1999

Architectural design options for ATM switches

Navindra Yadav
Lehigh University

Recommended Citation

Yadav, Navindra, "Architectural design options for ATM switches" (1999). *Theses and Dissertations*. Paper 603.

Yadov, Navindra

Architectural design options for ATM switches

May 31, 1999

Architectural Design Options for ATM Switches

By

Navindra Yadav

A Thesis

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Computer Science

Lehigh University

January 1999

This thesis is accepted and approved in partial fulfillment of the requirements for the Master of Science.

1/8/99

Date

Thesis Advisor

Chairperson of Department

Acknowledgments

I would like to express my sincere gratitude to my advisor Professor Michael J. Schulte, for his support, encouragement and advice. It was a great experience working in the computer architecture and arithmetic (CAAR) laboratory.

I would especially like to extend my thanks to my CAAR lab friends and colleagues. They made the lab, one of the best places I have worked. I would like to extend my sincere thanks to James Stine ("James is the best!!!"), Ahmet Akkas and Vitaly Zelov. I do not know what I would have done without them.

Last but not the least, I would like to express my gratitude and love to my mother and father for their constant support throughout my life. I would like to extend my love to Moni, my wife, for her patience and for waking me every day, even though she was more than a thousand miles away from me.

Table of Contents

List of Tables

List of Figures

Abstract

Asynchronous transfer mode (ATM) is rapidly becoming the primary transfer mode for implementing broadband integrated services digital networks (B-ISDN). These ISDN networks carry data, full motion video, audio, etc. Nowadays, networks with transfer rates of 100 mega bits per second (MBPS) to 622 MBPS are common, with users demanding even high transfer rates. Giga bits per second (GBPS) and tera bits per second (TBPS) networks are being designed and tested, to satisfy the increased bandwidth requirements.

ATM networks consist of two major parts; transfer medium and switches. The medium typically used is optical fiber, which can theoretically support infinite bandwidth. Hence, it is not the limiting factor, but rather ATM switches are the bottleneck to achieving high transfer rates. An ATM switch consists of two main components; switching elements and a switching fabric. The switching fabric is composed of identical switching elements, which are interconnected in a specific topology.

In this thesis, various ATM switch architectures are evaluated. Switching elements are classified as input queued, central queued, or output queued. Simulators are developed for the three different categories. The results generated from the simulators are analyzed and conclusions are drawn. Output queued switching elements, which are faster than central queued elements (because of less memory control overhead) can match the cell loss rate of central queued switching elements at the expense of a small increase in the amount of memory available.

A theoretical analysis of a multistage interconnection network, without internal cell loss based on stacked banyan-balancing networks, indicates that it is better than a sorting-trap-butterfly network by a factor of $\log_2(N)$ in terms of the number of switching elements each cell has to pass through.

1. Motivation

With the rapid growth of the Internet, the role of computer networks is becoming extremely important. Currently the drive in the computer industry is for computer hardware and software to achieve higher performance with greater reliability. In the design of high-speed networks, fast response time and high data bandwidth are critical.

Computer networks consist of two main components: media to transfer information and network elements. To meet the increased bandwidth requirements, the performance of both these components has to be improved. After the advent of fiber optical cables, which are theoretically capable of supporting infinite bandwidth, network elements have become the limitation in the attempt to increase the bandwidth of computer networks.

One of the most promising technology for high-speed data, voice and video transfer is asynchronous transfer mode (ATM). Within ATM networks, ATM switches have been identified to be the bottleneck [12]. Currently, several ATM switches exists that use different architectural design concepts. To make faster ATM switches, operating in the tera-bits per second range, the switch architecture needs to use the right design decisions.

The currently available technology has also considerably improved, for instance the capacity of dynamic random access memory (DRAM) improves by four times in every three years. If in the design of a switch a few years ago a design decision was taken because the size of the DRAM was not sufficient, then the same restriction may not hold today. In this thesis, various ATM switch architectural design concepts are evaluated by developing simulators for different switches. The performance of two switching fabrics is also evaluated.

2. Introduction to Asynchronous Transfer Mode

This chapter provides a brief introduction to the concept of asynchronous transfer mode. It describes the basic principles of ATM networks, parameters used to evaluate the performance and traffic control in ATM networks, and the ATM protocol layer stack.

2.1 An Overview of Asynchronous Transfer Mode

Asynchronous Transfer Mode (ATM) is the transfer mode for implementing Broadband Integrated Service Digital Networks (B-ISDN). The term transfer includes both transmission and switching aspects, so a transfer mode is a specific way of transmitting and switching information in a network. In ATM, all information to be transferred is packed into fixed size slots called cells, which are 53 bytes long. The first 5 bytes form the header and the next 48 bytes form the information field. Figure 2-1 shows the structure of an ATM cell. The term asynchronous, in the context of multiplexed connections means that cells allocated to the medium may exhibit an irregular recurrence pattern as they are inserted into the medium according to the actual bandwidth of the connection.

Examples of a synchronous transmission mode (STM) and an ATM transmission pattern are shown in Figure 2-2 and Figure 2-3, respectively. In STM, a data unit associated with a given channel is identified by its position in the transmission frame, while in ATM a cell associated with a specific virtual channel may occur at any position. This property provides ATM flexibility in allocation of bit-rate (i.e. bandwidth) to a specific connection or channel. Bit-rates in STM are restricted to a set of predefined channel bit-rates.

Networks are commonly categorized as circuit switched or packet switched networks. In circuit switched networks a connection is established between the source and the destination before any information is transferred. In packet switched networks every packet

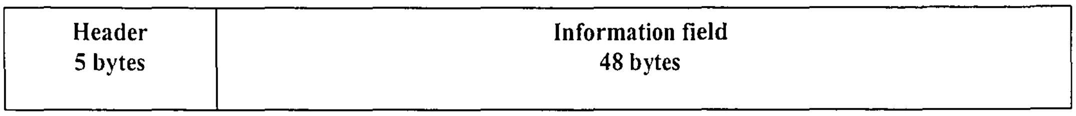


Figure 2-1: ATM cell structure.

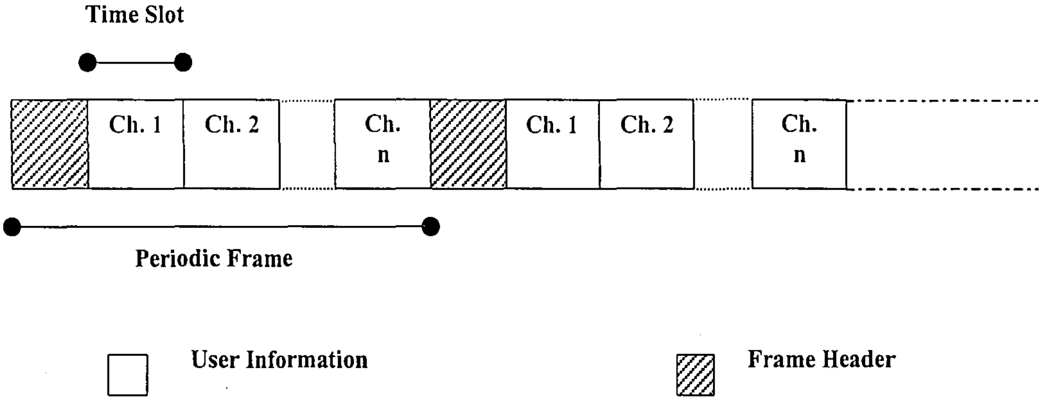


Figure 2-2: Synchronous Transfer Mode

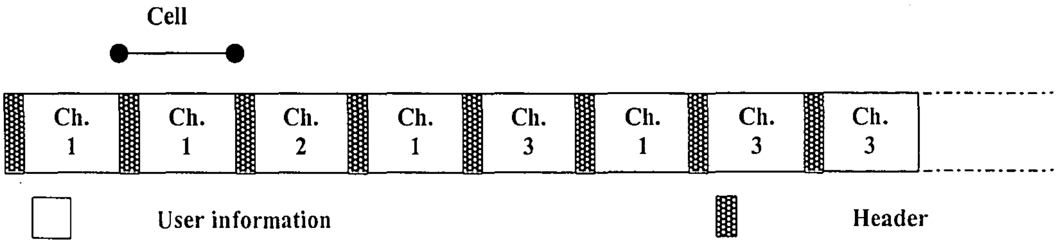


Figure 2-3: Asynchronous Transfer Mode

contains the address of the destination and is routed independently. Each of these techniques has their own advantages. ATM combines the advantages of both circuit and packet switched techniques. Circuit switched networks require a low processing overhead once a circuit is established, and the transfer delay of the information being carried is low and constant. Packet switched networks have more processing overhead while routing each packet, and the transfer delay of the packets is also not constant. However, packet switched networks can utilize the bandwidth of the network more efficiently, and are much more flexible in terms of the bit-rate assigned to individual connections.

ATM combines the advantages of both switching techniques by using the concept of virtual channels, which are connection-oriented channels. A connection-oriented channel is like a telephone call, because it involves a connection setup phase, an information transfer phase and a connection tear down phase. A connection within the ATM layer consists of one or more links, each of which is assigned an identifier. These identifiers remain unchanged for the duration of the connection. Also, cell routing is done in hardware to achieve low transfer delays.

ATM specifications guarantee cell sequence integrity under fault free conditions. Maintaining cell sequence integrity means that nowhere in the network can a cell belonging to a specific virtual channel connection overtake another cell belonging to the same virtual channel connection that was sent earlier.

2.2 Basic Principles of ATM

This section briefly describes the basic principles put forward by the International Telecommunication Union – Telecommunication (ITU-T) in [1].

2.2.1 Information Transfer

ATM is considered a packet oriented transfer mode based on asynchronous time division multiplexing and the use of fixed length cells. Each cell consists of an information field and a header. The header is primarily used to identify cells belonging to the same virtual channel within the asynchronous time division multiplex and to perform the appropriate routing. Cell sequence integrity is preserved on each virtual channel. The information field of ATM cells is carried transparently through the network. No processing, such as error control, is performed on the information field in the network. To accommodate various services such as voice, video, and data, several types of ATM Adaptation Layers (AAL) have been defined.

2.2.2 Routing

ATM is connection oriented. The header values are assigned to each section of a connection for the complete duration of the connection, and translated when switched from one section to another. Signaling information is carried on a separate virtual channel. Two sorts of connections are possible, virtual channel connections (VCC) and virtual path connections (VPC). A VPC can be considered to be an aggregate of VCCs. When switching, it must first be done based on the VPC and then on the VCC. For more details, refer to Section 2.3.

2.2.3 Resources

Since ATM is connection oriented, connections are established either semi-permanently or for the duration of the call in the case of switched services. This establishment includes not only allocation of a virtual channel identifier (VCI) or/and virtual path identifier (VPI), but also the allocation of the required resources for the user access and inside the network. These resources are expressed in terms of throughput (bit-rate) and quality of service (QOS). They maybe negotiated between the user and the network for switched connections, during the call setup phase or possibly during the call.

2.2.4 ATM Cell Identifiers

ATM cell identifiers, which include virtual path identifiers (VPI), virtual channel identifiers (VCI) and payload type identifiers (PTI), support recognition of an ATM cell on a physical transmission medium. Recognition of the cell is the basis for all further operations. VPI and VCI values are unique for cells belonging to the same virtual connection on a shared transmission medium. Within a particular virtual circuit, cells may be further distinguished by their PTI, which cannot be allocated freely, but depends on the type of payload carried by the cell. This field indicates whether the cell is carrying user information to be delivered transparently through the network or special network information. In case the field indicates network information, part of the information field indicates the type of network control, and the rest of the information field may be processed inside the network.

A number of pre-assigned ATM cell identifiers have been chosen in the ATM layer for particular cell streams. They are necessary for enabling communication with the network and performing network management. Other pre-assigned values define meta-signaling cells, point-to-point signaling cells and general-broadcast cells.

2.2.5 Throughput

Bandwidth has to be reserved in the network for each virtual connection. ATM offers the possibility to realize resource savings in the total bandwidth needed when multiplexing traffic for many variable bit-rate connections. The amount that can be saved, however, depends heavily on the number of multiplexed connections, on the burstiness of the traffic they carry, on the correlation between them, and on the quality of service (QOS) required. Peak cell rate (PCR) and sustainable cell rate (SCR) are used as throughput parameters. PCR is the maximum cell rate of the connection for the duration of the call. SCR is the maximum, mean cell rate measured over a period shorter than the duration of the call but longer than the cell inter-arrival interval.

2.2.6 Signaling

The negotiation between the user and the network with respect to the resources (e.g. VPI/VCI, throughput, QOS) is performed over a separate signaling virtual channel. The signaling protocol used over this channel is described in [2], [3], and [4].

2.2.7 Flow Control

There are three ATM layer service classes with respect to the bandwidth usage, i.e. constant bit-rate (CBR), variable bit-rate (VBR) and available bit-rate (ABR). An application requests for CBR, if its bandwidth needs are constant. A VBR application requires a variable amount of bandwidth. The bandwidth available in the network at any given time may not be completely allocated. ABR applications make use of this unallocated bandwidth. ABR applications can handle varying throughput and the possibility of increased delay.

To effectively manage the network bandwidth resource a fast signaling and control flow mechanism is used. There are two types of flow control mechanisms used in ATMs; generic flow control and resource management cells. More details about flow control can be found in [5].

2.3 Virtual Paths and Virtual Channels

Virtual channel (VC) is a unidirectional transport of ATM cells associated by a common unique identifier value. This identifier is called the virtual channel identifier (VCI) and is a part of the cell header. Virtual Path (VP) is a unidirectional transport of cells belonging to virtual channels that are associated by a common identifier value. This identifier is called the virtual path identifier (VPI) and is a part of the cell header. The relationship between virtual channels and a virtual path is show in Figure 2-4. A virtual path may carry one or more virtual channels.

A virtual channel link provides a means for unidirectional transport of ATM cells between the point where a VCI value is assigned and the point where it is translated or removed. The points where a VPI is assigned and translated terminate a virtual path link. A concatenation of VC links is called a virtual channel connection (VCC) and a concatenation of VP links is called a virtual path connection (VPC).

Switching of virtual channel connections / virtual path connections is achieved by translating one or both of the values of VCI/VPI in a VC/VP switching entity. VP switches terminate VP links and therefore have to translate incoming VPIs to the corresponding outgoing VPIs, according to the destination of the VP connection. The VCI values remain unchanged. Figure 2-5 shows a VP switch. A VC switch terminates both VC links and VP links and performs VPI and VCI translation. Figure 2-6 shows a VC switch.

2.4 ATM Cell Header structure

Figure 2-7 shows the structure of an ATM cell header at the network node interface. The first field is the routing field, which is subdivided into twelve-bit and sixteen-bit fields. The twelve-bit field holds the VPI and the sixteen-bit field holds the VCI. The payload type identifier (PTI) is coded in three bits. The cell loss priority (CLP) field consists of one-bit and indicates whether the cell has a higher priority (CLP=0) or is subject to discarding by the network (CLP=1). Finally, the header error check (HEC) field consists of eight bits and is used for error detection and correction on the header.

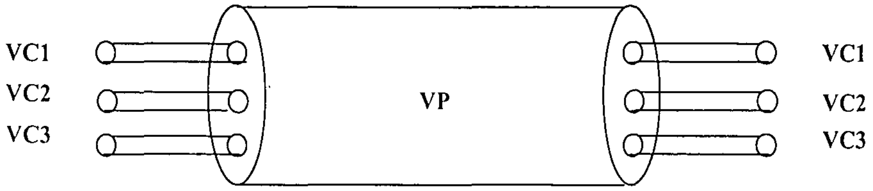


Figure 2-4: Relationship between a virtual path and virtual channel.

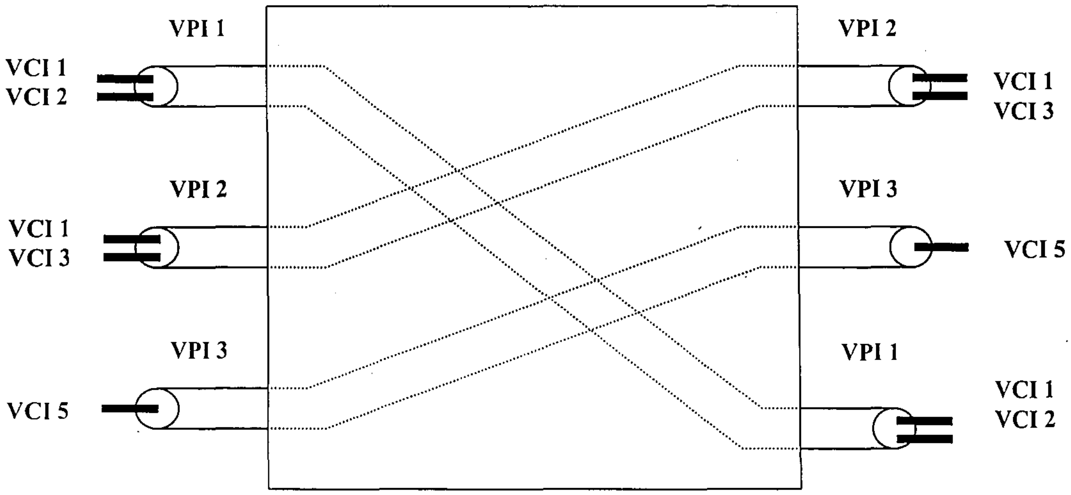


Figure 2-5: Virtual path switch

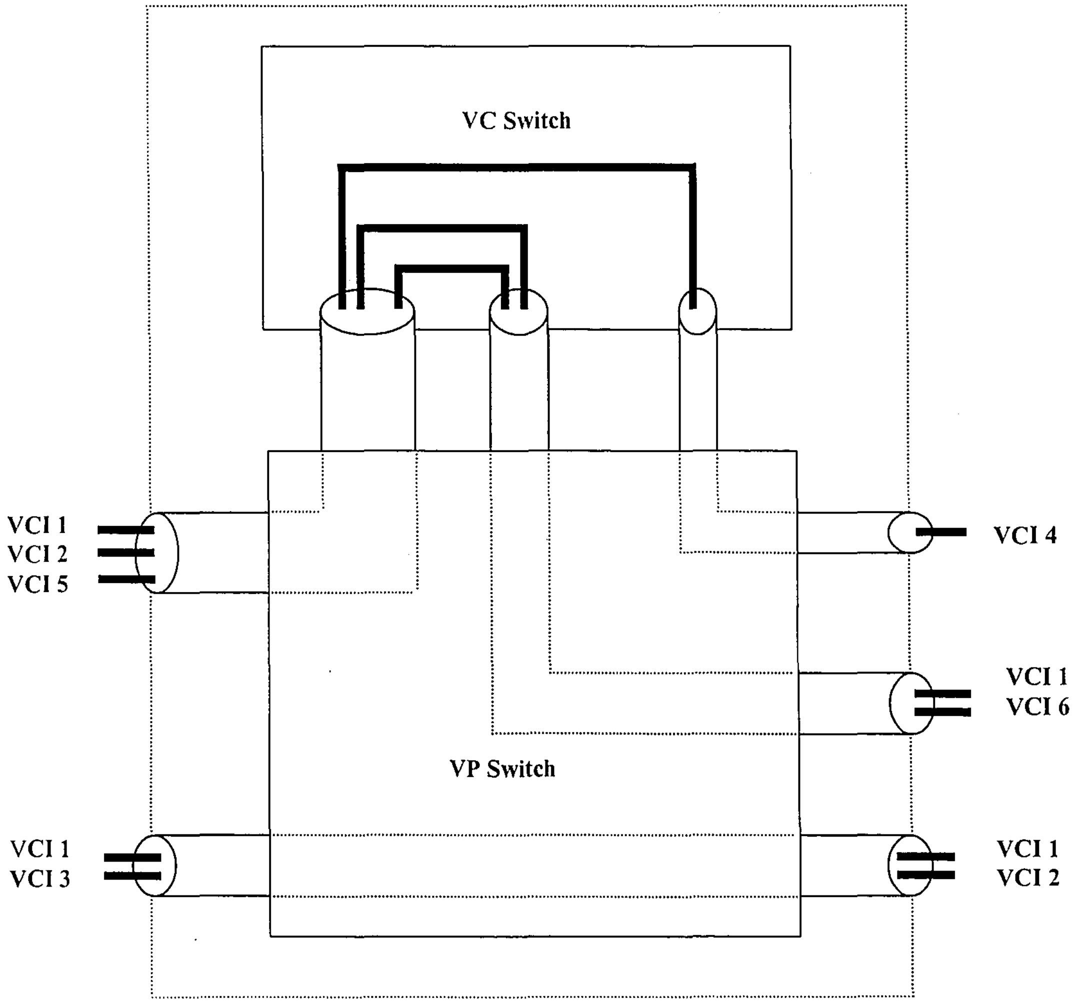
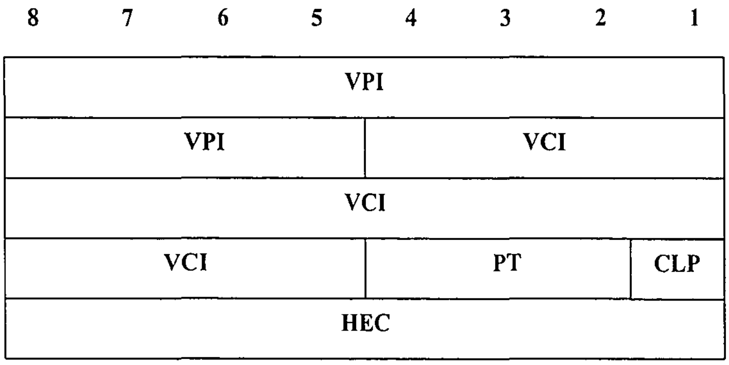


Figure 2-6: Virtual channel / Virtual path switch



VPI: Virtual Path Identifier (12 bits)
VCI: Virtual Channel Identifier (16 bits)
PT: Payload Type (3 bits)
CLP: Cell Loss Priority (1 bit)
HEC: Header Error Check (8bits)

Figure 2-7: ATM Cell header (Network-Network Interface)

2.5 Broadband Network Performance

Broadband networks based on ATM cell transfer must meet certain performance requirements. ATM performance parameters and measures need to be specified in addition to the performance parameters of existing networks. These parameters are knows as quality of service parameters. When a connection establish request is made, a set of minimum acceptable quality of service (QOS) parameters is also specified. A connection is established only if the network can guarantee the QOS parameters. Assume 'A' and 'B' are the two endpoints of a virtual channel in an ATM network. Cells belonging to the virtual connection are delivered from 'A' to 'B'. Because there is some transfer delay, the cell sent from 'A' arrives at 'B' within some time $\delta > 0$. A cell exit event occurs at 'A' when the first bit of the cell leaves 'A' and a cell entry event occurs at 'B' when the last bit of the cell reaches B.

In order to adequately describe the quality of ATM cell transfer, the following outcome categories are defined [6]:

1. **Successfully Transferred Cell :** If δ is less than a maximum allowed time T and the cell is not affected by bit errors, then the cell has been successfully transferred.
2. **Erroneous Cell :** If the cell arrives in due time but there are one or more bit errors in the received cell information field, the cell is erroneous.
3. **Lost Cell :** A lost cell outcome occurs if the cell arrives after time T or never reaches 'B'. Errors in the ATM cell header that cannot be corrected or cell buffer overflows in the switches lead to lost cells.
4. **Misinserted Cell :** If a cell that has not been sent from 'A' on the virtual channel to 'B', arrives at 'B' on the 'A' to 'B' virtual channel, then this miss-delivered

cell produces a misinserted cell outcome. Header errors that are not detected or are erroneously corrected may produce miss-inserted cells.

5. **Severely Erroneous Cell Block :** A severely erroneous cellblock occurs if more than M erroneous cells are observed in a block of N cells transmitted consecutively on a given connection.

By making use of the above considerations, performance parameters for the network are defined. The parameters and their definitions are listed in Table 2-1.

Parameter	Definition
Cell loss ratio	Ratio of lost cells to transmitted cell
Cell misinserted rate	Number of misinserted cells per connection per second
Cell error ratio	Ratio of erroneous cells to the number of delivered (successful + erroneous) cells
Severely erroneous cell block ratio	Ratio of the number of severely erroneous cell blocks to total number of cell blocks (fixed number of cells)
Mean cell transfer delay	Arithmetic average of a specified number of cell transfer delays
Cell delay variation	Difference between a single observation of cell transfer delay and the mean cell transfer delay on the same connection

Table 2-1: ATM performance parameters

Bit errors in erroneous cells can be corrected to a certain extent by error correction methods. Lost or misinserted cells can cause severe problems if they are not detected. For example, when using a constant bit-rate, the real time services synchronism between sending and receiving terminals may be disturbed. This can be overcome by using sequence numbers. Cell transfer delay and cell delay variation must be kept within a limited range in order to meet service requirements. Cell transfer delay and ATM switches, due to buffering introduce cell delay variation.

2.6 ATM Layer Model

ATM has three layers; the physical layer (PL), the ATM layer and ATM adaptation layer (AAL). It contains three planes: a user plane to transport user information, a control plane mainly composed of signaling information, and a management plane used to maintain the network. In addition, a third dimension is added to the protocol reference model (PRM), called the plane management, which is responsible for the management of different planes. Figure 2-8 shows the PRM. ITU-T recommendation I.321 describe the PRM for ATM [7].

The physical layer (PHY) transports information (bits and cells). The ATM layer mainly performs switching, routing and multiplexing. The AAL is mainly responsible for adapting service information to the ATM stream. Each of the layers can be further subdivided into sub-layers. Each sub-layer performs a number of functions, as shown in Table 2-2.

2.6.1 Physical Layer

The Physical layer is further sub-divided into the physical medium (PM) sub-layer which supports medium dependent bit functions, and the transmission convergence (TC) sub-layer which converts the ATM cell stream into bits to be transported over the physical medium. The physical layer for the UNI is described in [8].

2.6.1.1 Physical medium sub-layer

The physical medium sub-layer is responsible for the correct transmission and reception of bits on the appropriate physical medium. This sub-layer must guarantee a proper bit timing reconstruction at the receiver. The transmitter is responsible for inserting the required bit timing information and line coding.

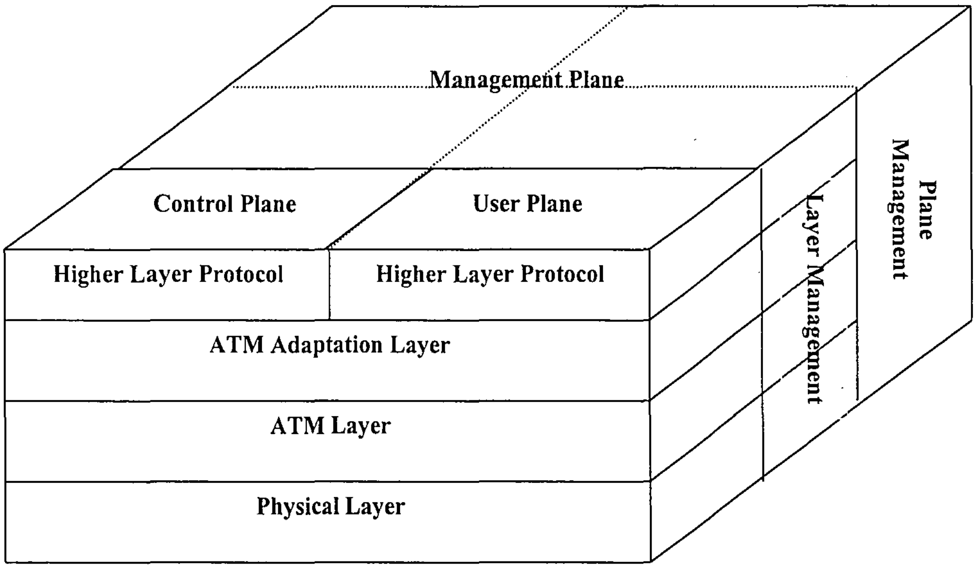


Figure 2-8: ATM Protocol Reference Model

Layer	Sub Layer	Function
AAL	Convergence Sub-layer (CSL)	Message identification and time/clock recovery
	Segmentation and Re-assembly Sub-layer (SAR)	Segment and re-assemble higher layer information
ATM		Generic flow control
		Cell VPI / VCI translation
		Cell multiplex / de-multiplex
		Cell rate de-coupling (with unassigned cells)
PHY	Transmission Convergence	Cell rate de-coupling (with idle cells)
		Header error check (HEC) generation / verification
		Cell scrambling / de-scrambling
		Cell delineation (using HEC)
		Path signal identification
		Frequency justification
		Frame scrambling / de-scrambling
		Frame generation / recovery
	Physical Medium	Bit timing
		Line coding
		Physical medium dependent scrambling / de-scrambling

Table 2-2: ATM protocol reference model sub-layers and functions

2.7.1.2 Transmission convergence sub-layer

The first function performed by this sub-layer is the adaptation to the transmission system used, such as synchronous digital hierarchy (SDH), plesiochronous digital hierarchy (PDH) or cell based. The second function of the sub-layer is generation of the header error check (HEC) for each cell at the transmitter and its validation at the receiver. During startup, to detect the proper cell boundary, an ATM network element needs to validate a number of HEC values for each cell. If it can successfully perform this check on a specific number of cells it means that the correct cell boundary has been detected. Once cell delineation has been found, an adaptive mechanism uses the HEC for correction and detection of cell header errors. Isolated single bit errors are corrected, but as soon as multiple consecutive cells show header errors, correction is given up for higher precision detection and elimination of cells with errors. To avoid the slipping of cells with undetected multiple header errors during periods of bit error bursts. Such errors may go undetected by the correction algorithm. Finally, this sub-layer must ensure insertion and suppression of unassigned cells to adapt the useful cells to the available payload.

2.6.2 ATM Layer

The ATM layer is independent of the physical medium used to transport the ATM cells. The ATM layer is responsible for performing the following functions.

- The multiplexing and de-multiplexing of cells of different connections identified by different VCI and VPI values, into a single cell stream on a physical layer.
- Translating the cell identifier, which is required in most cases when switching a cell from one physical link to another in an ATM switch. This translation can be performed on the VCI, the VPI, or both.
- Providing the user of a VCC or VPC with one QOS class, out of a number of classes supported by the network. Some services may require a certain QOS for one part of the connection, and a lower QOS for the remainder. The distinction within the connection is made by means of the CLP bit in the cell header.
- Performing management functions, such as setting the PTI field in the cell header to indicate congestion. When the PTI does not indicate user information, more information related to layer management is present in the information field of the cell.
- Extracting of the cell header before the cell is delivered to the adaptation layer and adding the cell header after the cell is delivered from the adaptation layer to be transmitted.

- Implementing a flow control mechanism on the user network interface using the generic flow control bits in the cell header.

2.6.3 ATM Adaptation Layer

The ATM adaptation layer enhances the service provided by the ATM layer to a level required by the next higher layer. It performs functions for the user, control, and management planes, and supports the mapping between the ATM layer and the next higher layer. The functions performed by the AAL layer depend on the higher layer requirements.

The AAL layer is divided into two sub-layers: the segmentation and re-assembly (SAR) sub-layer and the convergence sub-layer (CS). The main function of the SAR sub-layer is to segment the higher layer information into suitable size payloads for the consecutive ATM cells of a virtual connection on the transmit side. On the receiver side re-assembly of the contents of the cells is performed. The convergence sub-layer performs functions like message identification, time/clock recovery, etc. If an application finds the service provided by the ATM layer sufficient, then the AAL layer may be empty.

There are four classes of AAL layers. To obtain these four classes the services are classified according to three basic parameters.

- Time relation between source and destination: Some services have a time relation between the source and destination, while for others there is no such time relation. For instance, in 64 KBPS PCM voice, there is a clear time relation between the source and destination. Information transfer between computers has no time relation. Services with a time relation are also called real time services.
- Bit-rate: Some services have a constant bit-rate, while others have a variable bit-rate.

- Connection mode: Services can be either connectionless or connection oriented. Connectionless mode is similar to packet switching, whereas connection oriented mode is similar to circuit switching. In connection oriented mode, all the information is sent through the same path decided upon during connection setup. In connectionless mode this is not true.

Only four of the theoretical eight combinations of the three parameters yield existing services. For details refer to ITU-T 's I.362 recommendation [9]. Table 2-3 gives a quick reference.

- In Class A, a time relation exists between source and destination. The bit-rate is constant and the service is connection oriented. Typical examples include a 64 KBPS voice transmission and a fixed bit-rate video transmission.
- In Class B, again a time relation exists between the source and the destination, for a connection oriented services. However, the difference from class A is that Class B sources have a variable bit-rate. Examples of this include variable bit-rate audio and video.
- In Class C there is no time relation between the source and the destination and the bit-rate is variable. The service is connection oriented. Examples of this include connection oriented data transfer and signaling.
- Class D differs from Class C in that it is connectionless. An example of such a service is connectionless data transport, such as video broadcast.

	Class A	**Class B**	**Class C**	**Class D**
Timing relation between source and destination	Exists		Does not exist	
Bit-rate	Constant	Variable		
Connection mode	Connection oriented			Connectionless

Table 2-3: Service classes for ATM adaptation layer

3 ATM Switches

3.1 Introduction

In the past, various switching architectures were developed for different applications, such as voice and data, based on transfer modes like synchronous transfer mode (STM) and packet transfer mode (PTM). The switching architectures developed for STM and PTM are not directly applicable to broadband ATM [10].

The two main factors that have a large impact on the implementation of broadband ATM switching architectures are

- The high speeds at which the switch has to operate (from 150 to 600 MBPS)
- The statistical behavior of the ATM streams passing through the ATM switch

In addition, the small fixed cell size and a limited functionality cell header have an important influence on the definition of an optimal ATM switching architecture. A number of ATM switches are commercially available from various telecommunication companies ranging from very small (four inputs and outputs) to very large (thousands of inputs and outputs).

In an ATM switch, the ATM cells are transported from an input port (out of N input ports) to one or more output ports (out of M output ports). This switching from input port to output port might be combined with concentration, expansion, multiplexing and de-multiplexing of ATM traffic.

3.2 Switching

Switching is the transportation of information from an incoming ATM channel to an outgoing logical channel. A logical channel is characterized by

- A physical input port / output port having a physical port number

- A virtual channel identifier (VCI) and virtual path identifier (VPI)

To perform switching, both the input port and incoming virtual channel/path identifier have to be related to the output port and outgoing virtual channel/path identifier. In any ATM switching system these two functions have to be implemented and can be compared to functions applied in classical STM switching systems, as described in [11]. Relating the ports is comparable to a space switching function (S), as shown in Figure 3-1. Information from input port 1 is transported to output port 3, whereas information from input port 3 is transported to output port M. An important aspect of space switching is routing. Relating virtual channel/path identifiers is comparable to a time slot interchange (T) in a time switch as shown in Figure 3-2. Information from time slot i is switched to time slot j, whereas information from time slot k is transported to time slot 1.

Modern STM switches are often composed of these S and T stages, in different sorts of combinations like STSTS, STSSTS, or TSTTST. In ATM switching systems, however, the time identification in a fixed frame is replaced by logical channel identification. Since the pre-assigned time slot concept disappears in ATM switching systems, contention problems arise if 2 or more logical channels contend for the same time slot. This can be solved by temporarily queuing the ATM cells before sending them out. This queuing function is the second important aspect of ATM switching systems.

3.3 Concentration / multiplexing

In ATM, the distinction between multiplexing and concentration is rather vague. Multiplexing is used when the emphasis is put on the statistical merging of different ATM virtual channels (cell streams) on a single ATM stream. Concentration is used when the stress is on the reduction of a number of input ports to a smaller number of output ports.

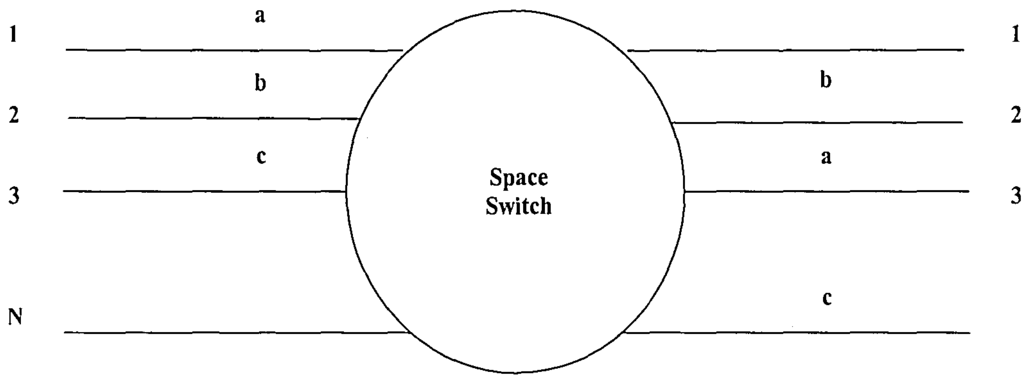


Figure 3-1: A Space Switch

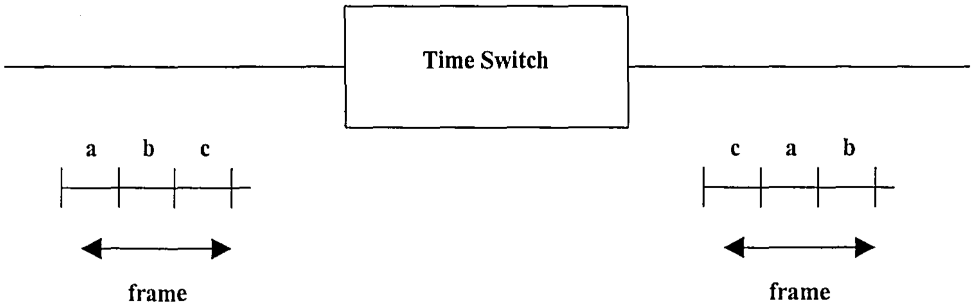


Figure 3-2: A Time Switch

3.4 Basic Function of an ATM Switch

The basic function of an ATM switch is shown in Figure 3-3 and described in detail in [12]. Incoming ATM cells are physically switched from an input port I_i to an output port O_j, and at the same time their header value is translated from an incoming value of α to an outgoing value of β. On each incoming and outgoing link, the values of the header are unique, but identical headers can be found on different links. For example, in Figure 3-3 and Table 3-1, x appears on two input ports I_1 and I_n. All cells which have a header equal to y on incoming link I_1 are switched to output port O_q and their header value is changed to m. All cells with a header x on link I_n are switched to output port O_1 and their header value is set to n.

Two main tasks that need to be performed by an ATM switch are:

- VPI / VCI translation
- Cell transport from its input to its dedicated output

An ATM switch performs VPI / VCI translation by modifying the cell header. Cell transport from its input to its dedicated output is achieved by space switching. It is possible for two cells arriving simultaneously from different input ports of the ATM switch, destined for the same output port (O_1). To prevent the two cells destined for the same output port from colliding and corrupting each other, either one of them has to be dropped or queued. Since a low cell loss rate is desired for ATM switches, cells are dropped only as a last resort. Buffers are incorporated into the design of an ATM switch to queue the cells that cannot be routed.

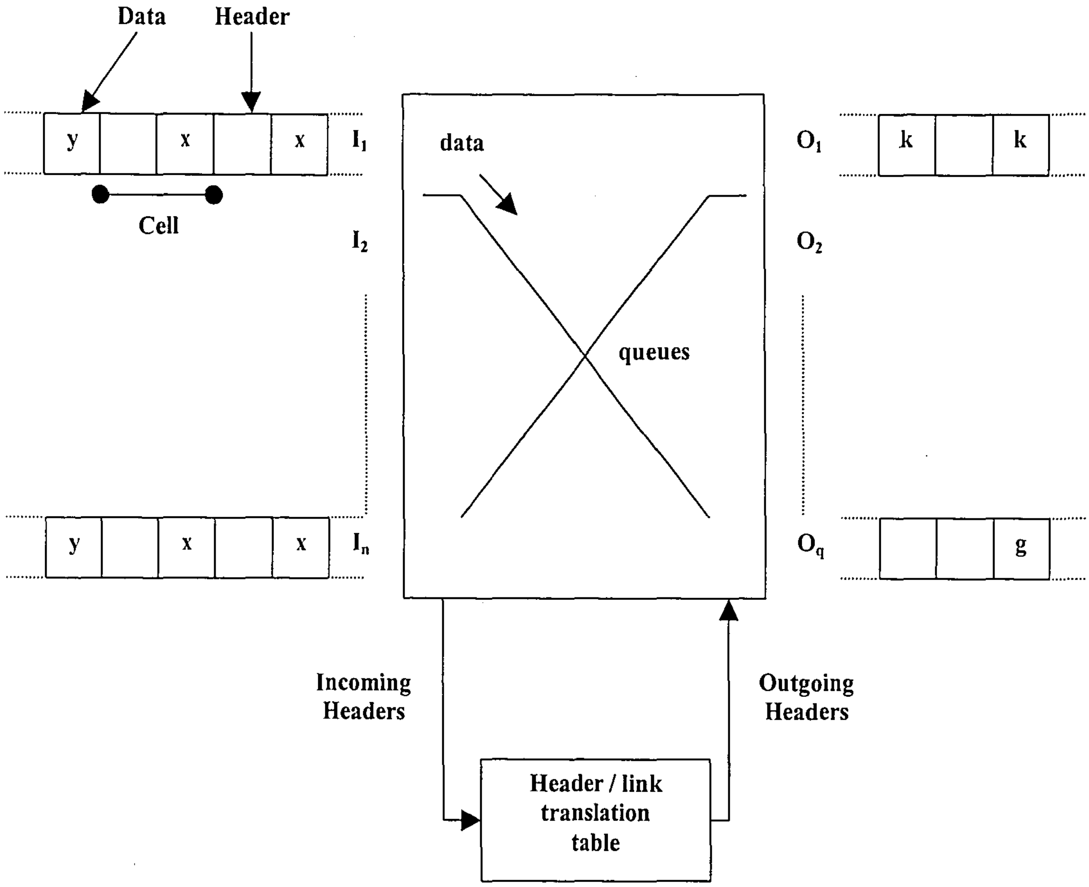


Figure 3-3: ATM switching principle

Incoming link	Header	Outgoing link	Header
I_1	X	O_1	K
	Y	O_q	M
	Z	O_2	L
I_n	X	O_1	N
	Y	O_2	I
	S	O_q	G

Table 3-1: Translation table

To conclude, an ATM cell performs the following three functions: routing (space switching), queuing and header translation. The way these functions are implemented and where in the switch these functions are located distinguishes one switch from another.

3.5 Switching Requirements

Broadband networks must be capable of transporting all kinds of information, ranging from voice to high quality video. These services have different requirements in terms of bit-rate (from a few KBPS up to hundreds of MBPS), behavior in time (constant bit-rate or variable bit-rate), semantic transparency (cell loss rate and bit error rate) and time transparency (delay and delay jitter). These different service requirements have to be met by broadband ATM switches.

3.5.1 Information Rate

Since the information rates of the different services are very diverse, a large number of information rates must be supported in broadband switches. These rates range from a few KBPS (e.g. for voice) up to values of 150 MBPS (e.g. for: high definition television (HDTV)). If an ATM switch operates at 150 MBPS, it does not mean that the switch internally operates at 150 MBPS. Its internal operating speed may be lower or greater than 150 MBPS depending on its architecture. For example, switching can be realized over parallel wires resulting in a lower internal speed, or several 150 MBPS input ports can be multiplexed on a single link resulting in a higher internal speed.

3.5.2 Performance

The performance of classical STM switches is mainly characterized by the switch's throughput, connection blocking probability, bit error rate and switching delay, as described in [13]. In an ATM environment, however, two other parameters are important, namely the cell loss or cell miss-insertion probability and the delay jitter. In ATM switches, as in STM switches, the fabrication technology and dimension of the switch mainly determine the throughput and the bit error rate [14] [15]. The effect of connection blocking, cell loss / miss-insertion probability and switching delay are different in ATM switches and are described below.

3.5.2.1 Connection blocking

Since ATM is connection oriented during the connection set-up phase, a logical connection needs to be established between a logical input port and output port. The probability of not finding enough resources between the input and output ports of the switch to guarantee the quality of all existing connections and the new connection is defined as connection blocking.

Some switch implementations do not have internal connections. This means that if enough resources (i.e. bandwidth and header values) are available on the input port and the output port of the switch, no new connection is blocked. Thus, a new connection is always accepted if enough resources are available on the external links, without an explicit check of the internal switch resources. Other switch implementations that have internal connections exhibit internal connection blocking. Internal connection blocking occurs when there are not enough resources for the new connection that can be allocated within the switch. Their dimension, the number of existing internal connections, and the load on those connections determine the blocking probability of internal connection blocking switches.

3.5.2.2 Cell loss and cell miss-insertion probability

In ATM switches, it is possible for many cells to be destined for the same link (this link can be internal or external to the switch). This may cause more cells than the queue's capacity to compete for the queue's storage space, resulting in cells being dropped or lost.

The probability of losing a cell must be kept within limits to ensure a high semantic transparency (clear communication between the transmitting and receiving application). Typical values for cell loss probability for ATM switches range between $10^{-8}$and 10^{-11}.

Some switch architectures are designed in such a way that they do not suffer from cells competing for the same resource internally (e.g. a queue). These architectures do not lose ATM cells internally, but may lose cells only at their input and output ports. These switches are called internally non-blocking switches.

Sometimes cells in an ATM switch may be misrouted only to arrive erroneously on another logical connection. This occurs when the cell header gets corrupted during transmission and it goes undetected through the error checking stage. The probability of this cell miss-insertion must be kept within limits. Typical, cell miss-insertion values are a factor of 1000 times less than the cell loss rate.

3.5.2.3 Switching Delay

Switching delay is the time to transmit an ATM cell through the switch. Typical values for the switching delay of ATM switches range between 10 and 1000 micro-seconds with a jitter of few 100 micro-seconds or less, described in [6]. Jitter is determined by the probability that the delay of the switch will exceed a certain value. For example a jitter of 100 microseconds for 10^{10} cells, means that the probability that the delay in the switch is larger than 100 microseconds is smaller than 10^{-10}.

3.6 Basic Architecture of an ATM Switch

The basic architecture of a butterfly based ATM switch is shown in Figure 3-4. In general an ATM switch consists of two parts: switching elements and switching fabric.

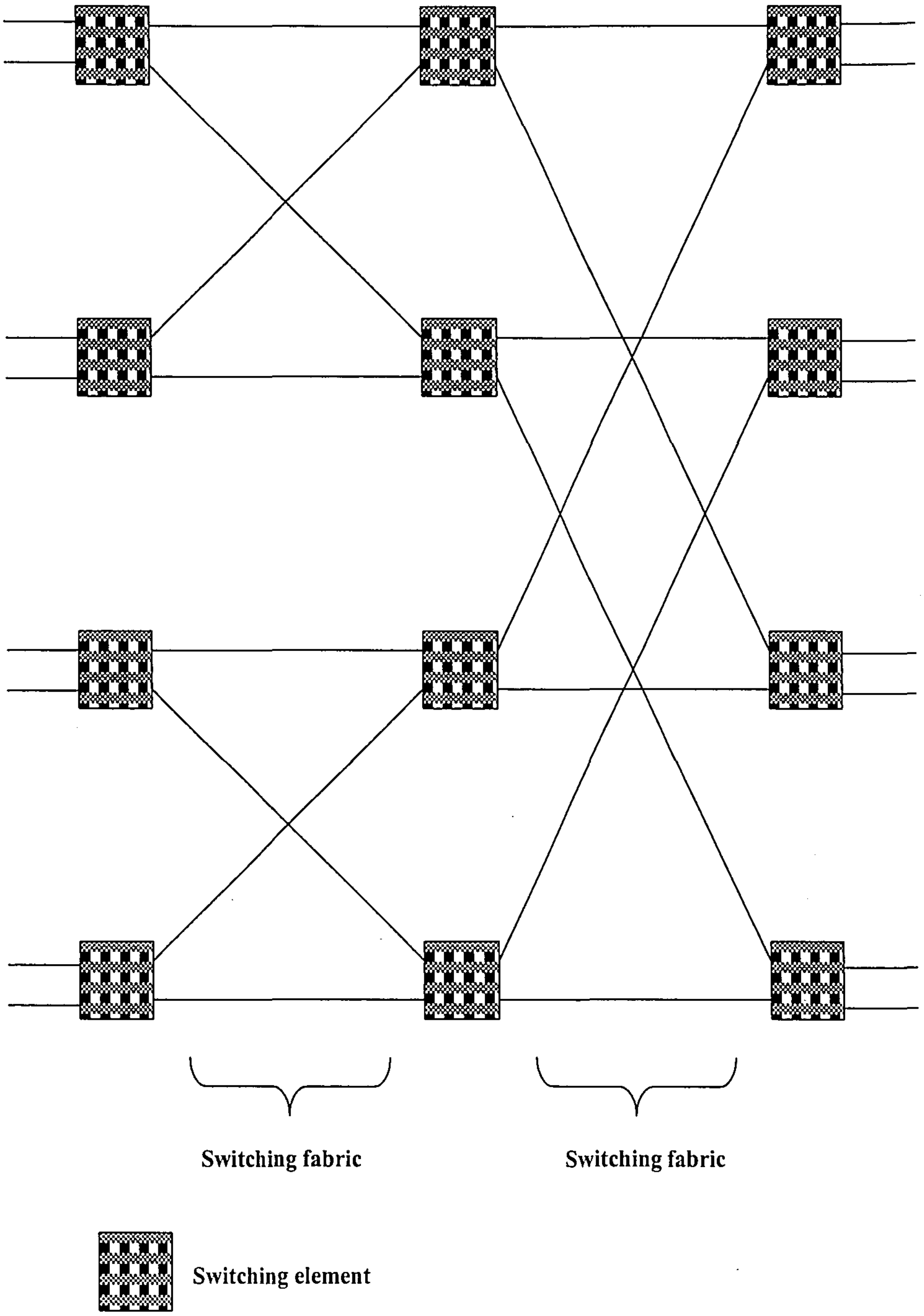


Figure 3-4: Butterfly switching fabric with switching elements at the its nodes

The switching element is the generic building block used to construct an ATM switching fabric. Switching elements are also called basic switching blocks [16]. A switching fabric is composed of identical basic switching elements interconnected in a specific topology. Thus, a switching fabric is defined when its topology is determined and when the switching elements are defined.

3.7 Switching Elements

ATM switching elements have a small number of input and output ports. Typical sizes for switching elements are 2 by 2 to 16 by 16. The operating speed of most switching elements ranges from 150 MBPS to 2.4 GBPS or greater. The size (number of input and output ports) and speed depend on the fabrication technology and the level of integration used.

This section discusses the queuing problems of a switching element. If two ATM cells arrive at two input ports of the switching element for the same output during the duration of one cell, one of them has to be queued for a later cell time. Depending on the particular architecture of the switching element and the required internal speed, it is possible to queue cells at the input port, output port, or internally in the switching element [17]. There are some switching element architectures which have a fully non-blocking (per cell) switch fabric. Such architectures do not need to maintain internal queues. This contention-free characteristic guarantees that if all incoming cells of the switch architectures are destined for a different output port, no internal contention will arise. Switching elements which buffer cells destined for the same output port can be categorized into three categories determined by the physical location of the buffers: at the input, at the output, or central in the switching element [18].

3.7.1 Input Queued Switching Element

The block diagram of a switching element with input queuing is shown in Figure 3-5. In an input queued switching element all incoming cells are queued in a buffer at the input port. The arbitration logic determines which cell to serve in case of contention. The switching transfer medium then transfers the ATM cells from the input queues to the output port without internal contention. The arbitration logic which decides which input port to serve can range from very simple, such as round robin to quite complex, such as, taking into account the input buffer filling levels, delays of cells, etc.

Input queued switching elements have a major disadvantage called head of the line (HOL) blocking. Suppose that the cell of input port i is selected to be transferred to output port p. If input port j also has a cell destined for output port p, this cell will be stopped, together with all following cells. Suppose that the second cell in the queue of input port j is destined to an output port q for which there is currently no cell waiting in the other queues. Then this cell cannot be served, since the cell in front of it in the queue is blocking the transfer.

3.7.2 Output Queued Switching Element

The block diagram of a switching element with output queuing is shown in Figure 3-6. In the output queued switches, cells of different input ports destined to the same output port are transferred during one cell time to their corresponding output port. Only a single cell is served at the output port. If there are more than one cells destined to the same output port, a contention problem arises, which is solved by having a separate queue for each output port.

In principle, cells can arrive simultaneously at all input ports destined to a single output port. To ensure that no cells is lost in the switching transfer medium before it arrives at the output queue, the cell transfer must be performed at N times the speed of the input ports.

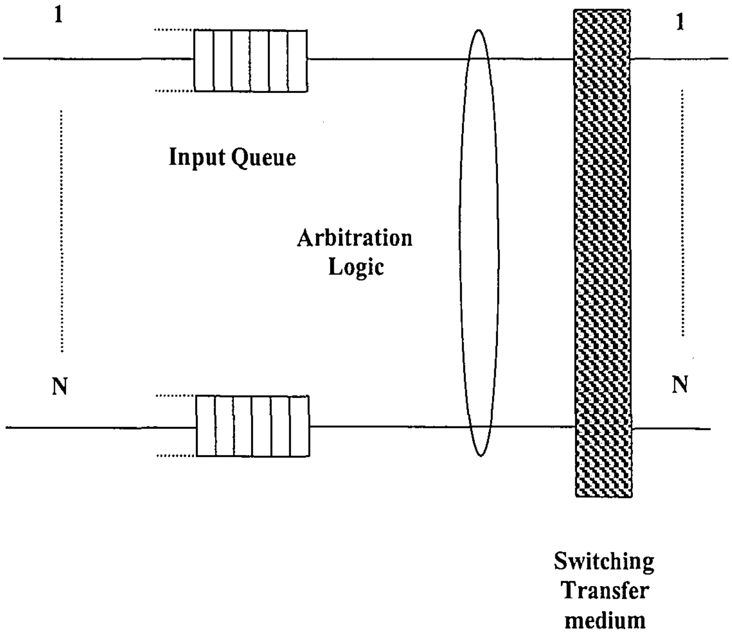


Figure 3-5: Switching element with input queues

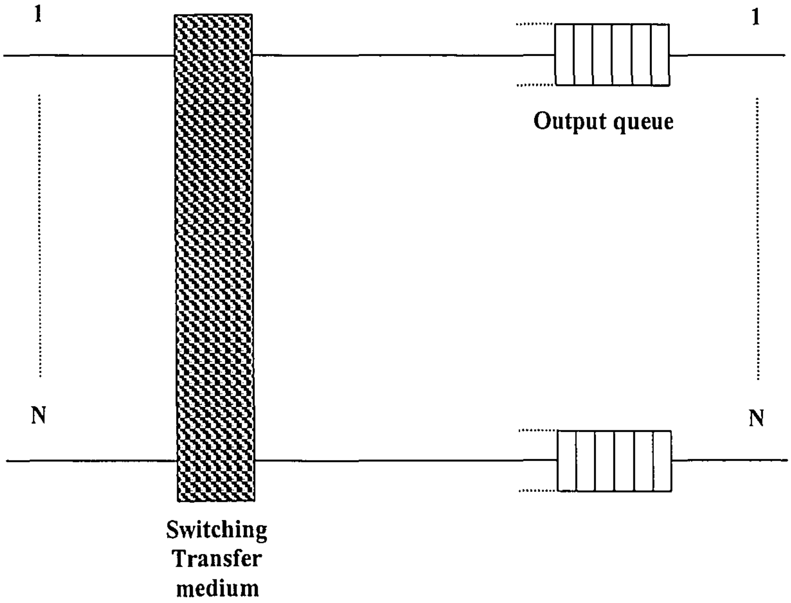


Figure 3-6: Switching element with output queues

The system must be able to write N cells in the queues during one cell times. In the switching transfer medium, no arbitration logic is required. The control of the output queues is based on a simple first in first out (FIFO) discipline to ensure that cells remain in the correct sequence.

3.7.3 Central Queued Switching Element

The block diagram of a switching element with central queuing is shown in Figure 3-7. In the central queued approach, the buffers are not dedicated to a single input port or output port, but shared between all input and output ports. Each incoming cell is directly stored in the central queue. Every output port selects the cells destined for it from the central memory in a FIFO discipline.

Internally, provisions are made to ensure that the output ports know which cells are destined for them. The read and write discipline of the central queue is not a simple FIFO discipline, since cells for different destinations are all merged in to a single queue. This means that the central memory is addressed in a random manner. The logical queues apply the FIFO discipline. Since cells can be written and read at random memory locations, a rather complex memory management system has to be used.

3.8 Switching Fabrics

This section gives a general classification of switching networks [19]. A survey of various switches can be found in [20], [21], [22]. Figure 3-8 gives a classification of switching networks. Switching fabrics are categorized into two general categories depending on the number of stages. A switching fabric is also known as a switching network.

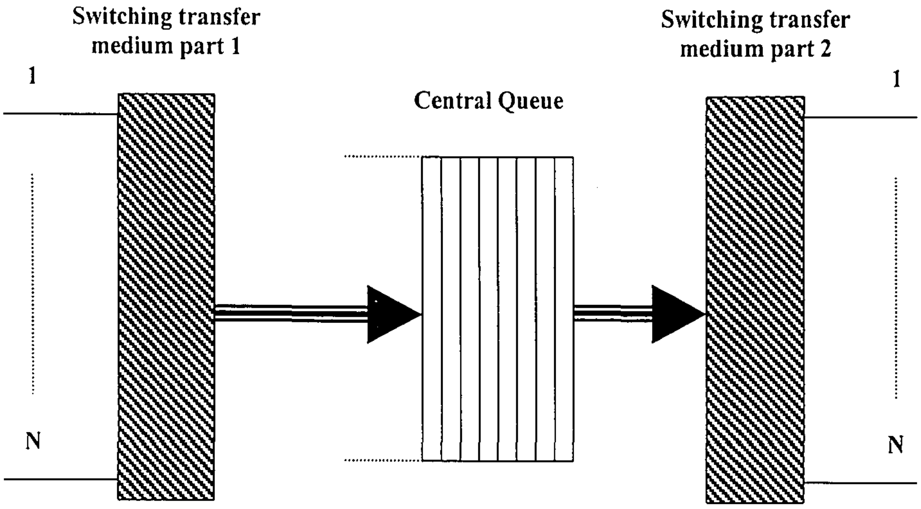

Switching transfer medium part 1
Central Queue
Switching transfer medium part 2
1
N
1
N

Figure 3-7: Switching element with central queuing

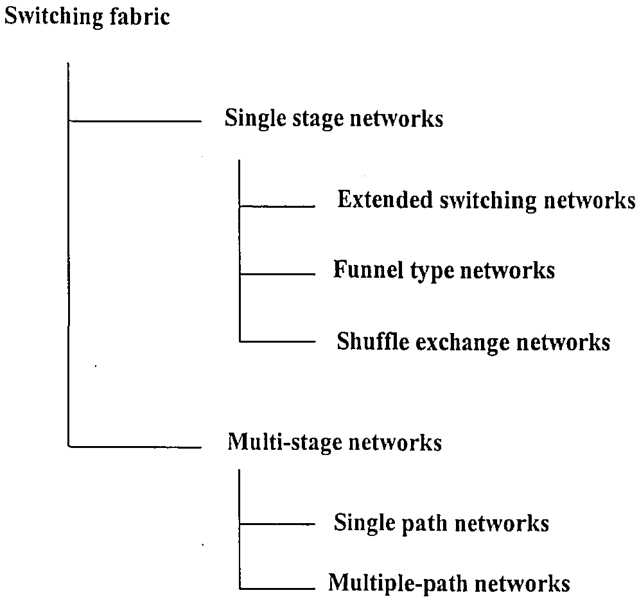


Figure 3-8: Classification of switching networks

3.8.1 Single Stage Networks

A single stage network is characterized by a single stage of switching elements connected to the inputs and output ports of the switching network / fabric. Categories of single stage fabrics are extended switching matrices, funnel type networks and shuffle-exchange networks. Single stage fabrics are useful for constructing smaller switches. With the increase in the number inputs, the number of switching elements also has to be increased, which in turn increases the delay in switching cells. To construct switches with 128 or more input / output ports, multi-stage fabrics are preferred.

3.8.2 Multi-Stage Networks

Multi-stage networks are built from several stages which are interconnected in a certain link pattern. According to the number of paths available for reaching a destination output port from a given input port, these fabrics are classified into two groups called single-path and multiple-path networks.

3.8.2.1 Single Path Networks

In single path networks, there is only one path to the destination from a given input. These networks are also called Banyan networks [23]. Since there is only one path to the destination port, routing is simple. The disadvantage is that internal blocking can occur as an internal link can be used simultaneously by different groups. Banyan networks can further be classified into sub groups.

In *L* level Banyan networks, only the switching elements of adjacent stages are interconnected. Each path in the network passes through exactly *L* stages. *L* level Banyan networks are further subdivided into regular and irregular Banyan networks. Regular Banyan networks are constructed from identical switching elements, whereas irregular Banyan networks use different types of switching elements. Regular Banyan networks have an advantage that they can be implemented easily because of they are constructed from identical elements.

3.8.2.2 Multiple Path Networks

In multiple-path networks, a multiplicity of alternative paths exists to the destination output from a given input port. This property has the advantage that internal blocking can be reduced or avoided altogether [24]. In most multiple-path networks, the internal path is determined during the connection setup phase. All cells on the connection use the same internal path. As all cells follow the same path, switching elements can uses a FIFO policy without worrying about cell sequence integrity.

Multiple path networks are further subdivided into folded and unfolded networks. In folded networks, all inputs and outputs are located at the same side of the switching network and the network's internal links operate in both directions. Folded networks have the advantage that the paths are short [25] [26]. For example, if the input and output ports are connected to the same switching element, cells can be reflected at the switching element and need not be passes to the last stage. The number of switching elements that the cells of a connection have to pass through depends on the location of the input and output lines.

In unfolded networks, the inputs are located on one side and the outputs on the opposite side of the network. Internal links are unidirectional and all cells have to pass through the same number of switching elements.

Multiple-path networks can also be realized by using several planes of Banyan networks in parallel. [27]. This is called vertical stacking. All cells belonging to the same connection pass through the same plane. The plane and the path within the plane are determined during the connection setup phase. An incoming cell is switched to its appropriate plane by a distribution unit located at each input port. In [24], it is shown that even with two planes in parallel a virtually non-blocking switching structure can be achieved. Adding a number of stages of a Banyan network, one after the other, yields a horizontal stack.

3.9 Cell Header Processing in Switching Fabrics

Switching fabrics need to perform VPI / VCI translation and route the cells to the correct output port. In order to fulfil these tasks, two approaches can be used

- Self-routing principle
- Table-controlled routing principle

Studies have been made to decide which principle is superior [28]. For large multi-stage switching networks the self-routing principle is preferred because it is superior in terms of control complexity and failure behavior. The need for a higher internal bit rate because of the cell extension in self-routing networks is not critical.

3.9.1 Self Routing Switching Elements

When using self-routing elements, VPI/VCI translation only has to be performed at the input port of the switching network. After the translation, a switching network internal header is added to the cell. This header precedes the cell header. Because of the cell header extension an increased internal network speed has to be maintained.

In a network with k stages, the internal header is subdivided into *k* sub-fields. Sub-field *i,* contains the destination output port number of the switching element in stage *i* of the switching fabric. Figure 3-9 shows the cell header processing in a switching network built up of self-routing switching elements.

Figure 3-10 shows a generic block diagram of a self-routing switching element [29]. It contains a central memory with logical output queues and is controlled by the routing information included in the internal cell header. In order to keep the buffer access speed within a realistic range a wide parallel memory interface is used requiring serial-to-parallel conversion at the input and parallel-to-serial conversion at the output.

3.9.2 Table Controlled Switching Elements

In table controlled switching elements, the VPI / VCI of the cell header is translated into a new identifier in each switching element. Therefore, the cell length does not need to be altered. Figure 3-11 shows the header processing in a switching network which consists of table controlled switching elements. The contents of the table are updated during the connection setup phase. Each table entry consists of the new VPI / VCI and the number of the appropriate output link.

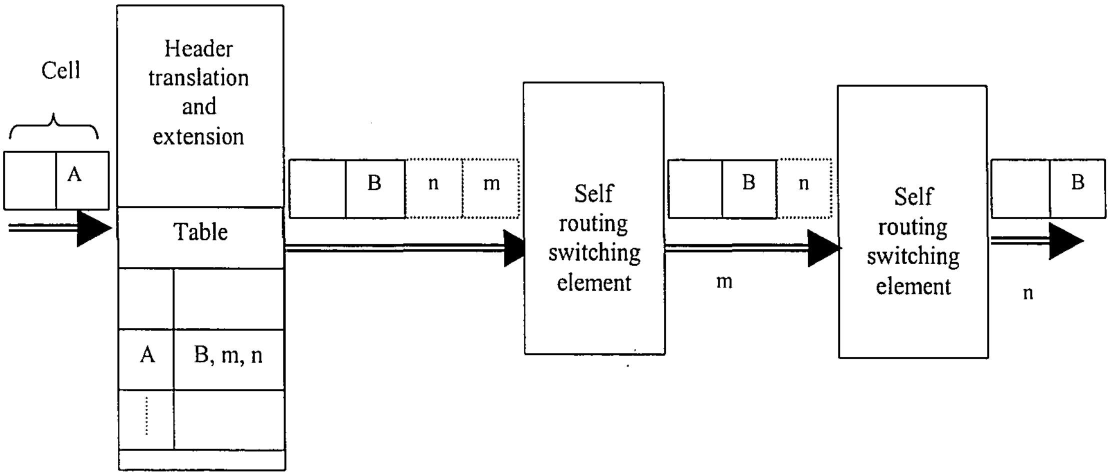

Cell
A
Header translation and extension
Table
A
B, m, n
B
n
m
Self routing switching element
B
n
m
Self routing switching element
B
n

Figure 3-9: Self-routing switching elements

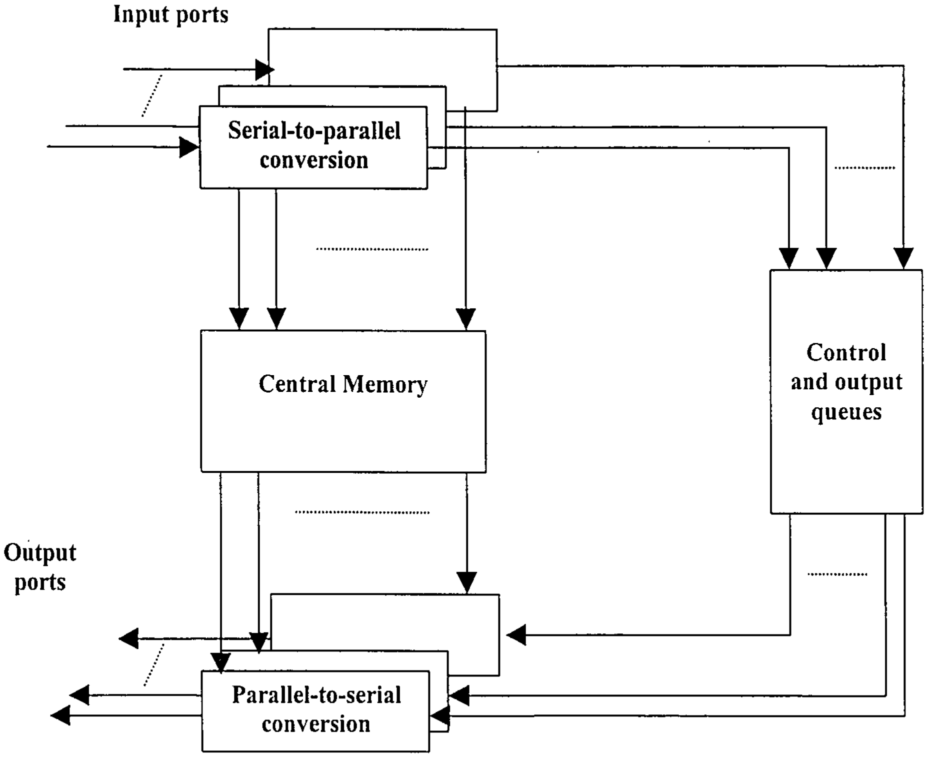

Input ports
Serial-to-parallel conversion
Central Memory
Control and output queues
Output ports
Parallel-to-serial conversion

Figure 3-10: Block diagram of a self-routing central memory switching element

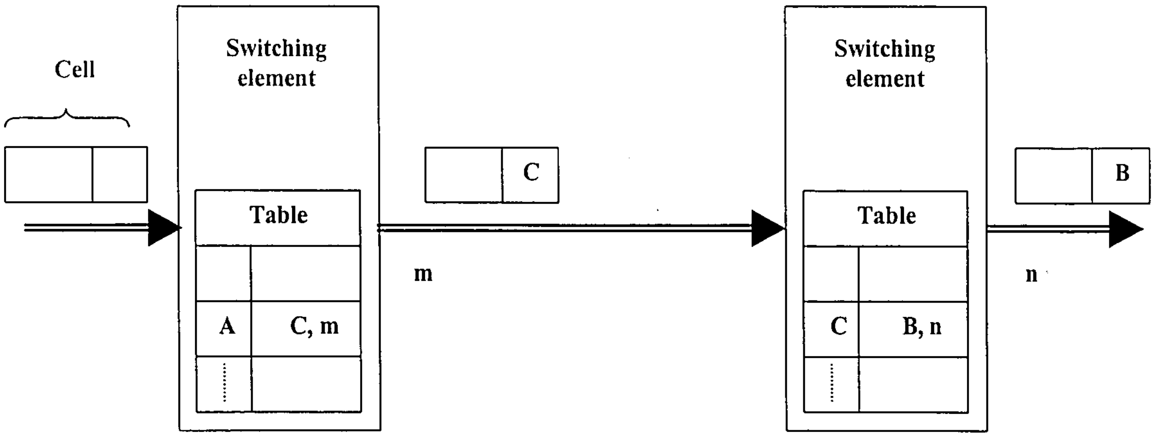


Figure 3-11: Table controlled switching elements

4. ATM Switch Simulation Results

This chapter describes the simulators developed for the three different categories of switching elements. Simulators described are for a regular input queued switching element, an output queued switching element (i.e. Knockout switching element [30], [31] [32]), and a central queued switching element (i.e. Coprin switching element [33]).

4.1 A Generic Input Queued Switching Element

Figure 4-1 shows a block diagram of an input queued switching element. The input buffers of an input queued switching element do not need to have very fast access times compared to an output or central buffered switching element. This is because the input buffers need to be written and read only once during one timeslot. However, all input queued switches have a major disadvantage called head of the line (HOL) blocking, explained in the previous chapter.

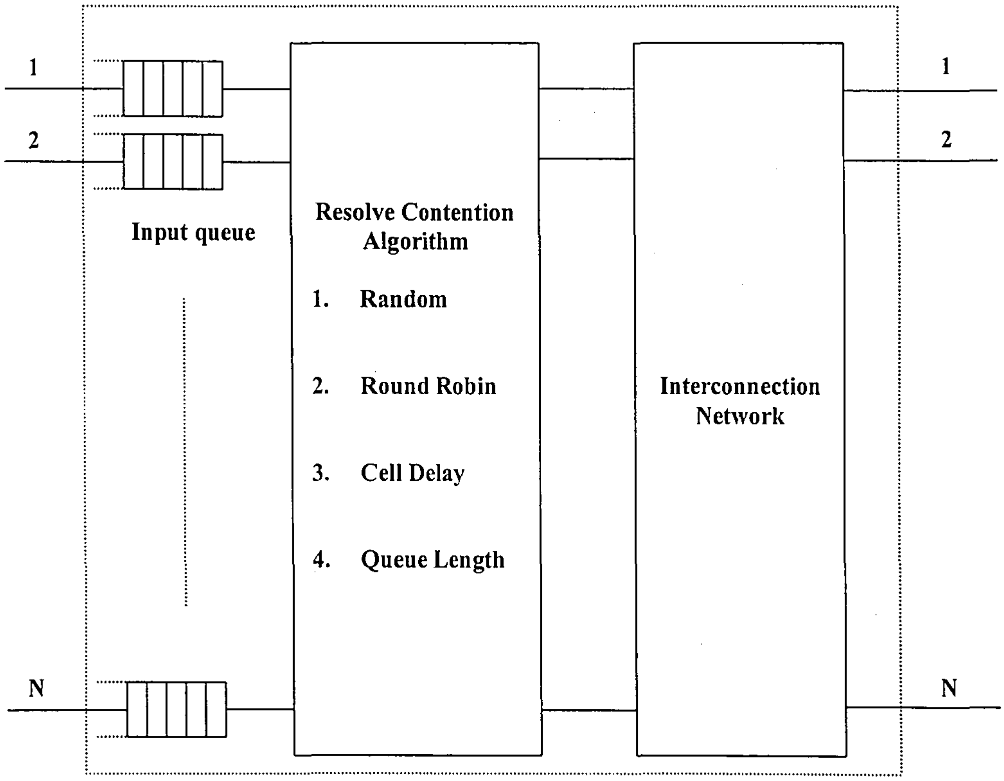


Figure 4-1: Input queued switching element simulator schematic

The input queued switching element has a generic architecture. More details about it can be found in the previous chapter and [18].

4.2 Output Queued Switching Element (Knockout Switching Element)

A Knockout switch is an example of an output queued switching element and is describe in detail in [30] [31] [34]. In a knockout switching element, the access time for an output memory has to be very fast, because for a N X N output queued switch it is possible for all the cells from the input port to be destined to the same output port. Therefore, the memory should be able to support N writes and 1 read (the cell to read out of the queue and sent out on the output link). In order to decrease the operating speed and complexity, the knockout switch, has more than one-output buffers at each output port. Figure 4-2 shows the schematic of a knockout switch and Figure 4-3 shows the details of a bus interface unit of a knockout switch.

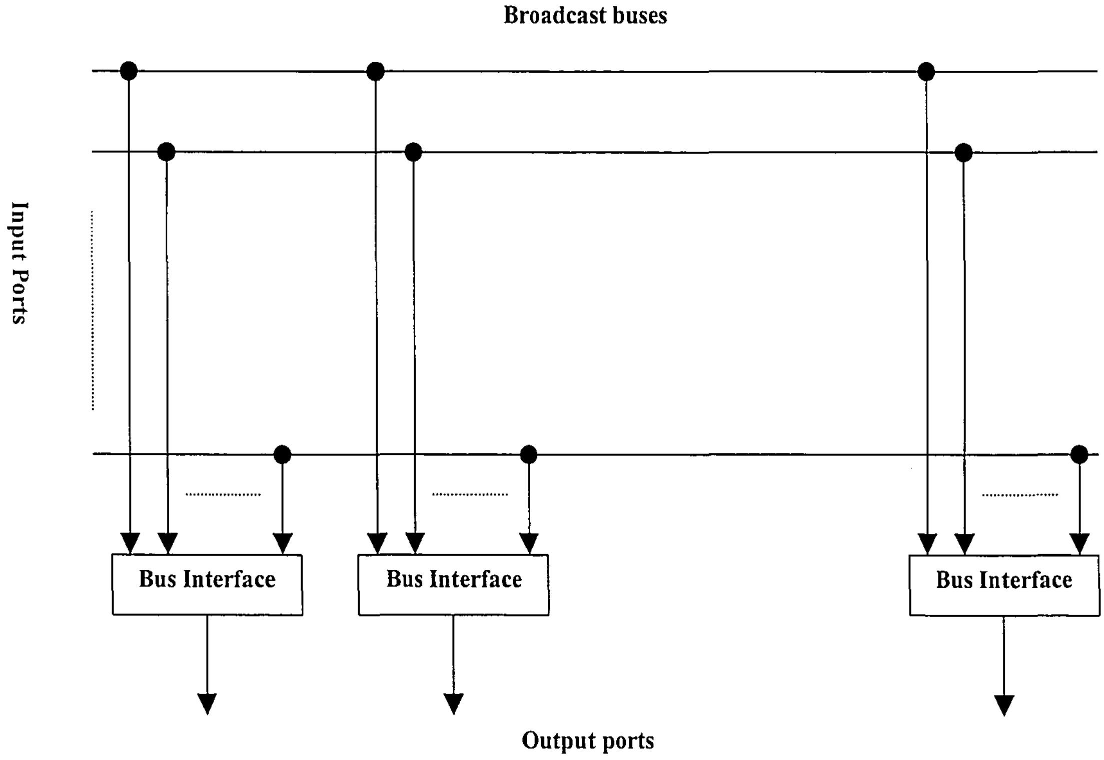


Figure 4-2: Knockout switching element simulator

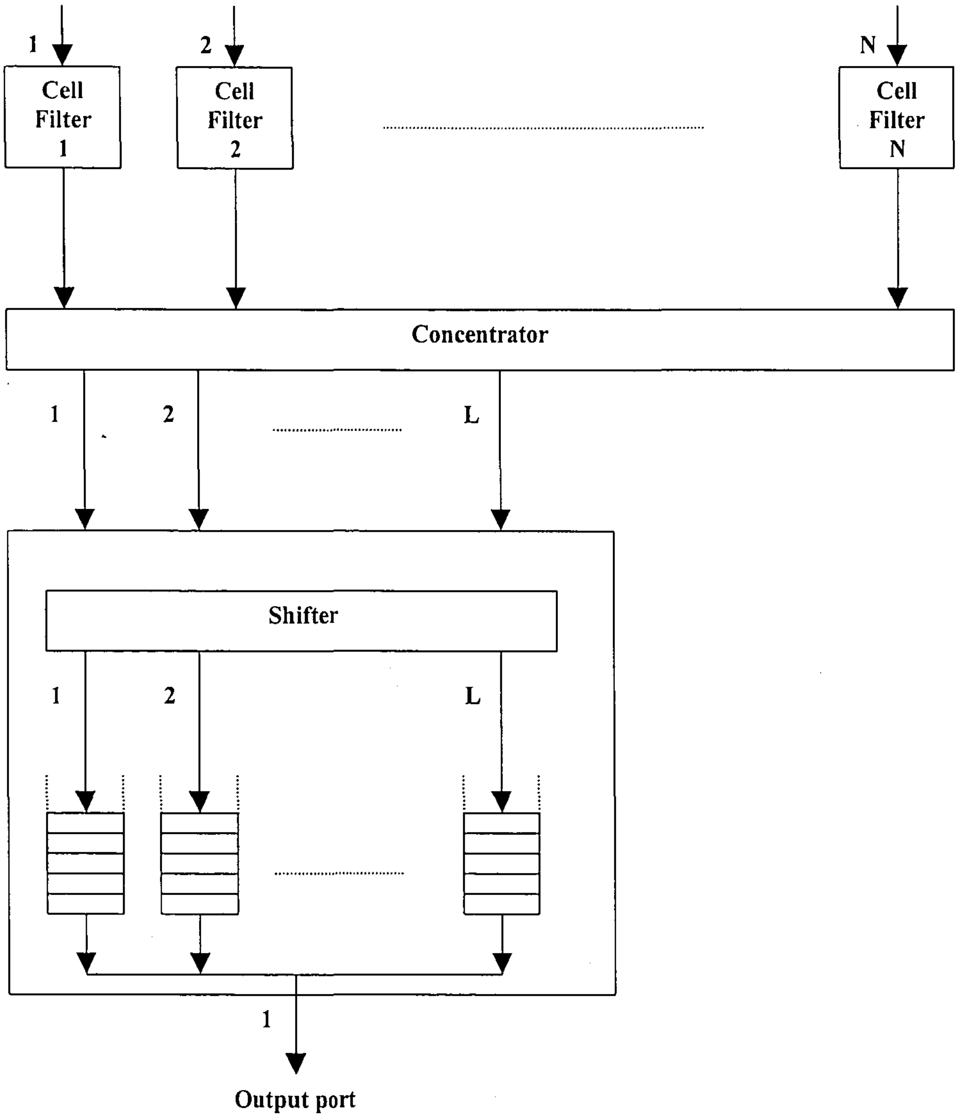


Figure 4-3: Knockout Bus Interface unit

Figure 4-2 shows a knockout switch with N input ports and N output ports, each operating at an equal speed. Fixed length ATM cells arrive on each input port in a regular time frame. The transfer medium is composed of N broadcast buses, one for each input port, each output port has access to cells arriving on all inputs, via a bus interface connected to each individual broadcast bus. As shown, each of the N input ports puts its cells on a separate broadcast bus, which is accessible to each of the output ports through a bus interface unit.

Because of this property the transfer medium is non-blocking and no cells are lost at the input ports of the knockout switch.

Cells may contend for a single output port in a bus interface of the Knockout switch. A major advantage of the bus structure is that each bus is driven by only one input port, allowing a simple implementation and high transmission rate compared with a bus shared by multiple input ports. If the bus has to be shared between input ports, timing has to maintained between all inputs.

In a knockout switch, since each the buses may have cells, not all of which are destined to the same output port, the bus interface units should only select cells destined to its output port. Cell filters perform the role of selecting cells. These cell filters examine the address of each incoming cell. If the cell is destined for that specific output port, the cell is passed to the concentrator, otherwise the cell is discarded. Each bus interface unit has N cell filters connected to one of the N broadcast buses on one side and a concentrator on the other side.

At a bus interface, several cells may arrive simultaneously. In the worst case, all N cells are destined for a single output port. Thus, the bus interface requires cell buffers somewhere. If zero cell loss has to be guaranteed in the transfer phase to the cell buffer, the memory must write at N times the speed of each input port. In a knockout switch this operating speed is reduced, by an intelligent bus interface, which acts as a concentrator, with a non-zero cell loss probability. The next part of the bus interface is the concentrator from N inputs to L intermediate outputs (L <= N). If K cells arrive simultaneously all destined for the same output port, then after passing through the concentrator, these K cells will arrive on inputs 1 to K of the concentrator, if K <= L. If K > L, then all L outputs of the concentrator will have cells, and K - L cells will be lost in the concentrator.

In [30], it is shown that L is not very sensitive to input load or the number of input ports, but is mainly influenced by the required cell loss probability. Hence, if L is selected to be 12, a cell loss probability of 10^{-10} can be achieved for any load and any value of N. This cell loss probability is only for the concentration stage of the knockout switching element. To get the complete cell loss probability of the switching element, the cell loss probability of the output queue also needs to be taken into account. Of the N input lines into the concentrator, it randomly selects L of the input lines with cells and transmits the cells on in its first L output lines. If there are cells on less than L of the input lines of the concentrator all the cells get select.

A concentrator is constructed using one or more 2 by 2 contention switches to form a tournament, with N players (inputs) and L winners (outputs). An L output tournament is constructed by having L trees, with a 2 by 2 switch used to construct each node of the L trees. The first tree has N leaf nodes, the cell that makes it through the root of this tree is sent out on the concentrator's first output line. The N - 1 losing cells are passed as inputs to a second tree with N - 1 leaves, the winner on this tree forms the second output of the concentrator and so on. Hence for a concentrator with N inputs and L outputs, there are L sections of competition, one for each output. A cell entering the concentrator is given L opportunities to exit successfully through a concentrator output. A cell losing L times is knocked out of the competition and is discarded by the concentrator. However, in all cases, cells are only lost if more than L arrive in a single cell timeslot. The knockout switch gets its name from these knockout tournaments.

The next part of the bus interface is the cell buffer responsible for storing the cells successfully passed by the concentrator, but which cannot be served simultaneously by the single output port. Since the buffers are located at the output of each bus interface, the option taken in the knockout switch for the cell buffer is the output queuing solution, the number of memory accesses during one cell time equals N+1 for a single ported memory.

In the knockout switch this number is reduced to L+1, since the concentrator has reduced the number of inputs from N to L. The implementation of this output queue is realized by L separate queues each of which need only two memory accesses (i.e. one for reading and one for writing) per cell time, instead of L+1. However, in order to distribute the load over all queues and to obtain the same cell loss performance as an output queue, the L separate queues must be shared, and operate virtually as a single queue. To achieve this, a shifter stage is provided which guarantees that all buffers are equally loaded and optimally used, but also that the cell sequence is guaranteed. This shifter is required since without it all P "winning" cells from the concentrator would arrive at the left queues (if P < L).

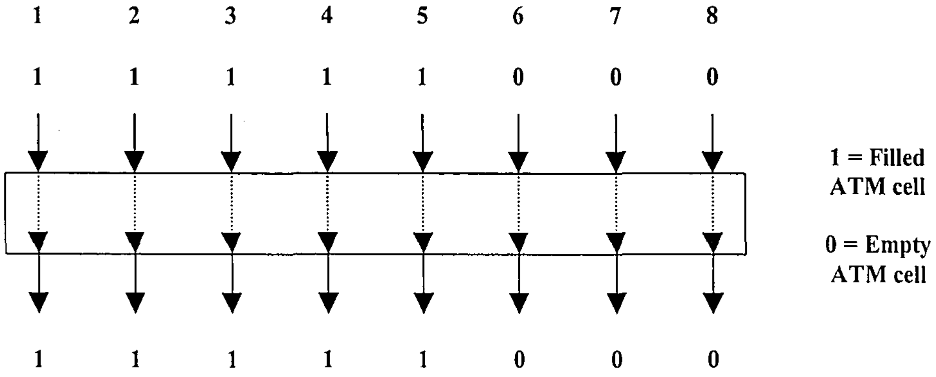


Figure 4-4: Eight input line shifter unit, first timeslot

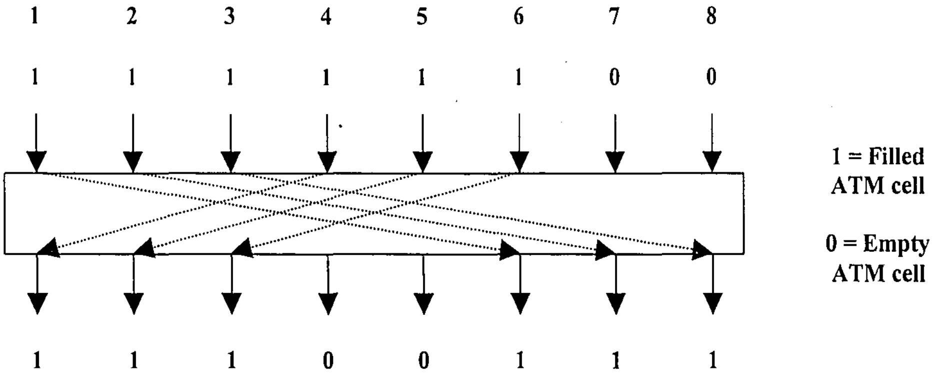


Figure 4-5: Eight input line shifter unit, second timeslot

The shifter provides a circular shift of the L inputs to the L outputs so that the L buffers are filled in acyclic way. Figure 4-4 and 4-5 shows the first and the second cell timeslots for an eight input line shifter. In the example, at the first cell time, five filled cells arrive at the first five inlets of the shifter. During this cell time the five cells are directly shifted through to the first five output lines. During the next cell time, six filled cells arrive, again at the first six inlets of the shifter. This is because the concentrator always concentrates on its first outputs. During the second cell time, the first cell will be shifted to output port six, the second to output port seven, etc. During the next cell time, the first cell will be shifted to the fourth buffer. With this technique, all cells buffers are filled uniformly.

A shifter can guarantee cell sequence, by cyclically reading out buffers. Formally, the shifter is described as follows. If S_i describes the number of positions the shifter has to shift to the right during cell time i, then

$$S_{i+1} = (S_i + K_i) \bmod L$$

Where, K_i represents the number of filled cells arrived during cell time i and S_1 is zero.

Total buffer memory of the output switching element = # concentrator outputs * # output ports * # cell locations

4.3 Central Queued Switching Element (Coprin Switching Element)

A Coprin switching element is an example of a central queued switching element and is described in detail in [33] [35]. The Coprin switch was originally designed for an experimental set-up called Prelude, to which data, voice and video sources were connected. It was designed and realized by the French Center National d'Etudes des Telecommunications (French CNET).

Like all other ATM switches, routing in the Coprin switching element is based on the header of the cell. Each ATM cell header contains a VPI, which is pre-set for each connection. The VPI is used in the Coprin switch to determine the physical output port; at the same time it is translated to an appropriate value for the outgoing link.

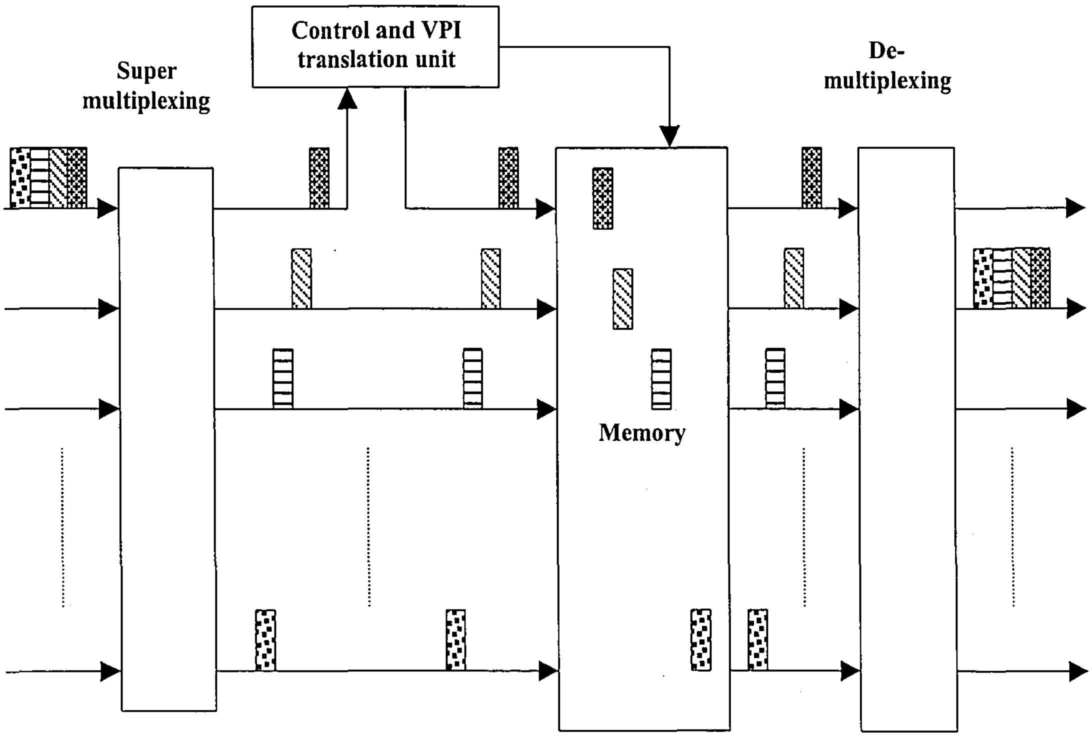


Figure 4-6: Coprin switching element

Figure 4-6 shows the block diagram of the Coprin switch element. Besides the standard functions like cell synchronization and header error control to be performed by each input port, the switch consists mainly of four parts. They are the super multiplexing, buffer memory, de-multiplexing, and control blocks. The Coprin switching element treats each stream as a parallel stream of information. The cell headers are all handled sequentially by a central control entity.

The functions of the four blocks are as follows:

- The super multiplexing block is responsible for converting the incoming serial cell stream to parallel data streams with all headers multiplexed into a special stream. Each parallel block has a size equal to the size of the cell header. Hence, for ATM cells with a payload size of forty-eight bytes and a header size of five bytes, an incoming cell is broken down into ten payload blocks and one header block.
- The de-multiplexing block reconstructs ATM cells, an action opposite to that performed by the super multiplexing block.
- The buffer memory stores the ATM cells, but since it receives the cells in a parallel format, it stores them internally in the parallel form.
- The control block manages the free and occupied locations of the queues of the buffer memory and performs header translation.

In the classical time division switching systems, different channels are put on a time division multiplexing (TDM) bus, which means that successive slots will contain information of different channels. The TDM bus is typically implemented via parallel wires, in order to reduce the physical speed of the wires.

The super multiplexing block performs a function that is comparable to TDM parallel bus multiplexing. Each physical wire of the bus, however, has a different meaning.

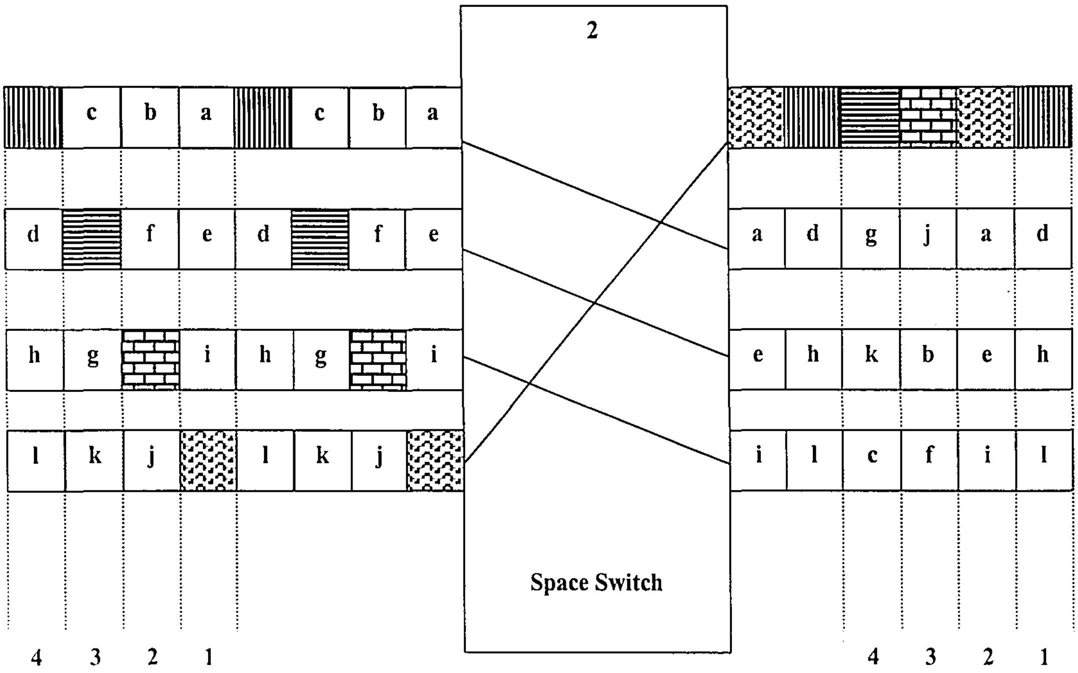


Figure 4-7: Stage 2 for a super multiplexing block in a Coprin switch element

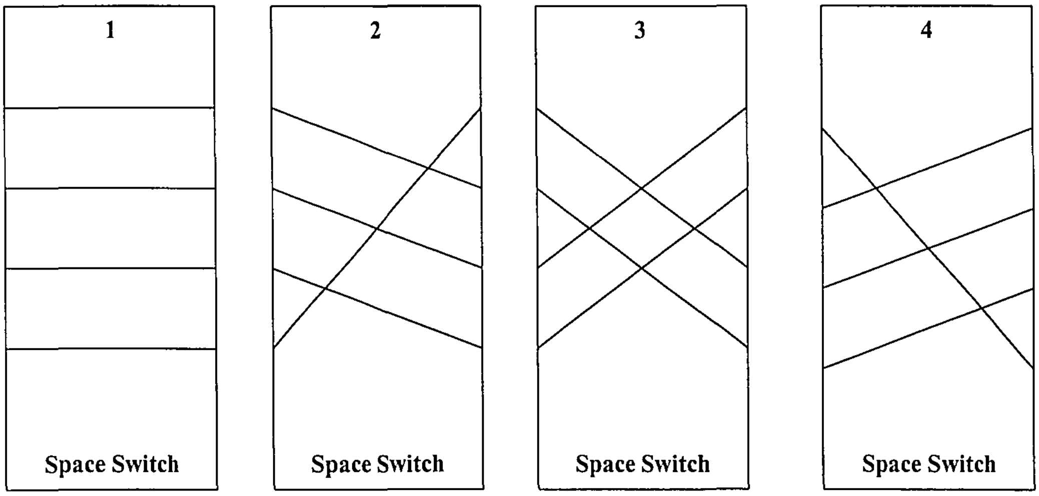


Figure 4-8: Four stages of the space switch

The first wire transports the first set of information bytes of all cells, the second wire transports the second set of information bytes, etc. The super multiplexing function is shown in Figure 4-7, for a four by four super multiplexing block, assuming a cell having a header of one byte and three bytes of data (a, b, c). A space switch of four by four implements this function by changing its state every byte. Figure 4-8, shows the four states of the space switch. In state one, input port i is connected to output port i; in state 2, input port i is connected to output port (i+1) mod 4; in state 3, input port i is connected to output port (i+2) mod 4 and in state 4, input port i is connected to output port (i+3) mod 4. For an N by N space switch in state k, input port i is connected to output port (i+k-1) mod N.

From the above paragraph it is clear that the size of the switch is related to the ratio of the cell information size and cell header size. From Figure 4-7, it can be seen that all header bytes are transferred to the first output port, all second-bytes (i.e. the first byte of the information field) to the second output port, etc. A prerequisite for the correct operation of the super multiplexing block is the alignment of the incoming cells at the input ports to the super multiplexing block, such that the headers on different input ports arrive at consecutive time slots.

The output of the super multiplexing block is forwarded to the buffer memory, the second part of the switch. The buffer memory, is a central queuing system. The central queue is organized and divided into banks, each bank being dedicated to a byte of the cell. This means that if byte one (i.e. the header) of a cell is stored in memory location A of memory bank one, byte two of the same cell (i.e. the first information byte) is stored in memory location A+1 of memory bank two, etc. Figure 4-9 shows an example buffer memory with four memory banks.

The third part of the Coprin switch element is a de-multiplexing block. If the control block of the switch sends a command to read address A, it means that address A is to be read from bank one, address A+1 from bank two, and so on. The reads are performed consecutively to guarantee that byte one of the cell is available on the physical output port one byte time before byte two of the same cell becomes available on the physical output port two, etc. The cells are reconstructed by a space switch which changes its state every other byte but in the reverse order of the super multiplexing block, first state 4, then state 3, then state 2 and finally state 1.

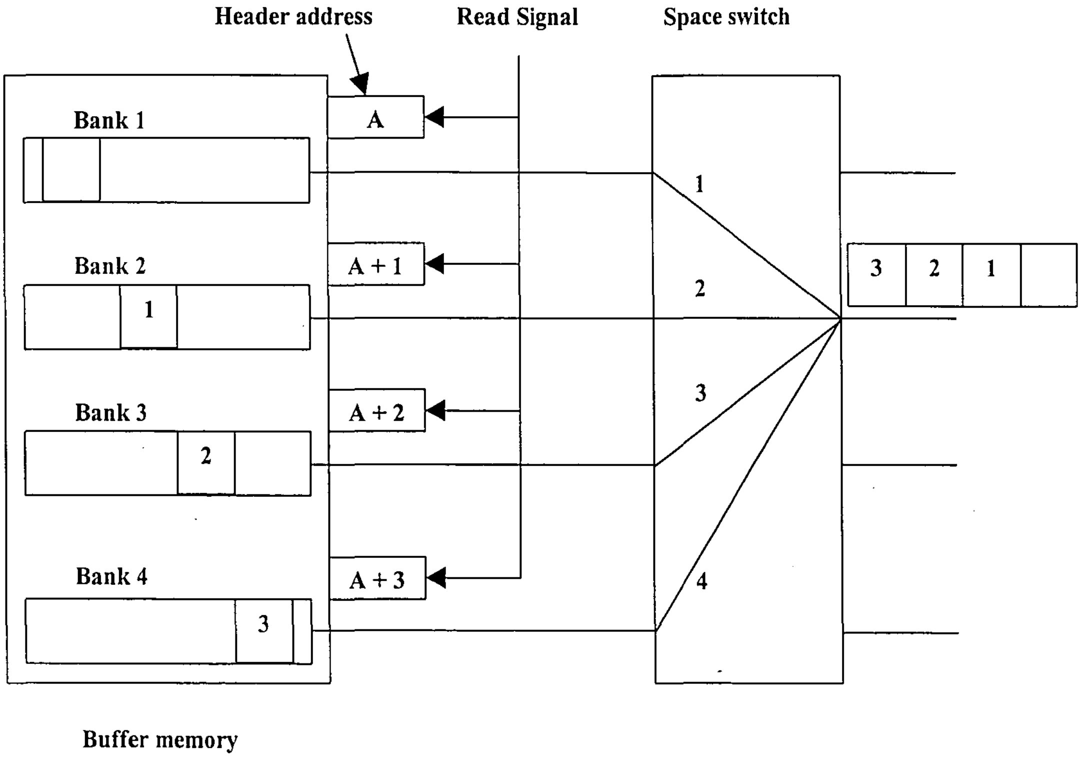


Figure 4-9: Buffer memory and de-multiplexing

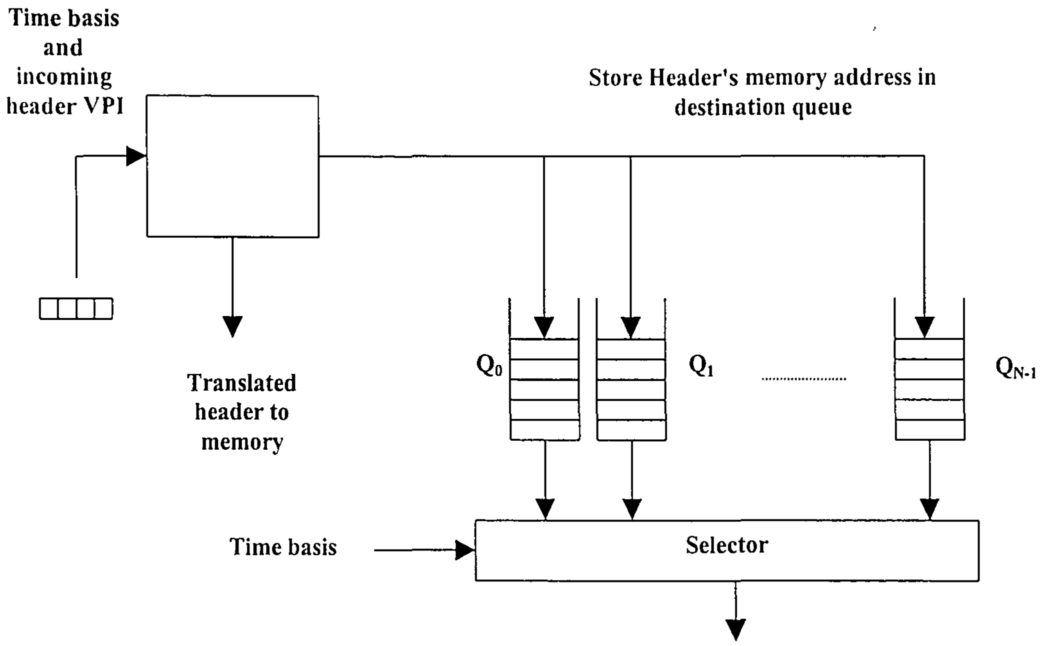

Time basis and incoming header VPI
Store Header's memory address in destination queue
Translated header to memory
Q_0
Q_1
Q_{N-1}
Time basis
Selector

Figure 4-10: Control block of a Coprin switch

The fourth part of the Coprin switch is the control of the memory buffers and the translation tables, as shown in Figure 4-10. The translation tables are addressed based on the time basis and the value of the incoming header. The time basis indicates the incoming port of the cell header. Using this information, the translation tables determine the outgoing port number and the outgoing VPI for the cell. In a Coprin switching element, if a cell's header is stored at address location i of the first memory bank, then address i is stored in the logical queue for output port i. When sending out cells from the output port k, the first address stored in the logical queue for port k is read.

4.4 Switching Elements Simulator Designs

Each switching element simulator is written in C++. It has four main components; a cell generator, routing table, switching element, and statistics collection component. Each switching element class is constructed to simulate a specific switching element it represents and hence differs in each implementation. Figure 4-11 shows the design and interaction between the classes the simulators.

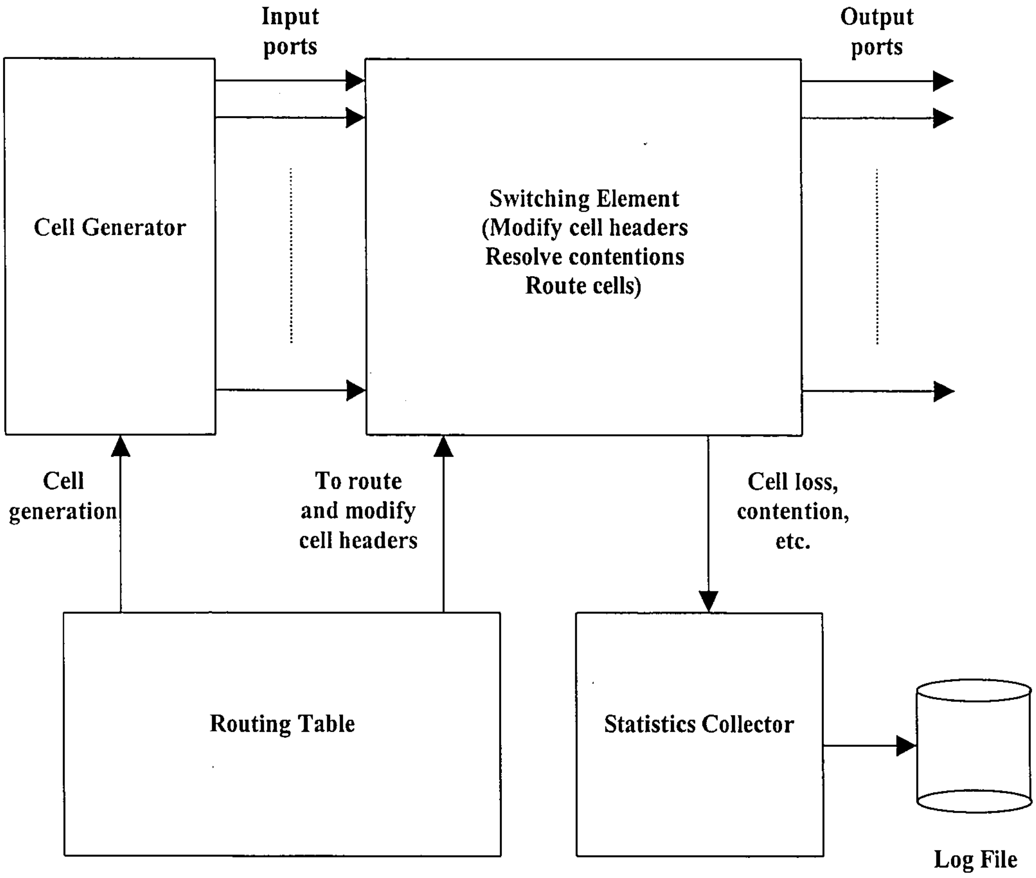

Input ports
Output ports
Cell Generator
Switching Element (Modify cell headers Resolve contentions Route cells)
Cell generation
To route and modify cell headers
Cell loss, contention, etc.
Routing Table
Statistics Collector
Log File

Figure 4-11: Simulator design and class interaction diagram

4.4.1 Input Queued Switching Element

The cell generator generates cells and puts them on each input line. The cell generation is controlled by a configurable parameter called input load (ranging between zero - ten). To simulate the generation of the load value, a random value is generated and divided by ten. If the remainder of the division is less than or equal to the load value, a cell is generated for the specific input port. For each input port and virtual connection, the faction of the total load on the input line contributed by the virtual connection is stored. To set the VPI in the header of the newly generated cell another random number is generated and divided by ten. Depending on the range in which the remainder lies a VPI is selected and set in the cell header.

The cell generated is then stored in the input queue. If the input queue is full, the cell is dropped and the cell loss counter is incremented. The first cell if any, is selected from the head of each queue. The correct outgoing port and VPI of the selected cells is looked up in the routing table using the input port and incoming VPI of the cell. A check is made to see if there is more than one cells destined to the same output port. If not, all the cell headers are modified to hold the correct outgoing links VPI and routed through the interconnection network. If there is a contention between cells going out on the output port, the contention is resolved by some algorithm (either random, round robin, delay based, or queue length based) and a winning cell selected. The cell headers of the winning cells are modified to hold the correct outgoing links VPI and routed through the interconnection network.

The simulator records information such as the number of input and output ports, simulation time, queue length, total memory, queue arbitration algorithm, total cells lost, average cell delay, cell loss per port, and routing table used for simulation.

4.4.2 Output Queued Switching Element

The simulator for an output queued switching element, has the same design as the input queued switching element, with the exception that the switching element is replaced with a Knockout-switching element, which is described in section 4.2. The concentrator of the Knockout-switching element has the same number of input and outputs ports so that there is no cell loss because of concentration.

4.4.3 Central Queued Switching Element

The simulator for a central queued switching element, has the same design as the input queued switching element, with the exception that the switching element is replaced with a Coprin switching element, which is described in section 4.3.

4.5 Computer Simulation Results for the Switching Elements

Input queued, output queued and central queued switching elements were simulated. For each of the three types of switching elements, two sizes 2 X 2 and 8 X 8 were simulated. The same routing tables were used to simulate all three switching elements. This precaution was taken so that the results would not get skewed, as different routing table generate different levels of contention, cell loss, etc, within a switching element.

Input Port	Input VPI	Output Port	Output VPI	Cumulative Load	Individual load	Priority
1	1	1	2	6	6	High
1	2	2	3	10	4	High
2	3	2	4	6	6	High
2	4	1	5	10	4	High

Table 4-1: Routing table for a 2 X 2 switching element

Input Port	Input VPI	Output Port	Output VPI	Cumulative Load	Individual load	Priority
1	1	1	2	3	3	High
1	2	2	3	4	1	High
1	3	3	4	5	1	High
1	4	4	5	6	1	High
1	5	5	6	7	1	High
1	6	6	7	8	1	High
1	7	7	8	9	1	High
1	8	8	9	10	1	High
2	9	2	10	3	3	High
2	10	1	11	4	1	High
2	11	3	12	5	1	High
2	12	4	13	6	1	High
2	13	5	14	7	1	High
2	14	6	15	8	1	High
2	15	7	16	9	1	High
2	16	8	17	10	1	High
3	17	3	18	3	3	High
3	18	1	19	4	1	High
3	19	2	20	5	1	High
3	20	4	21	6	1	High
3	21	5	22	7	1	High
3	22	6	23	8	1	High
3	23	7	24	9	1	High
3	24	8	25	10	1	High
4	25	4	26	3	3	High
4	26	1	27	4	1	High
4	27	2	28	5	1	High
4	28	3	29	6	1	High
4	29	5	30	7	1	High
4	30	6	31	8	1	High
4	31	7	32	9	1	High
4	32	8	33	10	1	High
5	33	5	34	3	3	High
5	34	1	35	4	1	High
5	35	2	36	5	1	High
5	36	3	37	6	1	High
5	37	4	38	7	1	High
5	38	6	39	8	1	High
5	39	7	40	9	1	High
5	40	8	41	10	1	High

Table 4-2 a: Routing table for an 8 X 8 switching element (Input ports 1 - 5)

Input Port	Input VPI	Output Port	Output VPI	Cumulative Load	Individual load	Priority
6	41	6	42	3	3	High
6	42	1	43	4	1	High
6	43	2	44	5	1	High
6	44	3	45	6	1	High
6	45	4	46	7	1	High
6	46	5	47	8	1	High
6	47	7	48	9	1	High
6	48	8	49	10	1	High
7	49	7	50	3	3	High
7	50	1	51	4	1	High
7	51	2	52	5	1	High
7	52	3	53	6	1	High
7	53	4	54	7	1	High
7	54	5	55	8	1	High
7	55	6	56	9	1	High
7	56	8	57	10	1	High
8	57	8	58	3	3	High
8	58	1	59	4	1	High
8	59	2	60	5	1	High
8	60	3	61	6	1	High
8	61	4	62	7	1	High
8	62	5	63	8	1	High
8	63	6	64	9	1	High
8	64	7	65	10	1	High

Table 4-2 b: Routing table for an 8 X 8 switching element (Input ports 6 - 8)

The routing tables for the 2 X 2 and 8 X 8 switching elements are shown in table 4-1 and 4-2, respectively. The third field determines which port is to be selected for a cell arriving on a specific input port with a given VPI. Its header is modified with the value in the output VPI field and is routed to the output port. The individual load field represents what fraction of cells generated for the specific input port are for that connection. The cumulative load field represents the sum of the individual load fractions for the input port. When a cell generator generates a new cell, it determines what value the VPI should hold by generating a random number between 1-10. The number is compared with entries of the cumulative load field for the specific input port, the upper row between which the number lies is selected as the connection which the cell represents and its VPI is entered into the header of the generated cell. The last field is used to determine the priority of the connection.

The routing table is constructed so that each output port is subjected to the same amount of load. The results generated by using such a routing table give a balanced view of all the three different switching elements.

4.6 Results

Simulation results were collected for 2 by 2 and 8 by 8 switching elements for each of the queuing schemes. The simulation programs ran for 10^2 to 10^8 ATM cell timeslots. The input load was varied from 10% to 100% for each combination. The total memory in term of ATM cells (53 bytes), available for queuing was varied from 2 to 1000 cells. Four queue arbitration algorithms were simulated for the input queued switching elements.

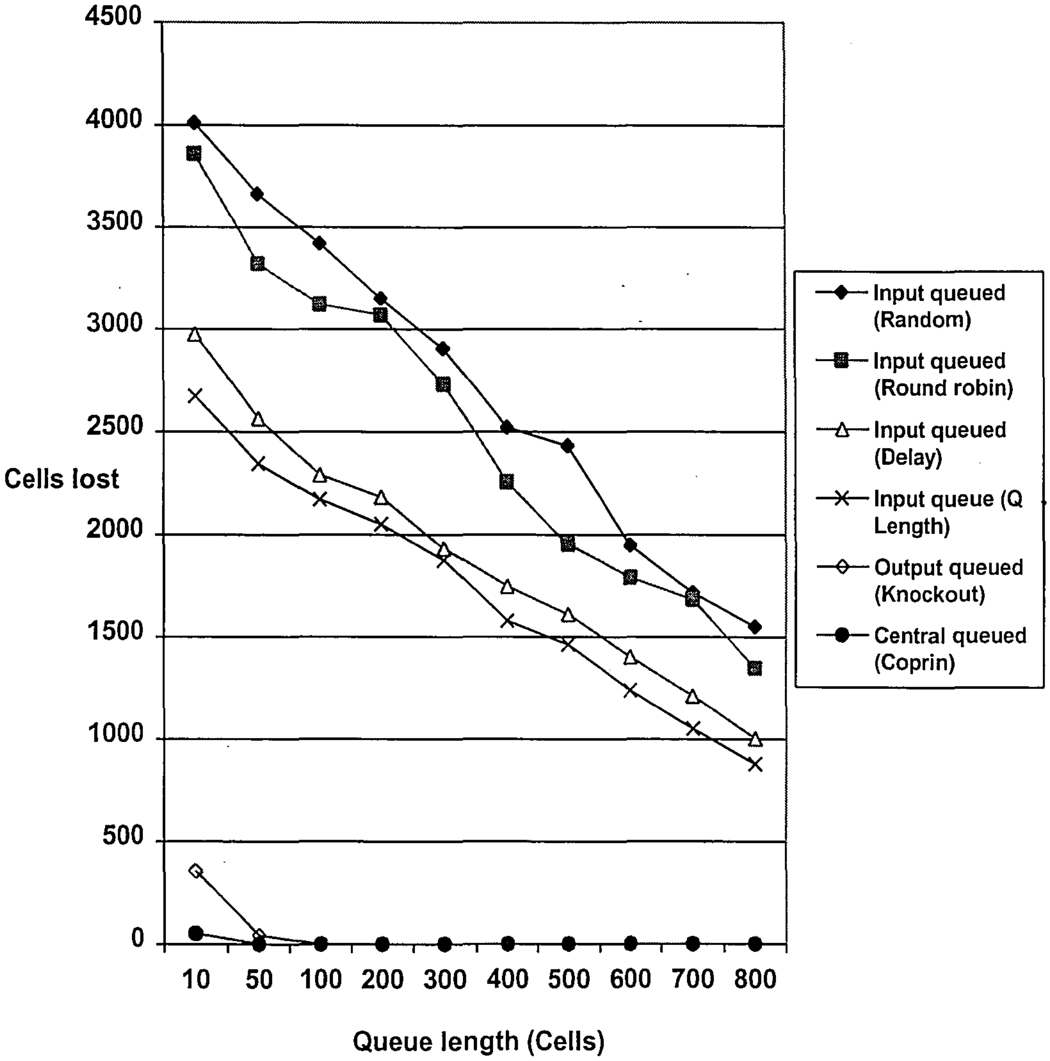


Figure 4-12: Cells lost vs. queue length for an 8 X 8 switching element with an input load of 90% and 10^8 ATM cell timeslots

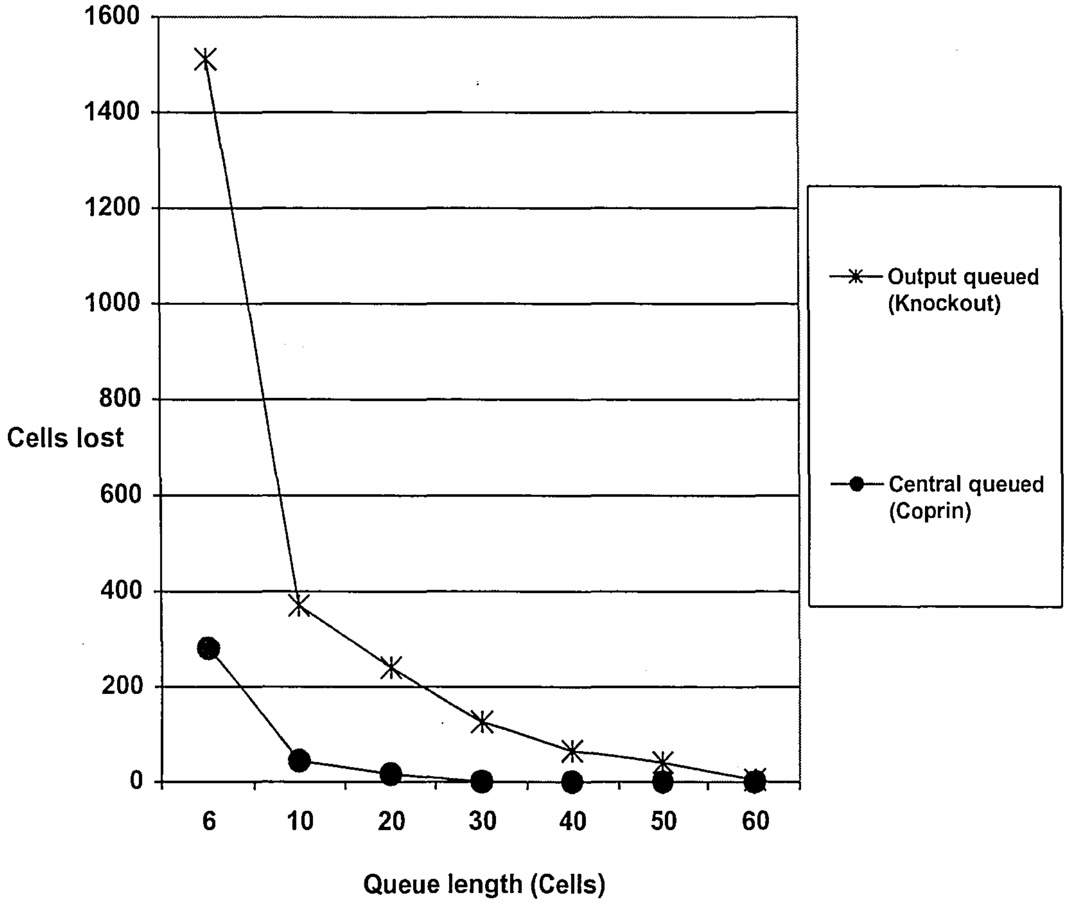


Figure 4-13: Cells lost vs. queue lengths for a 2 X 2 output and central queued switching elements with an input load of 90% and 10^8 ATM cell timeslots

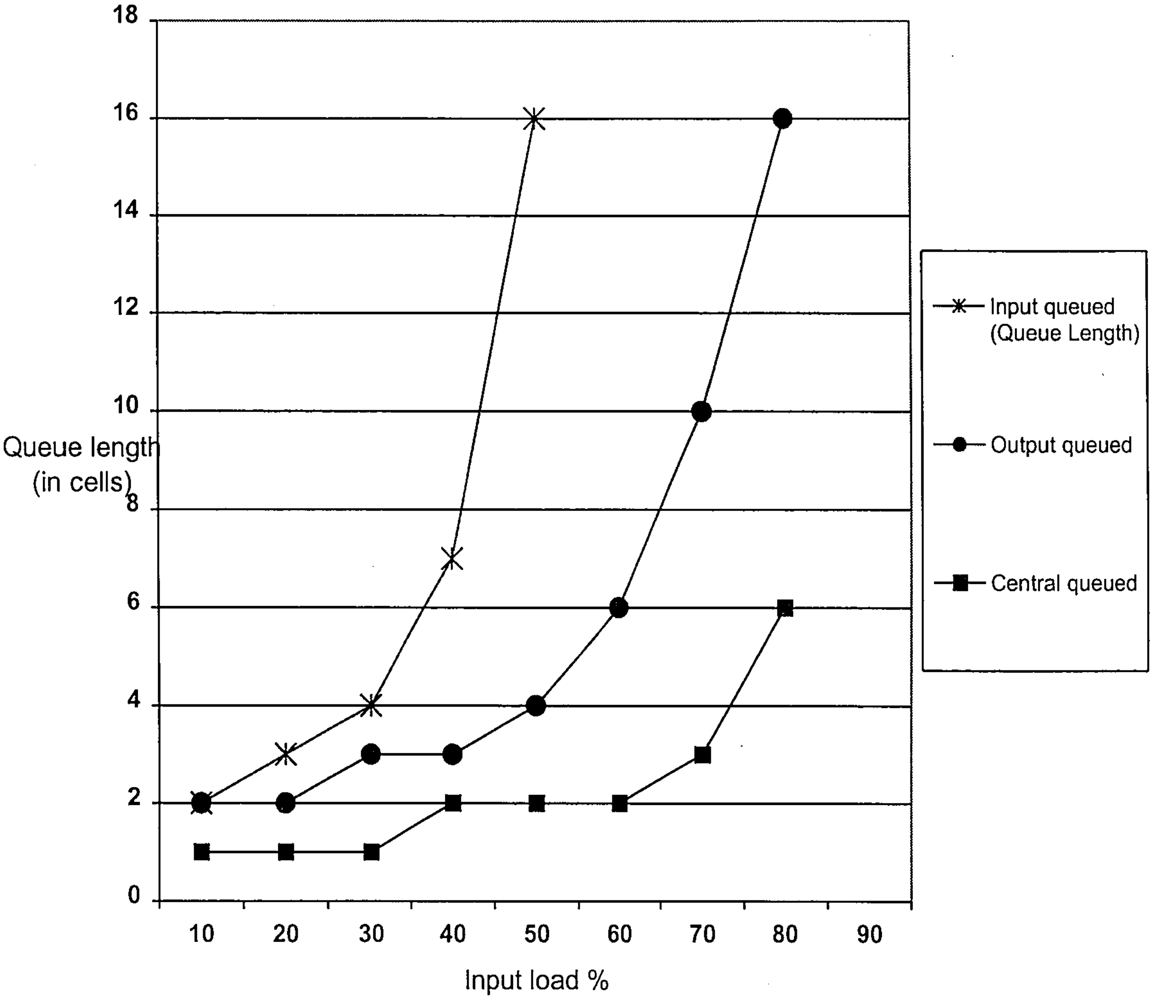


Figure 4-14: Queue length vs. input load for a 2 X 2 switching element, with a cell loss rate of 10^{-3}

4.7 Analysis of the results

Plots of some important statistics results generated by the computer simulation for the input, output and central queued switching elements are shown in Figures 4-12, 4-13, and 4-14.

4.7.1 Cells dropped vs queue length

The results shown in graph 4-12 are for an 8 by 8 switching element, with an input load of 90%. The simulation was carried out for 10^8 time slots. The central queued switching element (Coprin) performed the best (dropped the fewest number of cells for a given aggregate queue size), followed by the output queued (Knockout), and variants of the input queued switching elements. Between the input queued switching elements implementing different arbitration algorithms, the queue length algorithm performed the best followed by delay based, round robin and random selection algorithms. There is not a big performance difference between the central and output queuing schemes. There is a big performance difference, however, between the input queued scheme and other two schemes.

The reason why the input queued switch performed poorly and had more cell losses is attributed to the head of line (HOL) blocking phenomenon. HOL is explained in the previous chapter. The queue length arbitration scheme worked the best with minimum cell losses because it gives the highest routing priority to queues with a longer length and hence prevents queues from building up and overflowing. The delay dependent scheme faired better that the round robin scheme, because it gives the highest routing priority to the cell with the oldest time stamp, using run-time information as compared to a round robin scheme with does not take advantage of any run-time information. The random selection scheme faired the worst because it selected a queue randomly and there is a possibility that it may repeatedly select the same queue. As compared with the round robin scheme there is an equal selection probability for all queues. On the down side, the implementation of the queue length and delay based algorithms require much more logic than random and round robin arbitration algorithms.

In an output queued switching element, memory is wasted as compared to a central queued element. In a central queued element, all the memory is at one place and accessible to all the input and output ports. Whereas in an output queued switching element, the queue memory is distributed between the output ports. Distributing memory may result in a cell being dropped by an output queued switching element if the queue is full for the destination output port, even if there is a free slot in another output port's queue. In a central queuing scheme the cell would not be dropped under such a scenario as all the memory is accessible to all the ports.

4.7.2 Cells dropped vs queue lengths for central and output queuing

The results shown in graph 4-13 are for a 2 by 2 switching element, with an input load of 90%. The simulation was carried out for 10^8 time slots. The purpose of the graph is to clearly show the difference in performance between the central and output queuing schemes. The reason for the performance difference is the same as explained in the section above.

4.7.3 Queue length vs input load

Graph 4-14 displays the relationship between the required queue length and the input load so as to maintain a constant cell loss rate of 10^{-3}. Cell loss rate is computed by taking the ratio of the number of cells dropped to the number of cells received by the switching element at its input port. As in the earlier cases the input queuing performs the worst followed by output queuing and central queuing.

4.8 Queue Complexity

The three parameters influencing the complexity of the three different schemes are:

1. **Queue size**: The queue size determines the number of ATM cells that can be stored within a queue in case of congestion. Its size depends on the performance requirements of the system (cell loss ratio, load, delay) and the selected queuing principle.
2. **Memory speed**: The access time required of the queuing memory in the switching element depends on the queuing principle used, the switching elements dimension and the speed of the incoming and outgoing links of the switching element.
3. **Memory control**: To control the queue of a switching element, additional control logic is required. The complexity of this control logic depends on the queuing principle. For instance, the FIFO principle used in input and output queuing requires only simple control logic, whereas central queuing requires a dynamic memory management function.

These three different parameters make the choice of one of the three queuing systems very much dependent on the available chip technology and the speed at which the switch has to operate. To compare the three alternatives, a switching element with N input and output ports, memory width of W in bits and external bit rate of F is assumed. It is further assumed that the queue memory can be accessed in parallel. Table 4-3 shows a theoretical analysis of the three queuing schemes. In this example, the cell size is 53 bytes, the memory width W is 16 bits, the bit rate F is 150 MBPS, and the number of input and output ports N is 16.

	Input Queued	Output Queued	Central Queued
Single ported memory	W / (2 * F)	W / ((N + 1) * F)	W / (2 * N * F)
Example	53.3 ns	6.3 ns	3.8 ns
Dual ported memory	W / F	W / (N * F)	W / (N * F)
Example	106.6 ns	6.7 ns	6.7 ns

Table 4-3: Required memory access times for the three queuing schemes

4.8.1 Input queuing

For the input queuing scheme, in the worst case, the memory is accessed simultaneously at most only twice, once for writing by the input port and once for reading by the output port. These two-memory accesses have to be performed in one cell time. The access time is defined as the time between when a read is requested and when the desired word arrives.

The memory access time is W/(2*F) in case the of a single ported memory and W/F for dual ported memory (simultaneous read and write operations are possible). From Table 4-3, it is clear that the memory access time does not represent a major problem since, access times of 40 to 50 nanoseconds are achievable with commercial memory chips of 256 Megabits [36]. Using a dual ported memory slows down the memory access rate by half, and hence is attractive for input queuing systems.

The control logic of the input queue is simple as only a read and write pointer need to be maintained for each queue to implement the FIFO discipline. An arbitration algorithm needs to be implemented at the input queues to resolve contention if multiple cells at the heads of the queues are destined for the same output port. Input queued switching elements suffer from head of line blocking.

Input ports and output port queues are connected via a transfer medium. The transfer medium has to operate at a speed F, since the input queues solve the problem of contention by buffering the extra cells destined for the same output port.

4.8.2 Output queuing

In the worst case, the output queue can receive a cell from all N inputs simultaneously (N write operations), and one cell needs to be sent out over the output link (one read operation). All these (N+1) operations have to be performed during one cell time.

The memory access time of the output queue system equals W/((N+1)*F). If a dual ported memory is used, at most N write operations are performed during one cell time and the read operation is performed in parallel. The memory access time is then W / (N * F). This dual ported memory option only results in a moderate gain for the memory access time. On the other hand, dual ported memory typically requires one and a half times the chip surface of that of single ported memory. Therefore, dual ported memory is not very attractive for output queuing systems. As can be seen in Table 4.3, the access time of the memory in the example is rather small (6.3 nanoseconds) requiring on chip memory. The gain obtained by dual port memory in access time is not worthwhile given the increased complexity.

The control logic of the output queue is quite simple. It needs to only perform a pure FIFO operation, and thus requires a single read and write pointer to control the address selection of the queue memory during read and write operations.

Input ports and output port queues are connected via a transfer medium. If no cells are to be lost in the transfer medium, it must operate at a speed of (N * F) to ensure that all N cells are transferred during one cell time. To operate the transfer medium at such the high speeds, the design of the transfer medium is complicated. If a simple TDM bus is used, reflections become a major problem at these high speeds, especially if the switching element has to be implemented by several chips. However, if the bus length is kept very small (e.g. on chip) these reflection problems disappear, making the TDM bus an ideal solution for a transfer medium due to its low complexity.

4.8.3 Central Queue

In the worst case, the central queue can receive a cell from all N inputs simultaneously (N write operations), and N cells may needs to be sent out over the N output link (N read operations). All these (2*N) operations have to be performed during one cell time.

Hence, the memory access time is equal to W/ (2 * N * F), since each input and output port can access the memory simultaneously. For dual ported memory the access time is reduced to W / (N * F). Using a dual ported memory slows down the memory access rate by half, and hence is attractive to use for central queuing systems.

The transfer media at the input and output of the central memory must operate at (N*F), if it has to be non-blocking. The non-blocking property of a transfer medium, implies that the transfer medium does not lose any cells. If cell loss is acceptable at the transfer medium, a lower operating speed than (N*F) can be used.

The control of the central memory is the most complicated of all three queuing schemes. This is because all the cells are stored and mixed in the central memory. Cells destined for a particular output port are not stored contiguously. Dynamic memory allocation techniques are performed in hardware for speed. To achieve high speeds, the memory needs to be combined with the control of the queue on a single chip.

5. Routing in ATM Switching Fabrics

This chapter describes switching fabrics used to route packets in an ATM switch. The ideas discussed use 2 by 2 switching elements at the nodes of the fabrics, however, the concepts can easily be extended to include larger switching elements. First, the commonly used sorting-trap-butterfly network is discussed. Later, a stacked-banyan-balancing network is evaluated and the performance of the two networks is compared. A sorting circuit-banyan network can be constructed by placing a sorting circuit [37] in front of a banyan network [23].

5.1 Banyan and Delta Networks

Any network for which there is a unique path from each input to each output is called a banyan network. For instance, the $\log_2(N)$ dimensional butterfly is an example of a 2N-input and 2N-output banyan network. In order to facilitate packet routing, the k output edges of each switch in a banyan network are typically labeled 0, 1, …., k-1. Any path from an input to an output can then be specified by the sequence of edge labels traversed by the path. Figure 5-1 shows a two-dimensional butterfly network as an example of an eight-input, eight-output banyan network.

A banyan network, for which the label sequence of every path to the same output is the same, is known as a delta network. For example, the butterfly is a delta network for which the label sequence of each path matches the binary representation of the destination of the path. Delta networks are useful for packet routing applications since each switching decision made for a packet at a particular node depends on only a single bit of the destination address of the packet. If the endpoints of the edges of a banyan network G can be labeled so that both the network and its reverse are delta networks, then G is said to be a bidelta network. For instance, a butterfly network is also an example of a bidelta network.

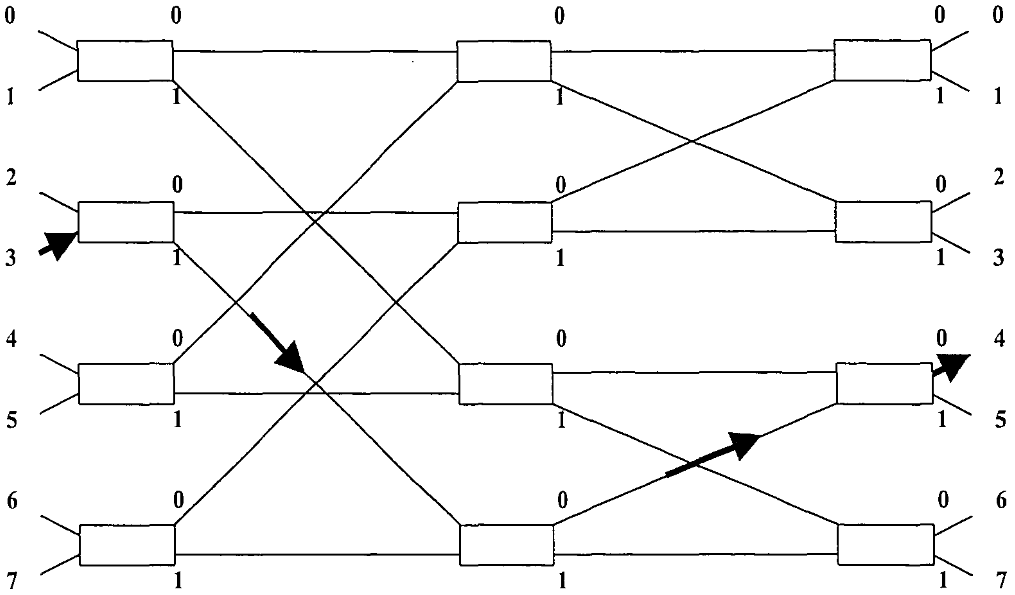


Figure 5-1: Butterfly: A banyan, delta and bidelta network

Each input is connected to each output by a unique path that traverses each level of the network precisely once. Each edge is labeled for the butterfly network. Every path to output i traverses $b_1, b_2, \ldots, b_{\log N}$, where $b_1\, b_2 \ldots b_{\log N}$ is the binary representation of output i. For example, the path from input three to output four follows the labels 1, 0, 0.

From the previous discussion, it can clearly be seen that to route a packet from an input to output the packet passes through exactly log N stages. Even if all the input packets are destined for a unique output, contention may still arise. It is possible for two packets destined to different outputs to contend for the same output link in the nodes of the butterfly. For example, if there are two packets. Packet A at input 2 has output 5 as its destination and packet B at input 3 has output 4 as its destination, then the two packets contend for the same out link of the first and second level nodes of the butterfly.

If all input packets are destined for different output ports, then there can be at most two packets contending for the same output link in any node of the butterfly. This contention problem can be resolved by adding a single cell buffer in each node of the butterfly and operating the internal links of the butterfly at twice the operating speed of the input and output links. Whenever there is contention, one of the cells can be stored in a buffer and the other sent. Since the link operates at twice the speed, it can send the buffered cell right after the first is successfully transmitted without dropping a cell.

5.2 Odd-Even Merge Sort

Odd-even merge sort is one of the oldest, and most widely used parallel sorting algorithm. Odd-even merge sort belongs to the class of divide and conquer algorithms. It works by recursively merging larger and larger sorted lists. Given an unsorted list of N items, the algorithm starts by partitioning the items into N sub-lists of length one. Next, it mergers these unit-length lists into N/2 sub-lists of length 2 and so on. At the end, two sorted lists of length N/2 are merged to yield a single sorted list of length N.

The key to odd-even merge sort is the way merging is done. To merge two sorted lists A and B, where A = $a_0, a_1, \ldots.., a_{M-1}$ and B = $b_0, b_1, \ldots.., b_{M-1}$ into a single list L (where M is a power of two), A and B are first partitioned odd and even index sub-lists.

Even(A) = $a_0, a_2, \ldots..., a_{M-2}$

Odd(A) = $a_1, a_3, \ldots...., a_{M-1}$

Even(B) = $b_0, b_2, \ldots..., b_{M-2}$

Odd(B) = $b_1, b_3, \ldots...., b_{M-1}$

As A and B are sorted, so are the even and odd index sub-lists.

Recursion is used to merge even (A) with odd (B) to form a sorted list C, and odd (A) with even (B) to form another sorted list D. To form L, lists C and D are merged. The advantage of the algorithm lies in the fact that the task of merging C and D is much easier than the task of merging A and B.

If $C = c_0, c_1, \ldots\ldots, c_{M-1}$ and $D = d_0, d_1, \ldots\ldots, d_{M-1}$, then lists C and D are merged in the following way to yield the sorted list L.

$L' = c_0, d_0, c_1, d_1, \ldots\ldots, c_{M-1}, d_{M-1}$

Then by comparing each c_i with d_i and switching the values if they are out of order the new list L is generated.

For example,

A = 2, 3, 4, 8 and B = 1, 5, 6, 7

Then even (A) = 2, 4, odd (B) = 5, 7, yielding C = 2, 4, 5, 7

even (B) = 1, 6, odd (A) = 3, 8, yielding D = 1, 3, 6, 8

L' = 2, 1, 4, 3, 5, 6, 7, 8

L = 1, 2, 3, 4, 5, 6, 7, 8

The proof for odd-even merge sort is given in [37].

5.3 Sorting Circuit of depth $\log_2(N) (\log_2(N) + 1) / 2$

The odd-even merge sort algorithm for N items described in the previous section can be implemented in hardware using a sorting circuit with a depth of $\log_2(N) (\mathrm{Log}_2(N) + 1) / 2$, described in [34]. A sorting circuit is a device consisting of registers and comparators. There is one register for each item being sorted. At each step, the registers are grouped together in pairs so that comparisons can be made. Each register is involved in at most one comparison during each step. The depth of the circuit is the number of steps in which comparisons can be made. Figure 5-2 shows a four input sorting circuit. The circles represent registers and the vertical lines represent comparators. After each comparison, the smaller of the compared items is placed in the uppermost of the compared registers.

An important point to be noted is that a merging circuit has the same interconnection as a butterfly network. A $\log_2(M)$ depth sorting circuit is similar to an M-input butterfly network. Hence, a sorting circuit is also known as butterfly comparator circuit. A sorting circuit is recursively constructed using merging circuits. To sort a list of N items $\log_2(N)$ merging circuits need to be used. The depth of the sorting circuit constructed is $\log_2(N) (1 + \log_2 (N)) / 2$. If N is a power of two then the depth of the last merging circuit is $\log_2(N)$, the second last merging circuit's depth is $\log_2(N/2)$, and so on.

Depth of the Sorting circuit = $\log_2(N) + \log_2 (N/2) + \dots\dots. + \log_2 (2)$

$= 1 + 2 + \dots. + \log_2 (N) = \log_2 (N) (1 + \log_2 (N)) / 2$

Figure 5-2 and 5-3 show four item and 16 item sorting circuits, respectively. From these figures it can be seen that both merging and sorting circuits can be recursively constructed. The sorting circuit requires the data to pass through $O (\log_2(N))^2$ stages or levels.

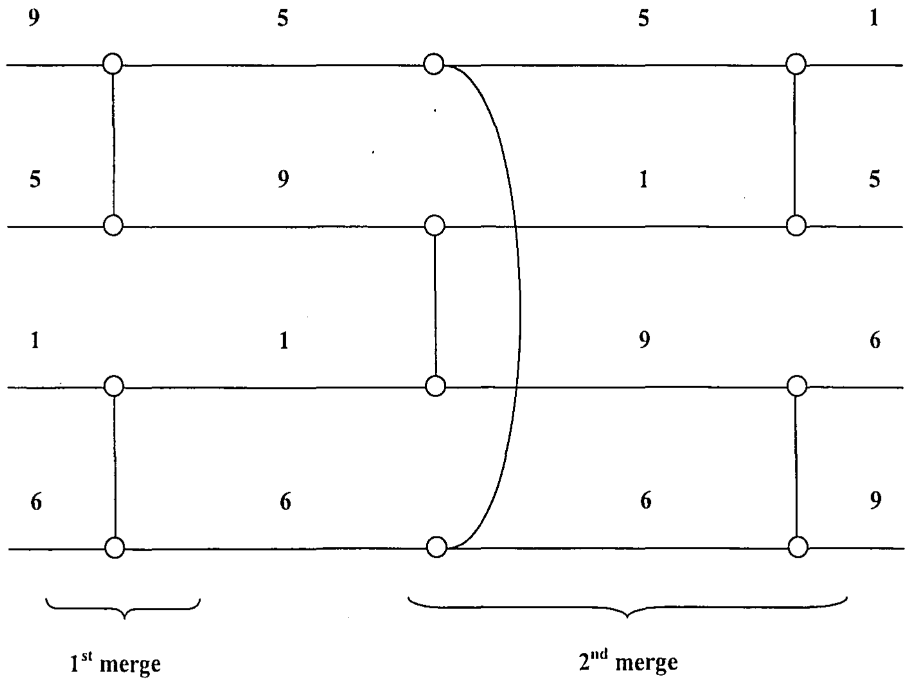


Figure 5-2: A four-input sorting circuit of depth three.

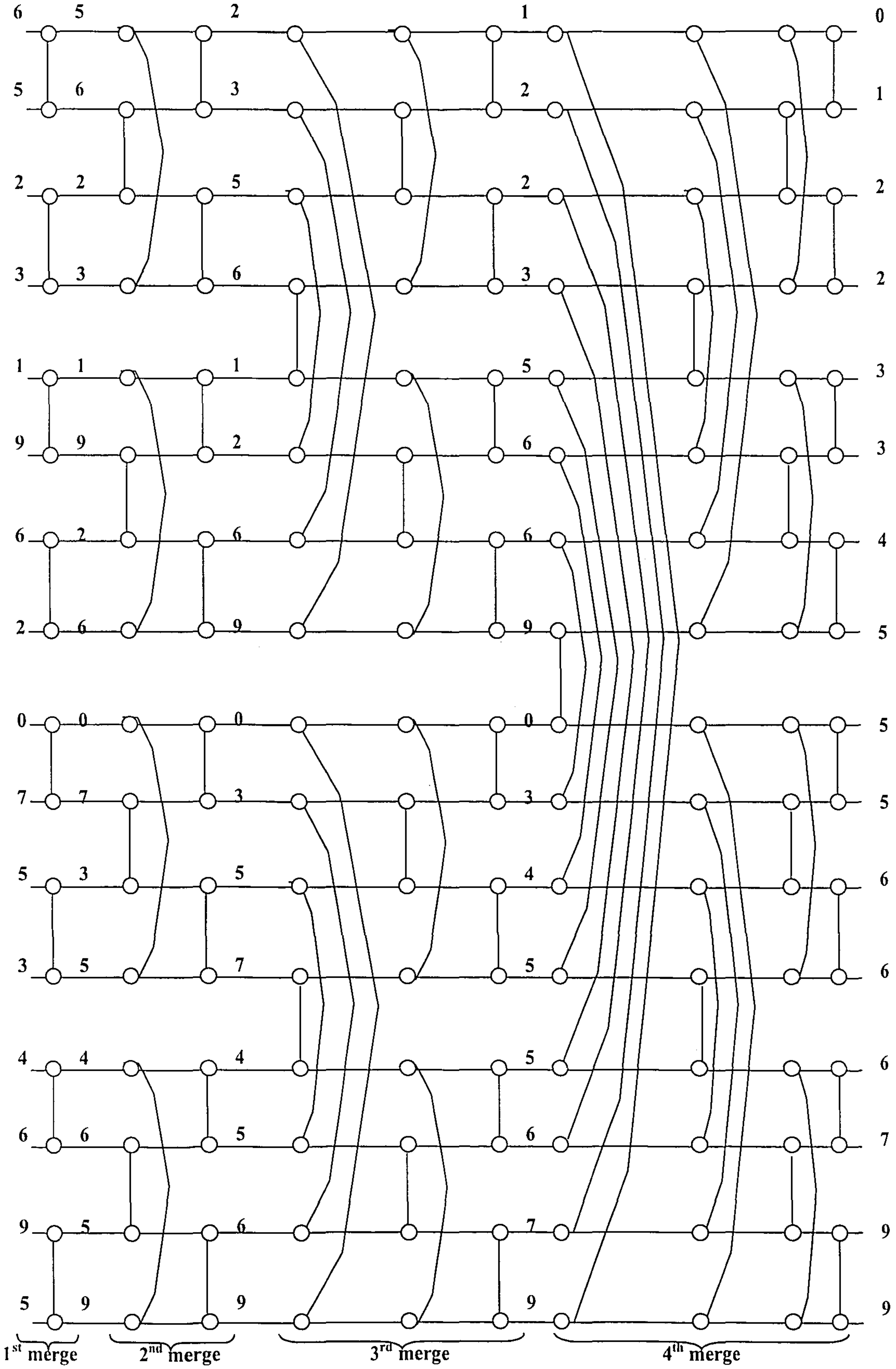


Figure 5-3: A sixteen item sorting circuit of depth ten

5.4 Reducing a Routing Problem to a Sorting Problem

Most of the fast deterministic algorithms for packet routing on hypercubes are based on sorting and monotone routing [37]. Monotone routing problems are problems for which the relative order of the packets at the inputs is preserved even at the outputs after routing. Routing N packets can be easily achieved in two steps. In the first step, the packets are sorted using a sorting circuit (depth $\log_2(N)\ (1 + \log_2(N))\ /\ 2$). At the output of the sorting circuit, the packets are arranged according to their destinations. In the next step, these packets are fed to a $\log_2(N)$ dimensional reversed butterfly network. If all the input packets are destined to unique destinations, then the butterfly network can route the packets without any two packets colliding at a node (no packets are lost) [37].

In the routing algorithm, a packet travels the cross edge of the butterfly at the i^{th} level only if the i^{th} bit of its origin column differs from i^{th} bit of the destination column. This simple algorithm ensures that there are no collisions provided all the input packets are destined to unique output ports. The proof is given in [37].

As it is quite likely for ATM cells from different input ports to be destined to the same output port, a trap network is added to the above routing circuit. The trap network is inserted between the sorting network and the butterfly network. The trap network detects cells with identical destination addresses and all but one of these cells is fed back to the input of the sorting network. Cells that have to pass through the sorting network again are assigned a higher priority in order to maintain cell sequence integrity.

The main disadvantage of this network is that buffers need to be maintained in the sorting network, to prevent cell loss. The number of stages through which the cells have to pass is equal to the sum of the stages in the sorting network and the butterfly, which is

$[\log_2(N)\ (1 + \log_2(N))/2] + \log_2(N)$. Figure 5-4 shows the block diagram of a sorting-trap-butterfly network.

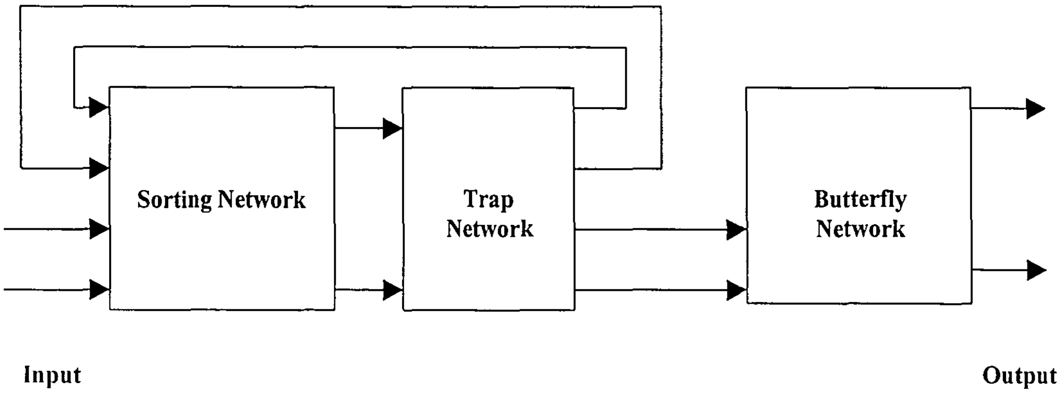


Figure 5-4: A sorting-trap-butterfly routing network

5.5 Solving the Routing Problem Using $\log_2(N)$ Stage Butterfly and a Balancing Network

The previously described banyan network works well if all the input packets are destined for different output ports. This, however, is not a realistic scenario in ATM switches. A way to avoid this problem without dropping any cells in an N input port switch is to have a vertical stack of N banyan networks between the input and output ports. A balancing network is inserted between the input ports and the banyan networks. During the connection setup phase the balancing network selects a banyan network. All cells of that connection pass through the same banyan network. If there are N banyan networks, no two cells destined to the same output port are present in the same banyan network. It is possible for more than one cells to reach the same output port from different planes, so buffers are needed at the output ports. The number of stages through which the cells have to pass is $\log_2(N)$. Improvement is gained at the expense of extra hardware. Since there is no cell loss in the switching fabric, it is can be classified a multi-path interconnection network with out internal cell loss. Figure 5-5 shows the block diagram of a stacked banyan-balancing network.

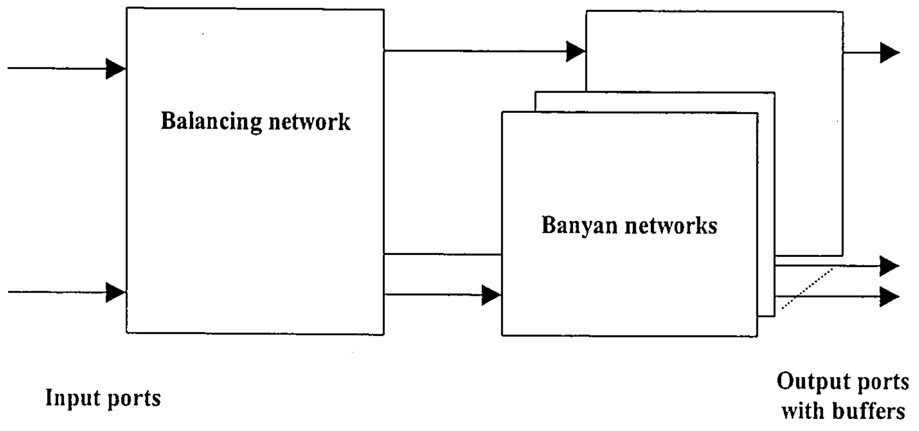


Figure 5-5: Stacked-banyan-balancing network

6. Conclusions

Currently, output queued switching elements appear to be the method of choice when designing new ATM switch architectures because of the following reasons:

1. The performance of input queued switching elements is satisfactory only under low input loads (below 40%). Under higher loads the cell loss rate of the input queued switching element becomes unacceptable.
2. For central queued switching elements, all the cells are stored in a central memory. Dynamic memory allocation and control algorithms need to be implemented in hardware on the same chip as the memory for speed. This means more area on the memory chip needs to be set aside for the controller logic. Also, dynamic random access memory (DRAM) access times improve by twenty-two percent every three years, whereas DRAM capacity increases four-fold every three years [36].
3. From the simulation results of an 8 by 8 output switching element, to maintain a cell loss probability under 10^{-8} and for a given input load of 90%, the total amount of queuing memory is 5300 bytes. Under similar conditions, the memory requirements of a central queued switching element is 3180 bytes. Output queued switching elements need more memory to maintain the same cell loss rate as central queued switching elements. However, output queuing requires simple control logic. Hence, since memory capacity is growing much faster than memory operating speeds, it is possible to design an ATM switch making use of output queued switching elements operating at high speeds.

To further speedup the implementation using output queued switching elements, a fifty three bytes wide memory can be used so that an ATM cell can be read or written in one access of the memory. Further improvement can be gained by using a multi-ported memory. For example, if a seventeen port, fifty-three byte wide memory is used in an output queued switching element, the required access time of the memory to read one cell is (53 * 8) / (150 * 10^6) = 2.82 milli-seconds. Even an implementation using a five port (one read by the output port and four writes by four input ports), fifty three byte wide memory has an ATM cell access time of is (53 * 8) / (4 *150) = 70.6 milli-seconds. A three port memory would need to have an access time of 35.3 milli-seconds, which is possible using current DRAM technology.

The two switching fabrics compared are a sorting circuit-banyan network and a stacked banyan-balancing network. In a sorting circuit-banyan network, ATM cells have to pass through three stages of the switching fabric, a sorting network, a trap network and a butterfly network. In a sorting circuit-banyan network having N-input and N-ouput ports, the sorting circuit stage has a depth of $\log_2(N)\ (1 + \log_2(N))\ /\ 2$ switching elements (comparators), as described in section 5-4. The trap network is implemented with a constant number (C) of levels of switching elements [37]. The butterfly network requires only $\log_2(N)$ switching elements. Therefore, the total number of switching elements an ATM cell has to pass through is $[\log_2(N)\ (1+\log_2(N))/2] + C + \log_2(N)$, which is of an order of $[\log_2(N)]^2$.

In a stacked banyan-balancing network, ATM cells have to pass through two stages, a balancing network and a banyan network. When an ATM cell arrives at the input port of a balancing network, it directs the cell to the appropriate stacked banyan network. This operation is quite fast as it involves looking up a small table. The next stage through which the cell passes is a banyan network. The number of switching elements through which the ATM cell has to pass is $\log_2(N)$, for an N-input and N-output fabric. Therefore, the total number of switching elements an ATM cell has to pass through is $\log_2(N)$, which is of an order of $\log_2(N)$ less than a sorting circuit-banyan network.

Appendix - Abbreviations

AAL	ATM adaptation layer
ABR	Available bit-rate
ANSI	American national standard institute
ATM	Asynchronous transfer mode
B-ISDN	Broadband integrated services digital network
B-NT	B-ISDN network termination
B-TA	B-ISDN terminal adaptor
B-TE	B-ISDN terminal equipment
CBR	Constant bit-rate
CCITT	International telegraph and telecommunication consultative committee
CI	Congestion indication
CL	Connection less
CLP	Cell loss priority
CO	Connection oriented
CRC	Cyclic redundancy check
CS	Convergence sub-layer
CSMA/CD	Carrier sense multiple access with collision detection
CSPDN	Circuit switched public data network
ETSI	European telecommunication standard institute
FDDI	Fiber distributed data interface
FEC	Forward error correction
FIFO	First in first out
FR	Frame relay
GFC	Generic flow control
HDTV	High definition television
HEC	Header error control
HOL	Head of line
IC	Input controller
ID	Identifier
IEEE	Institute of electrical and electronic engineers
IP	Internet protocol
ISDN	Integrated services digital network
ISO	International standards organization
ITU	International telecommunication union
LAN	Local area network
LLC	Logical link control
LSB	Least significant bit
MSB	Most significant bit
MSVC	Meta-signaling virtual channel
MT	Message type
MTP	Message transfer part
NNI	Network node interface
NPC	Network parameter control
OC	Output controller
OSI	Open system interconnection

PAD	Padding
PCI	Protocol control information
PDH	Plesiochronous digital hierarchy
PDU	Protocol data unit
PL	Physical layer
PM	Physical medium
PRM	Protocol reference model
PSPDN	Packet switched public data network
PSVC	Point-to-point signaling virtual channel
PSVCI	Point-to-point signaling virtual channel identifier
PT	Payload type
PTI	Payload type identifier
PV	Protocol version
PVC	Permanent virtual channel
QOS	Quality of service
RAM	Random access memory
RI	Reference identifier
SAP	Service access point
SAR	Segmentation and re-assembly
SDH	Synchronous digital hierarchy
SDU	Service data unit
SIR	Sustained information rate
SN	Sequence number
SONET	Synchronous optical network
SS7	Common channel signaling system number 7
ST	Segment type
STM	Synchronous transfer mode
SVC	Signaling virtual channel
TC	Transmission convergence sub-layer
TCP	Transmission control protocol
TE	Terminal equipment
TMN	Telecommunication management network
UDP	User datagram protocol
UNI	User network interface
UPC	Usage parameter control
VBR	Variable bit-rate
VC	Virtual Channel
VCC	Virtual channel connection
VCI	Virtual channel identifier
VP	Virtual path
VPC	Virtual path connection
VPI	Virtual path identifier
UBR	Unspecified bit-rate
WAN	Wide area network

References

---

1. ITU-T: Recommendation I.150. "BISDN ATM Functional Characteristics.," Geneva, 1993.
2. ITU-T: Recommendation Q.2110. "BISDN Signaling ATM Adaptation Layer (SAAL) – Service Specific Connection Oriented Protocol," Geneva, 1993.
3. ITU-T: Recommendation Q.2130. "BISDN Signaling ATM Adaptation Layer (SAAL) – Service Specific Coordination Function For The Support Of Signaling At The UNI," Geneva, 1993.
4. ITU-T: Recommendation Q.2140. "BISDN Signaling ATM Adaptation Layer (SAAL) – Service Specific Coordination Function For The Support Of Signaling At The NNI," Geneva, 1993.
5. ITU-T: Recommendation I.371. "Traffic Control And Congestion Control In B-ISDN," Geneva, 1993.
6. ITU-T: Recommendation I.356. "BISDN ATM Layer Cell Transfer Performance," Geneva, 1993.
7. ITU-T: Recommendation I.321. "ATM BISDN Protocol Reference Model And Its Application," Geneva, 1993.
8. ITU-T: Recommendation I.432. "BISDN User Network Interface Physical Layer Specification," Geneva, 1993.
9. ITU-T: Recommendation I.362. "BISDN ATM Adaptation Layer Functional Description," Geneva, 1993.
10. De Prycker, M., "Asynchronous Transfer Mode - Solutions For Broadband ISDN," Prentice Hall, 1995, pg. 160–181.
11. Glon, J. P., Debuysscher, P., Paul J. L., "An ATM Switching Unit Architecture For B-ISDN," International Conference on Communications (ICC) '90, Stockholm, June 1990, pg. 21–25.
12. Ahmadi, H., Denzel, W., "A Survey Of Modern High Performance Switching Techniques," IEEE Journal on Selected Areas in Communication, Vol. 7, No. 7, September 1989, pg. 1091-1103.
13. Karol, M., I Ching-Li, "Performance Analysis Of A Growable Architecture For Broadband Packet (ATM) Switching," Global communications (Globecom) '89, Dallas, November 1989, pg. 52-59.

14. Banniza, T. R., Eilenberger, G. J., Pauwels, B., Therasse, Y., "Design And Technology Aspects Of VLSI's For ATM Switches," IEEE Journal on Selected Areas in Communication, Vol. 9, No. 8, October 1991, pg. 1221-1231.

15. Gard, I., Rooth, J., "An ATM Switching Implementation – Technique And Technology," ICC '90, Stockholm, June 1990, pg. 43-47.

16. Barri, P., "A Switching Element For Asynchronous Transfer Mode," Fifth International Workshop on Integrated Electronics and Photonics in Communication, North Carolina, October 1987, pg. 14-20.

17. Hluchyi, M. G., Karol M. J., "Queuing In Space Division Packet Switching," Information and Communication Conference (Infocom)'88, New Orleans, March 1988, pg. 19-26.

18. Karol, M. J., Hluchyj, M. G., Morgan S. P., "Input Versus Output Queuing In A Space-Division Packet Switch," IEEE Transactions on Communication, Vol 35, No. 12, December 1987, pg. 1347-1356.

19. Handel, R., Huber, M. N, Schroder, S., "ATM Networks – Concepts, Protocols, Applications," 1995, Addison Wesley, pg. 170-183.

20. Degan, J. J., Luderer, G. W. R., Vaidya, A. K., "Fast Packet Technology For Future Switches," AT&T Technical Journal, Vol. 68, No. 2, March / April 1989, pg. 36-50.

21. Luderer, G. W. R., Knauer, S.C., " The Evolution Of Space Division Packet Switches," Proceedings of the XIII International Switching Symposium, Vol. V, Stockholm, 1990, pg. 211-216.

22. Tobagi, F. A., "Fast Packet Switch Architectures For Broadband Integrated Services Digital Networks," Proceedings of the IEEE, Vol. 78, No. 1, January 1990, pg. 133-167.

23. Goke, L. R., Lipovski, G. J., "Banyan Networks For Partitioning Multiprocessor Systems," First Annual Symposium on Computer Architecture, 1973, pg. 21-28.

24. Padmanabhan, K., Lawrie, D. H., “A Class Of Redundant Path Multistage Interconnection Networks,” IEEE Transactions on Computers, Vol. 32, No. 12, December 1983, pg. 2103-2110.

25. Lutz, K. A., “Considerations On ATM Switching Techniques,” International Journal of Digital and Analog Cabled Systems, Vol. 1, No. 4 April 1992, pg. 237-243.

26. Theimer, T.H., “Performance Comparison Of Routing Strategies In ATM Switches,” Proceedings of the XIII international Teletraffic Congress, Copenhagen, 1991, pg. 923-928.

27. Lea, C.T., “Multi–Log N Self-Routing Networks And Their Applications In High Speed Electronic And Photonic Switching Systems,” Proceedings of the Infocom ’89, Ottawa, 1989, pg. 877-886.

28. Huber, M. N., Rathgeb, E. P., Theimer, T. H., “Self Routing Banyan Networks In An ATM Environment,” ICC ’88, Tel Aviv, October 1988, pg. 167-174.

29. Fischer, W., Fundneider, O., Goeldner, E. H., Lutz, K. A, “A Scalable ATM Switching System Structure,” IEEE Journal on Selected Areas in Communications, Vol.. 9, No. 8, October 1991, pg. 1299-1307.

30. Yeh, Y. S., Hluchyj, M.G., Acampora, A., “The Knockout Switch: A Simple, Modular Architecture For High Performance Packet Switching,” ICC ’87, Phoenix, March 1987, pg 37-44.

31. Yeh, Y. S., Hluchyj, M.G., Acampora, A., “The Knockout Switch: A Simple, Modular Architecture For High Performance Packet Switching,” IEEE Journal on Selected Areas in Communications, Vol. 5, No. 8, October 1987, pg 701-715.

32. Eng, K.Y., Karol, M.J., Cyr, G.J., Pashan, M.A., “A 160-Gb/S ATM Switch Prototype Using The Concentrator-Based Growable Switch Architecture,” ICC '95, Seattle, vol.1, pg. 550 –554.

33. Coudreuse, J. P., Servel, M., “Prelude: An Asynchronous Time-Division Switched Network,” ICC ’89, Seattle, June 1989, pg 22-28.

34. Chao, H.J., Byeong-Seog Choe, “Design And Analysis Of A Large-Scale Multicast Output Buffered ATM Switch,” IEEE/ACM Transactions on Networking, 1997, vol 32 , pg. 126-138

35. Hashemi, M.R.; Leon-Garcia, A., "The Single Queue Switch," Proceedings of the sixteenth annual joint conference of the IEEE computer and communications societies, 1997, vol 2 , pg. 533-540.

36. Hennessy J. L., Patterson D. A., "Computer Architecture a Quantitative Approach," 1996, Morgan Kaufmann, pg. 372-483.

37. Leighton F. T, "Introduction to parallel algorithms and architectures: Arrays, trees, hypercubes," 1992, Morgan Kaufmann , pg. 525-650.

Vita

Navindra Yadav was born in India on April 13, 1974 to Indra Yadav and N.S. Yadav. He attended Maulana Azad College of Technology, Bhopal, India and received a Bachelor of Engineering degree in Computer Science and Engineering in 1995. He worked for two years with Siemens Information Systems Ltd. In August 1997, he started his graduate studies in the Department of Electrical Engineering and computer Science, Lehigh University. Mr. Yadav is a student member of IEEE.

END

OF

TITLE