design has a tandem combination of a number of queues.

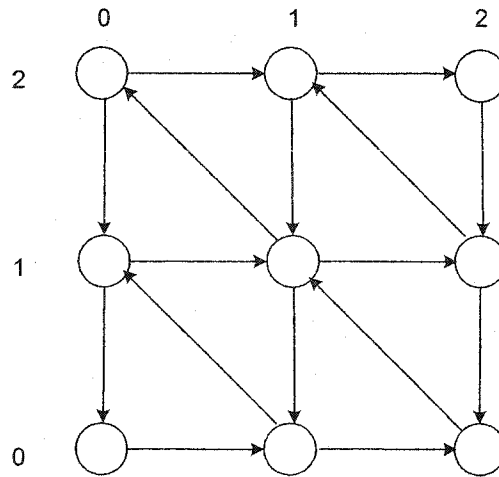


Figure 5.29: A Simple Example of Scheduler's Servers (Transition Rates)

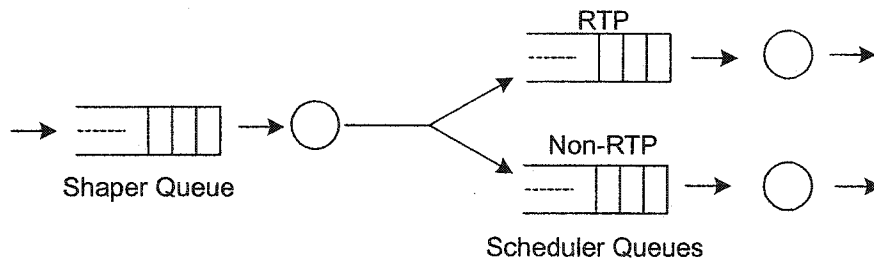


Figure 5.30: Feed-Forward Network of Queues

When one writes about the scheduler and shaper queues only, they can be written as a feed-forward network of queues, wherein a packet does not visit any queue more than once. There is no feedback or closed loop for these two blocks.

5.15.2 Network of Queues

This can also be explained as an extension of the product form solution for the steady state pmf to a broader class of queuing networks, because Burke's theorem allows a packet not to visit a single queue more than once. So Jackson's theorem has a feedback queue, where queues will not be Poisson. In a network of queues, there are feedback concepts, where the same packets traverse the same queue more than once.

5.15.3 Closed Queue Networks

Fixed number of packets becomes recursive at times to form a closed network of queues. External arrival rates are zero, and there is always a fixed number of packets. Steady state pmf is the product form, but queue states are no longer independent. *Figure 5.31* will explain the meaning of closed queues.

5.15.4 Closed Queuing Network in IPP

As mentioned before, a closed queuing network is one in which the same set of packets traverse through the same set of queues for various operations. Some of the operations are feedback paths; which consist of closed loops to add delay, and buffers for storage and making copies. In IPP design, there is just one queue that forms a closed loop. It is the multicast unit that needs to make copies of the same packets in order to send them to a different set of destinations. A simple closed queue network that has a number of queues and servers is shown in *Figure 5.31*.

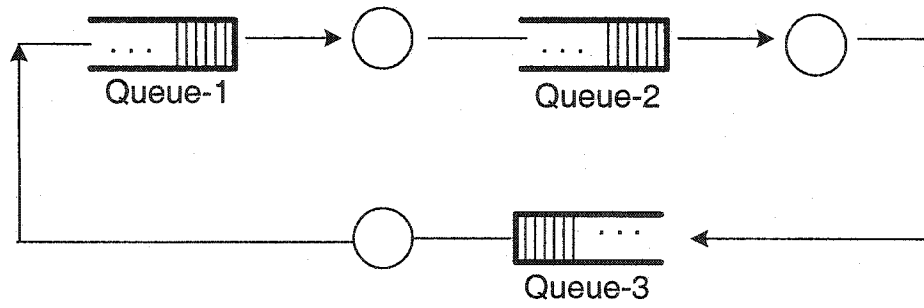


Figure 5.31: Closed Network Model of IPP

The concept of feedback and closed loops can be seen in the multicasting unit. Multicasting is the process of copying the same packet to send them to a different set of destinations. A packet comes out of the copier with two copies, and then these two packets are fed back to the copier queues, so that there are four copies of the same packet, and this process is repeated until the required number of the same packet is obtained. Later these multicast packets are sent to the leaky bucket, and they go on into the design. The multicast copier as a closed network model can be visualized with Figure 5.32.

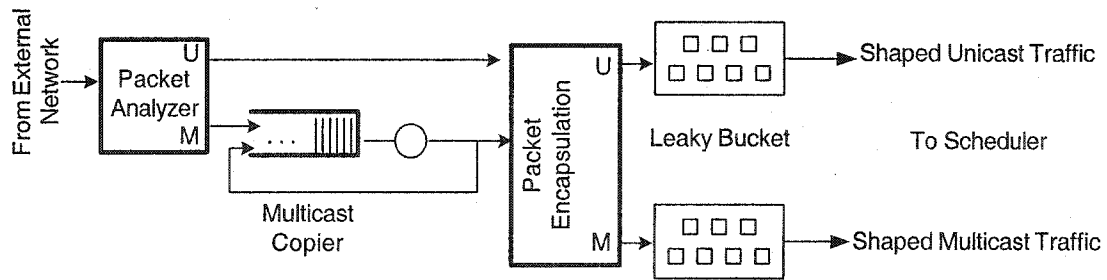


Figure 5.32: IPP as a Closed Queue

Thus it can be seen that the entire Input Port Processor (IPP) design could be subjected to an in-depth statistical analysis to estimate the numerous parameters involved. Parameters like the mean number of packets, mean total packet delay, delay distribution, and arriving packets distribution have been calculated, by modeling the real time packets of the scheduler. The behavior of the non-RTP packets that are the scheduler's output are also modeled, in order to plot the carried load, the mean packet delay, etc. The total time taken by a packet inside the shaper-scheduler blocks is then calculated. This is followed by the visualization of IPP as a multi-server system, one for unicast and the other for multicast packets, each serving real as well as non-real time packets. Estimation of various parameters including mean delay; mean waiting time, total number of packets, etc. for this multi-server system was performed. The scheduler was designed to have a single server and multi-server combinations, and a comparison to show that multi-server systems are faster was carried out.

The second part was the analysis of the traffic shaper, where it was said that the outgoing shaped packets traverse at a constant predictable rate, mentioned the residual service time to estimate the arriving packets distribution, and analyzed the nature of shaper traffic with and without priority service that shows enhanced functionality. Plots were drawn, which signified the relative mean waiting times for priority and non-priority systems for various types of packets. The shaper output was further scoped to estimate the waiting time distributions in the shaper buffers, and the delay and mean waiting time for this new system were derived. The concluding part had the Burke's and Jackson's

theorem applications that explained the network of queues concept and the closed queuing network model. The IPP design parts were again modeled and tested to this feed-forward network just to show the enhanced flexibility of this design. Thus, by considering the various design blocks inside the IPP, and by deriving the parameters associated with the different queuing systems, a complete performance evaluation of the entire system has been obtained.

Chapter 6 Conclusion

The development of the internet to its acme, the essentiality of networking protocols, the incorporation of multifunctional systems, etc. make scientists come up with seminal innovative ideas with respect to every single field of networking including protocol development using software, device design using hardware, and implementing both with various tools and Operating Systems (OS). In this thesis, coming up with some creative work in terms of the switch network and device with respect to the hardware design, and analyzing the same, was considered. In this section, the methodology in coming up with that creative work will be discussed.

First of all, an introduction for any reader into this oceanic field of networking was given. The applications and the need of this sector of technology that has been driving the world lately were defined. The pros and cons of various types of networks, be it wired or wireless, followed by the classification of network protocols, networking software and systems, the communication link types, and various network hardware devices were discussed. Since this thesis was scoped for hardware development, emphasis was given to the latest (current) technologies in explaining almost all innovative inventions being applied in every day's life lately. Thus, it would be sufficient for anybody to understand the motive behind this entire work.

Secondly, an evaluation of network interfacing and explanation of various technologies that needs to be embedded in any switching system was given. The whole idea of designing and evaluating a switching system was further enumerated by discussing traditional switch fabric designs, conventional bottlenecks, and advantages faced by various industry players. This was completed with a delineation of a generic switching system, and an explanation of what exactly happens inside the fabric.

Thirdly, a summary of the Spherical Switching Network (SSN), a technical publication, was put forth. It was described by first defining some basic switching concepts, and then the entire functionality of the system. It has the 16 X 16 network, with each switch element consisting of port processors and a crossbar. Later, the performance evaluation curves that suggested the superiority of this system to all its peers were detailed. This section concluded with a brief elucidation of why this particular system was chosen for this thesis.

Fourthly, the hardware modules of the Input Port Processor (IPP) of the switch element were designed. An implementation of the Leaky-Bucket algorithm for traffic shaping, with the RTL design and simulations was completed. By shaping input traffic, various edges including congestion control and effective bandwidth utilization were obtained. This was preceded by multicasting, and followed by scheduling, which again was done by Weighted Fair Queuing (WFQ). A number of blocks including FIFO buffers, serial to parallel converters and vice-versa, memory, stack, routing tables,

encapsulations, etc. were designed, and all of these with datapaths, RTL codes and timing diagrams made up the required design.

Fifthly, a performance evaluation of the complete IPP design was made. This was done by building a stochastic model, and visualizing the IPP as various queuing systems. Since each of these queues is based on differences in each parameter, it was carried by calculating all of them, and plotting the values that would help in deciding how each queue operates in the design. The real time and non-real time packets in the scheduler, the non-exponential constant output traffic from the shaper, and the multicast queues from the copier were modeled in a probabilistic way, and the plots were drawn using Matlab. Since each of the queues are different in nature and behavior, estimation of the important system numbers like delay, waiting time, number of packets, etc. was performed. This section gives an insight of how exactly the IPP functions and exists. The analytical part is creative because, since the proposal of the SSN, one would see what happens in the IPP in a mathematical vision.

Finally, some recommendations for the betterment of the whole system performance will be mentioned in a separate section called future-work that is likely to follow. This would provide additive continuity to the venture, and paves the way for the improvement of system performance. In that way, this design would become compatible for various cutting-edge prevalent technologies, and help in obtaining the industry specifications.

The thesis thereby concludes with what can be mentioned as a culminating work in the advanced field of ASIC/VLSI design embedded with networking technologies.

Chapter 7 Future Work

Colossal documentation with regard to this research and development has already been delineated. A lot more can be added to the entire system, both in terms of design as well as analysis. They can be explained as follow:

With respect to the Input Port Processor (IPP) design, only Internet Protocol (IP) packets have been considered here. This could well be extended to Asynchronous Transfer Mode (ATM) packets as well. Further enhancement of the implementation of various networking protocols including MPLS (Multi Protocol Label Switching), ICMP (Internet Control Message Protocol), ARP (Address Resolution Protocol), etc. is possible. The entire processor could be assumed to have bigger shapers that could assume bigger packets that are more busy in nature. Scheduling can be done for more types of data in addition to real and non-real time packets. Different algorithms than leaky bucket and WFQ can be utilized for the above purposes. Multicasting can be done at a broader level by assuming more copies of packets to more destinations. The complete SSN system can be extended to more planes that work in parallel.

Figure 7.1 is the parallel architecture of the crossbar mesh. This architecture has sixteen lines, representing sixteen ports of SSN switch fabric. These sixteen ports are broken down to nine parallel planes for faster information using de-multiplexers. If it is assumed that 64 K information has arrived, this control bit is sent to the control plane of

SSN Crossbar. The rest of the information being the actual data itself, is sent to the remaining eight planes.

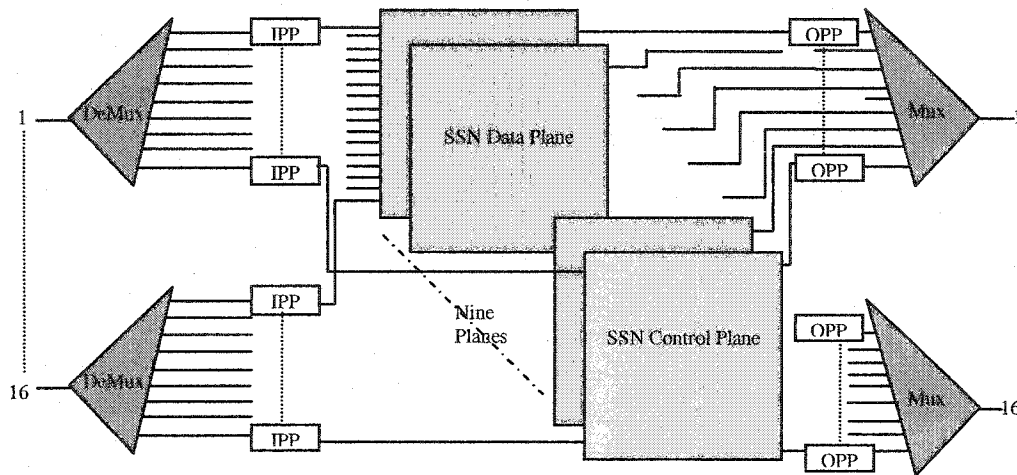


Figure 7.1: Parallel Architecture of Crossbar Mesh

Each and every line would have its own IPP and OPP. The routing for all the nine planes would just be the same and hence there is only one controller for the crossbar. The deflections will also be in the same manner except that the first plane has control bits. So it needs to employ 1:9 de-multiplexers and 9:1 multiplexers to process the data. This system would easily increment the speed of operation with terms of packet transfer, and enhance the performance as well. It was also mentioned in the analysis, that increase in the number of servers will enhance the performance of the IPP with respect to waiting time, delay, and the overall speed of the network. Since this will make the design more

complex, the increase in the number of planes could be a great scope for the project expansion.

With respect to the analytical part, a mathematical model was developed to derive each parameter and come up with the plots. To make it simpler, it could be well modeled or simulated using a tool like OPNET that has all built-in functions to make a network or any communication system. The parameter estimation in that case will not involve too many computations, because the tool has all functions embedded to it, thus calculating as well as plotting the parameters would be less complex. To be more technical, a traffic analyzer could be built using software, and be used to control the behavior of the system. Also, a real time network environment could be created in the lab and by plugging this analyzer into the virtual network, the feel of the application in a working environment could be witnessed. Also, congestion and flow controls can be implemented to the IPP for making the output even better.

All these areas mentioned above, would involve a lot of research, development and analysis. On the perspective of future engineers, these could evolve to be very innovative and challenging. Thus it can be noted that expanding the horizons of the IPP and SSN is not too complicated and it has a lot more scope to it.

REFERENCES

- [1] Mir, N. (2002). An Efficient Switching Fabric for Next-Generation large-scale Computer Networking. *IEEE Journal of Computer Networks*, vol. 6, no. 4, April, 1200-1212.
- [2] Chao, H. (1992). Architecture design for regulating and scheduling user's traffic in ATM networks. *Conference Proceedings on Communications Architecture and protocols, Baltimore, MD, USA, August, 77-87*.
- [3] *Agere Systems Advance Product Brief. (1997)*. ATM Switch Element (ASX) LUC4AS01. Retrieved May 10, 2002, from http://www.agere.com/metro_regional_transport/docs/PN96065.pdf
- [4] Tannenbaum, A. (1996). *Computer Networks. Prentice Hall*.
- [5] Peterson, L. (1999). *Computer Networks: A Systems Approach. Morgan-Kaufmann*.
- [6] Nutt, G. (2001). *Operating Systems: A Modern Perspective. Addison-Wesley*.
- [7] Leon-Garcia, A. (1994). *Probability and Random Processes for Electrical Engineering. Addison Wesley*.
- [8] Giacomelli, J., Hickey, J., Marcus, W., Sincoskie, W., Littlewood, M. (1991). Sunshine: A high-performance self-routing broadband packet switch architecture. *IEEE Journal of Selected Areas in Communications*, vol. 9, no.8, December, 1289-1298.
- [9] Suzuki, H., Nagano, H., Suzuki, T., Takeuchi, T., Iwasaki, S. (1989). Output-buffer switch architecture for asynchronous transfer mode. *Proceedings of the International Switching Symposium, March, 99-103*.
- [10] McDysan, D., Spohn, D. (1999). *ATM Theory and Applications. McGraw-Hill Series of Computer Communications*.
- [11] Kawahara, K. (1995). Performance analysis of backpressure congestion control: Preliminary case. *Proceedings of IEEE GLOBECOM'95, November, 304-309*.
- [12] Gallager, D. (1991). *Data Networks. Prentice Hall*.
- [13] Turner, J. (1988). Design of a broadcast packet switching network. *IEEE Transactions on Communications*, vol. 36, no. 6, June, 734-743.

- [14] Yeh, Y., Hluchyj, M., Acampora, A. (1987). The Knockout Switch: A simple, modular architecture for high performance packet switching. *IEEE Journal of Selected Areas in Communications*, vol. 5, no. 8, October, 1274-1283.
- [15] Ahmadi, H., Denzel, W. (1989). A survey of modern high-performance switching techniques. *IEEE Journal of Selected Areas in Communications*, vol. 7, no. 7, September, 1091-1103.
- [16] Floyd, S., Jacobson, V. (1993). Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, vol.1 no.4, August, 397-413.
- [17] Semeria, C. (2002). T-Series Routing Platforms: System and Packet Forwarding Architecture. *Juniper Networks White Paper*. Retrieved November 10, 2002, from http://www.telecomweb.com/reports/Tseries_RoutingPlatform_WP.pdf
- [18] Bianchi, G., Turner, J., (1993). Improved Queueing Analysis of Shared Buffer Switching Networks. *IEEE/ACM Transactions on Networking*, vol.1, no.4, August, 482-490.
- [19] Demers, A., Keshav, S., Shenker, S. (1990). Analysis and Simulation of a Fair Queueing Algorithm. *ACM SIGCOMM'89*, 3-12.
- [20] Golestani, S. (1994). A Self-Clocked Fair Queueing Scheme for Broadband Applications. *IEEE INFOCOM'94*, April, 636-646.
- [21] Bennett, J., Zhang, H. (1996). WF2Q: Worst-case Fair Weighted Fair Queueing. *IEEE INFOCOM'96*, March, 120-128.
- [22] Keshav, S. (1991). On the Efficient Implementation of Fair Queueing. *Internetworking: Research and Experiences*, vol. 2, 157-173.
- [23] Francini, A., Chiussi, F. (2001). A Weighted Fair Queueing Scheduler With Decoupled Bandwidth and Delay Guarantees for the Support of Voice Traffic. *IEEE Globecom'01*, November.
- [24] Dash, T. (2000). Network Primer - Traffic Regulators. Education Development Center Inc. Retrieved May 20, 2002, from <http://www2.edc.org/cope/networkprimer/primch5.pdf>
- [25] Xilinx eSP Team (1994). End-Market Technologies. Xilinx, Inc. Retrieved May 25, 2002, Retrieved from <http://www.xilinx.com/esp/technologies/index.htm>

- [26] Ahmadi, H., Denzel, W. (1989). A Survey of Modern High Performance Switching. *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 7, September, 1091-1103.
- [27] Banniza, T., (1991). Design and Technology Aspects of VLSIs for ATM Switches. *IEEE Journal on Selected Areas in Communications*, vol. 9, October, 1255-1264.
- [28] Bianchi, G., Turner, J. (1993). Improved Queuing Analysis of Shared Buffer Switching Networks. *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, August, 482-490.
- [29] Bubenick, R., Turner, J. (1989). Performance of a Broadcast Packet Switch. *IEEE Transactions on Communications*, January, 60-69.
- [30] Yeh, Y., Hluchyj, M., Acampora, A. (1987). The Knockout Switch: A Simple, Modular Architecture for High-Performance Packet Switching. *IEEE Journal on Selected Areas in Communications*, vol. 5, no. 8, October, 1274-1283.
- [31] Tobagi, F., Kwok, T., Chiussi, F. (1991). Architecture, Performance, and Implementation of the Tandem Banyan Fast Packet Switch. *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, October, 1173-1193.
- [32] Chao, H. (1991). A Recursive Modular Terabit/Second ATM Switch. *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, October, 1161-1172.
- [33] Chao, H., Choe, B. (1995). Design and Analysis of a Large-Scale Multicast Output Buffered ATM Switch. *IEEE/ACM Trans. on Networking*, vol. 3, no. 2, April, 126-138.
- [34] Corazza, G., Raffaelli, C. (1993). Performance Evaluation of Input-Buffered Replicated Banyan Networks. *IEEE Transactions on Communications*, vol. 41, no. 6, June, 841-845.
- [35] Day, C., Giacobelli, J., Hickey, J. (1987). Applications of Self-routing Switches to Data Fiber Optic Networks. *Proceedings of the International Switching Symposium*, 519-423.
- [36] De Prycker, M., Somer, M. (1987). Performance of a Service Independent Switching Network with Distributed Control. *IEEE Journal on Selected Areas in Communications*, vol. 5, no. 8, October, 1293- 1301.
- [37] Hui, J., Arthers, E. (1987). A Broadband Packet Switch for Integrated Transport. *IEEE Journal on Selected Areas in Communications*, vol 5, no. 8, October, 1264-1273.

- [38] Jenq, Y. (1983). Performance Analysis of a Packet Switch Based on Single-Buffered Banyan Network. *IEEE Journal on Selected Areas in Communications*, vol. 1, no. 6, December, 1014-1021.
- [39] Lee, T. (1990). A Modular Architecture for Very Large Packet Switches. *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, July, 1097-1106.
- [40] Min, P., Saidi, H., Hegde, M. (1995). A Nonblocking Architecture for Broadband Multichannel Switching. *IEEE/ACM Transactions on Networking*, vol. 3, no. 2, March, 181-198.
- [41] Newman, P. (1988). A Fast Packet Switch for the Integrated Services Backbone Network. *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, December, 1468-1479.
- [42] Nojima, S., Tsutsui, E., Fukuda, H., Hashimoto, M. (1987). Integrated Services Packet Network Using Bus Matrix Switch. *IEEE Journal on Selected Areas in Communications*, vol. 5, no. 8, October, 1284-1292.
- [43] Pattavina, A. (1990). A Broadband Packet Switch with Input and Output Queuing. *Proceedings of the International Switching Symposium*, 11-16.
- [44] Tobagi, F., (1990). Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks. *Proceedings of IEEE*, vol. 78, no. 1, January, 133-167.
- [45] Turner, J. (1993). Queuing Analysis of Buffered Switching Networks. *IEEE Transactions on Communications*, vol. 41, no. 2, February, 412-420.
- [46] Stiliadis, D., Varma, A. (1997). A General Methodology for Designing Efficient Traffic Scheduling and Shaping Algorithms. *Proceedings of IEEE INFOCOM '97*.