

Traffic-Driven Low-Power Design and Modeling of VLSI Satellite Switching Fabrics

by

Amr G. Wassal

A thesis

presented to the University of Waterloo

in fulfilment of the

thesis requirement for the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2000

©Amr G. Wassal 2000



National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-53523-1

Canada

The University of Waterloo requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

Abstract

The dream of the global village is rapidly becoming a fact of life with a dense and diverse web of communication links and technologies extending all over the world. However, a few spots here and there are yet to be connected. Moreover, the new nomads of the communication age demand to be connected whether in the sea, in the sky or on the road. *Global satellite networks* promised in the next few years, that move the backbone into the sky connecting people from virtually everywhere on earth, seem to be the answer. Nevertheless, these global satellite networks will not become a reality until a myriad of technical problems and issues are resolved.

This work discusses the important issue of migrating the terrestrial packet switching fabrics on-board satellites. The approach is three-fold. First, the physical and performance constraints imposed by the new environment must be identified and used to choose and adopt a suitable fabric architecture from the wide variety available. Next, a mathematical model is formulated for the fabric architecture to optimize certain requirements, power dissipation in this case, under the remaining constraints. This is used to explore the design space at the system-level and choose appropriate values for different parameters such that the architecture satisfies the requirements and constraints. Finally, the buffer management problem and the scheduling engine design are investigated in more detail. The scheduling engine is one of the major components of the fabric and an architecture based on circular sorting queues is proposed, analyzed, and prototyped.

Acknowledgements

All praise is due to Allah, Most Gracious, Most Merciful, Whose help and bounties are ever dominating throughout my life.

I would like to express my gratitude to my supervisor, Prof. M.A. Hasan, for his support, encouragement and supervision throughout my program. My deep appreciation goes to my colleagues and friends in the VLSI and the broadband communications research groups for many helpful and valuable suggestions. I also would like to thank Prof. A. Vannelli for his insightful discussions and suggestions.

This work has been supported in part by NSERC, the Canadian Microelectronics Corporation and an Ontario Graduate Scholarship. This support is greatly appreciated.

My deep gratitude to my parents and my wife for their continuous encouragement, patience, support and assistance.

Contents

1	Introduction	1
1.1	Motivation and Problem Definition	1
1.2	Solution Approach and Formulation	4
2	Switching and Teletraffic Engineering	6
2.1	Introduction	6
2.1.1	STM, Packet Routing & ATM	7
2.1.2	Layer 2-3 Integration	8
2.2	A Generic Switching Architecture	10
2.2.1	Definitions	12
2.3	Teletraffic Modeling	14
2.3.1	Traffic Source Models in Broadband Networks	14
2.3.2	Modeling of Queueing Systems	23
2.3.3	Analytical Solution Methods	24
2.3.4	Performance Metrics	28
2.4	Buffering Schemes	29
2.4.1	Output Buffering	30

2.4.2	Input Buffering	31
2.4.3	Combined Input-Output Buffering	32
2.5	Classification of Switching Fabrics	33
2.5.1	Shared Memory Approach	34
2.5.2	Shared Medium Approach	35
2.5.3	Fully Interconnected Approach	36
2.5.4	Space Division Approach	37
2.6	Summary	40
3	Shared Memory Switching Fabric	43
3.1	Introduction	43
3.2	Requirements for On-Board Satellite Systems	43
3.3	Switching Fabric Architectures	48
3.3.1	Shared Memory Architectures	49
3.3.2	Buffer Sharing Effect	53
3.4	Proposed Architecture	60
3.4.1	Memory Bandwidth Issues	62
3.4.2	Design Scalability	64
3.4.3	Fault Tolerance and Reliability	67
3.5	Summary	72
4	Power Optimization Methodology	74
4.1	Introduction	74
4.2	Low-Power System-Level Design Techniques	76
4.2.1	Power Measurements	76

4.2.2	Power Sources	77
4.2.3	Power Estimation and Reduction Techniques	79
4.2.4	Low-Power Approach	85
4.3	System-Level Power Optimization of the Shared Memory Architecture	90
4.3.1	Power Analysis and Modeling	91
4.3.2	Performance Modeling	100
4.3.3	Other System-Level Requirements	109
4.3.4	Error Sensitivity of the Model	112
4.3.5	Optimization for Low-Power	115
4.3.6	Scalable Architecture Model	121
4.4	Summary	125
5	Scheduling and Buffer Management	127
5.1	Introduction	127
5.2	Buffer Management Using Sorting Queues	131
5.2.1	Architectures for Sorting Queues	132
5.2.2	The Proposed Sorting Queue	133
5.2.3	Design of the Tag Structure	137
5.2.4	Multicast Support	138
5.3	Analysis of Priority Queues	144
5.3.1	Multi-class Priority Analysis	145
5.3.2	Numerical Results	153
5.4	VLSI Implementation	159
5.5	Summary	162

6 Conclusion and Future Work	163
Bibliography	169

List of Tables

1.1	Recent global satellite networks projects and their main features.	2
2.1	Survey of the state-of-the-art ATM fabrics in the recently published literature	41
3.1	The main features of the different fabric design approaches.	49
3.2	Comparison of basic shared memory ATM switching architectures.	50
4.1	Experimental results for low-power fabric design.	121
4.2	Experimental results for scalable low-power fabric design using $\lambda = 0.35\mu\text{m}$, $\text{CLP} = 10^{-10}$ and $\mathcal{N} = 512$	125

List of Figures

1.1	TIA's TR34.1 architecture for SATATM.	3
2.1	An IP/ATM architecture showing how the switches and routers handle different flows.	9
2.2	A generic model for ATM switches.	10
2.3	Time scale hierarchy in traffic modeling.	17
2.4	State transition diagram for an ON-OFF source and the corresponding average periods.	18
2.5	Examples of different buffering schemes.	30
2.6	Basic structure of a shared memory switch.	34
2.7	A shared bus switch.	36
2.8	A fully interconnected switch.	37
2.9	A switching element, a 4×4 banyan and an 8×8 banyan switching fabric.	39
3.1	Linked-list based ATM switch architecture.	51
3.2	SBDA ATM switch architecture.	51
3.3	Hybrid shared and dedicated output buffer architecture.	52

3.4	CAM-based ATM switch architecture.	52
3.5	State-transition diagram of an output buffer occupancy.	54
3.6	Analytical results for (a) output and (b) shared buffers with random uniform offered load $\rho = 0.8$	59
3.7	Analytical and simulation results for different switching fabrics using shared memory.	59
3.8	Proposed shared memory switching fabric architecture.	61
3.9	An example of multicasting from memory block SMB2 to output ports OP1 and OP3.	61
3.10	The relation between the DRAM density and its peak data rate for recent technologies.	63
3.11	General configuration of $\mathcal{C}(i, j, n_1, n_2, n_3)$ Clos network.	65
3.12	Recursive approach to the SNB Clos network.	66
3.13	General configuration of $\mathcal{C}(i, j, n_1, n_2, n_3)$ Clos network with a single redundancy at slice and module levels.	69
3.14	Common commutation techniques (a) mux/demux, (b) ladder and (c) bus.	71
4.1	Low-power design flow: (a) classical feedback flow, (b) level-by-level flow.	81
4.2	Reducing the power by frequency reduction and parallelization.	86
4.3	Register banks with (a) synchronous load enable or (b) clock gating and (c) its timing diagram.	89

4.4	Transition activity factor vs. bit number in an ATM cell for various traffic streams.	97
4.5	Sample results from the solution of the D-BMAP modeled traffic and their linear approximations on a logarithmic scale.	104
4.6	Sample results of the matrix-geometric algorithm solving a D-BMAP model of 8 identical bursty sources with $\rho \approx 1.48$	107
4.7	Relative errors for power dissipation estimates (a) fixing the $IP_{c,1}$ traffic model and (b) fixing the $0.35\mu\text{m}$ CMOS technology.	113
4.8	Relative errors for power dissipation estimates for different power macro-models (using the $IP_{c,1}$ traffic model and the $0.5\mu\text{m}$ CMOS technology).	114
4.9	Relative errors for the area estimates compared to the actual layout area for different fabrication technologies.	115
4.10	Switching fabric design in satellite applications at the system-level formulated as an integer non-linear optimization problem.	116
4.11	Normalized figure of merit for the proposed architecture implemented using different technologies and parameter.	118
4.12	The traffic models used to define the performance constraints through the stages of a Clos network.	124
4.13	Relative error for the power estimates using the shared memory modules framework as compared to a detailed formulation framework (The error surface is fit to the sampled points).	124
5.1	The control algorithm of the sorting units.	134

5.2	The basic structure of a sorting unit.	136
5.3	Architecture of the circular-queue management unit.	137
5.4	The tag cell structure.	137
5.5	Simulation results for different multicasting traffic management schemes. Effective load at the inputs versus: (a) average throughput, (b) cell loss probability, (c) unicast cells average delay, and (d) multicast cells average delay.	141
5.6	An example of multicasting using SWSR with output masking from a shared memory block to several output ports.	144
5.7	Finite buffer model for three priority classes.	154
5.8	Cell loss probability vs. total offered load.	155
5.9	Cell loss probability vs. buffer size.	156
5.10	Mean waiting time vs. total offered load.	157
5.11	Mean waiting time vs. buffer size.	158
5.12	Microphotograph of the prototype of the queue management unit. . .	160
5.13	Block diagram of the internal testing structures in the chip.	161

Chapter 1

Introduction

1.1 Motivation and Problem Definition

Within a few years, several satellite constellations will be launched and devoted to broadband communications with on-board processing and switching capabilities. Motivated by the rapid growth of the Internet, the increasing demand for bandwidth and service and the potential of expanding and connecting terrestrial networks and providing services to areas with limited or underdeveloped communications infrastructures, the research community was prompted to concentrate its efforts on finding efficient architectures for the backbone in the sky or the *global satellite networks*.

Today, several commercial projects are underway, some of which are listed in table 1.1 along with their main features [2, 95]. Governmental research, such as NASA's, is also shaping these networks [60] and lately several standardization bodies such as the ATM Forum, the ITU and the TIA/EIA have started devising stan-

Table 1.1: Recent global satellite networks projects and their main features.

Project	Origin	Constellation	On-board Switching	ISL	Data Rate
ASTROLINK	USA	9 GEOs	ATM-based	Yes	Up to 9.6Mbps
CELESTRI	USA	9 GEOs and 63 LEOs	ATM-based	Yes	Up to 155Mbps
CYBERSTAR	USA	3 GEOs	Packet-based	TBD	400K/30Mbps
EUROSKYWAY	Europe	8 GEOs	Packet-based	Yes	Up to 6Mbps
N-STAR	Japan	GEOs	ATM-based		
SKYBRIDGE	Europe	64 LEOs	Bent Pipe	No	16K-2Mbps uplink 16Kbps downlink
TELEDESIC	USA	288 LEOs	Packet-based	Yes	16K-64Mbps
WEST	Europe	GEOs and MEOs	ATM-based	TBD	TBD

dards and regulations for broadband networks over satellites [30, 38]. The network architecture SATATM defined by the TIA's TR34.1 committee is shown in figure 1.1. The ITU's WP4B is developing recommendations for performance, *S.atm*, and availability, *S.atm-av*, objectives for ATM over satellites to complement existing recommendations, I.356 and I.357, for terrestrial ATM networks.

Attractive proposals for the transport technology to be used include the *Asynchronous Transfer Mode*, ATM, on one hand with its support for multimedia traffic, guaranteed quality of service, QoS, near-error-free performance characteristics [6, 58]. On the other hand, the ubiquitous *Transmission Control Protocol/Internet Protocol*, TCP/IP, stack is another candidate. Various resources in academia, industry and even standardization entities have been devoted recently to study the viability of ATM and TCP/IP over satellites [27, 48]. Most of this research focuses on developing efficient access schemes, traffic and congestion control schemes, error control coding techniques and seamless integration with existing networks [95].

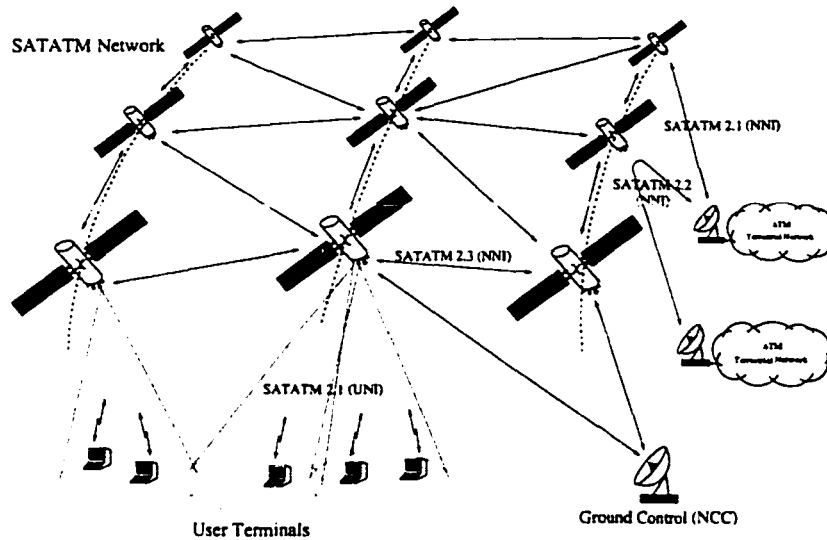


Figure 1.1: TIA's TR34.1 architecture for SATATM.

At the same time and irrespective of the transport technology to be adopted, the operating conditions in the satellite environment impose a set of strict physical requirements on the electronic processing systems on-board [46]. These requirements include limited power dissipation so as not to burden the limited satellite resources which are based mainly on stored solar energy. The processing systems design should be subject also to limited payload size and weight. The space environment imposes thermal dissipation and radiation resistance requirements which can be interpreted into power dissipation, cooling techniques, packaging technologies, fault-tolerance and error control constraints. Launching the satellites can impose more requirements such as mechanical vibrations resistance among others.

In this work, several tradeoffs and architectural designs are investigated in order to address the problem of satisfying the two sets of physical and performance

constraints simultaneously through efficient VLSI architectural design and modeling. Performance constraints are derived from the standards mentioned above so as to make the implemented system compliant with them. However, since physical requirements are dependent on the satellite implementation and the resources available on-board, these will have to be determined and accounted for on a case-by-case basis and provisions to accommodate that are made in the design methodology.

1.2 Solution Approach and Formulation

The application considered in this work is on-board packet switching fabrics. With the very stringent implementation requirements and strict performance constraints of this application, our objective is to find methods to develop switching fabrics that utilize the available high performance VLSI technologies in such a way so as to integrate as much as possible of the switch functionality within as few chips as possible, and to employ power reduction techniques to develop a low-power switch. To this end, chapter 2 presents an overview of the state of the art in switching fabric architectures and buffering schemes. It also gives an introduction to teletraffic engineering which will be used throughout this work to ensure that the traffic metrics used to quantify the architecture performance are within the specified constraints.

In chapter 3, the design requirements for satellite switching fabrics are discussed in detail. Then, the use of shared memory architectures is justified and a shared memory switching fabric is proposed. Architectural considerations to satisfy both sets of on-board physical and performance constraints are also discussed. Low-

power system-level design techniques are outlined at the beginning of chapter 4 followed by a traffic-driven design framework that formulates the VLSI design of the switching fabric as an integer non-linear optimization problem. In this problem, power dissipation is the objective function to be minimized under several constraints defined by the remaining physical and performance constraints. This framework uses a teletraffic model to integrate the statistics of the network traffic into the power dissipation model as well as in the buffer sizing problem. The buffer management problem is discussed in chapter 5 and a versatile architecture for buffer management is presented along with its VLSI implementation. Finally, chapter 6 presents our conclusions, summarizes the contributions presented in this thesis and outlines possible future extensions of this work.

Chapter 2

Switching and Teletraffic Engineering

2.1 Introduction

The aim of an ATM switch design is to increase speed, capacity and overall performance of the *Broadband-Integrated Services Digital Network*, B-ISDN. ATM switching differs from conventional switching because of the high-speed interfaces to the switch. In addition, the statistical capability of the ATM streams passing through the switching systems places additional demands on the switch. Moreover, transporting various types of traffic, each with different requirements and quality of service, is not a trivial matter. To meet all of these requirements, ATM switches have to be significantly different from conventional switches. Various design alternatives were proposed in the literature and many of them were actually implemented in commercial products.

2.1.1 STM, Packet Routing & ATM

Synchronous transfer mode, STM, or *circuit switching* was the first switching and multiplexing¹ technique to be considered for B-ISDN due to its compatibility with most existing systems that employ time-division multiplexing. However, rigidly-structured STM is very inefficient in handling the different and variable-bit rates required by the services which are expected to be supported by B-ISDN.

In packet networks such as those based on the TCP/IP protocol stack, packets are transmitted from a source to a destination through multiple routers in a store-and-forward manner. Sources transmit their packets as soon as they are available, which is referred to as, *statistical multiplexing*. In contrast to circuit switching, the switching function here needs to be performed only when packets, or cells, are present at the inputs of the router. These networks were designed when only poor transmission links were available. Complex protocols were necessary to perform flow and error control to offer an acceptable end-to-end performance. Also, complex buffer management was required because of the variable length packets. Consequently, large delays and jitters made conventional packet switching unsuitable for high-speed integrated services networks.

ATM has been chosen as the transfer mode for B-ISDN because it overcomes the problems of STM and conventional packet routing altogether. ATM is a high-speed connection-oriented packet-switching technique with minimal functionality in the network. High-level protocols, such as error control, are performed on an end-

¹Multiplexing refers to the arbitration of access to a link and should be distinguished from switching.

to-end not a link-to-link basis. This is possible because of the high quality links used. Also, ATM uses fixed-length packets, called cells, that consist of 53 bytes each; 48 for the payload and 5 for the header. This choice simplifies the design of switching nodes, and reduces delay and jitter. Being a connection-oriented transfer mode, ATM connections involve three phases: set-up, information transfer, and tear-down. It is worth pointing out that the term “asynchronous” in ATM does not necessarily imply asynchronous transmission or switching systems. Rather, it implies aperiodicity; i.e., no source shall own a time slot on a periodic basis [5]. The association of a cell with a connection is made explicit through the header of that cell.

2.1.2 Layer 2-3 Integration

Because of the ubiquity of the TCP/IP protocol stack and to take advantage of the high throughput and QoS that could be attained using switching technologies, many different proposals have been recently put forward to integrate layer 2 switching, such as ATM switching, and layer 3 routing, such as IP routing. Such proposals include Ipsilon’s IP switching, Toshiba’s CSR, CISCO’s Tag switching and IBM’s ARIS [32]. The *Internet Engineering Task Force*, IETF, is developing an *Internet Integrated Services* model. The working group on *Multi-Protocol Label Switching*, MPLS, aims to standardize the paradigm that integrates layer 2 switching with layer 3 routing.

Regardless of the details of the implementation, the switching fabric used in ATM switches is still at the core of each of those proposed switching routers. We

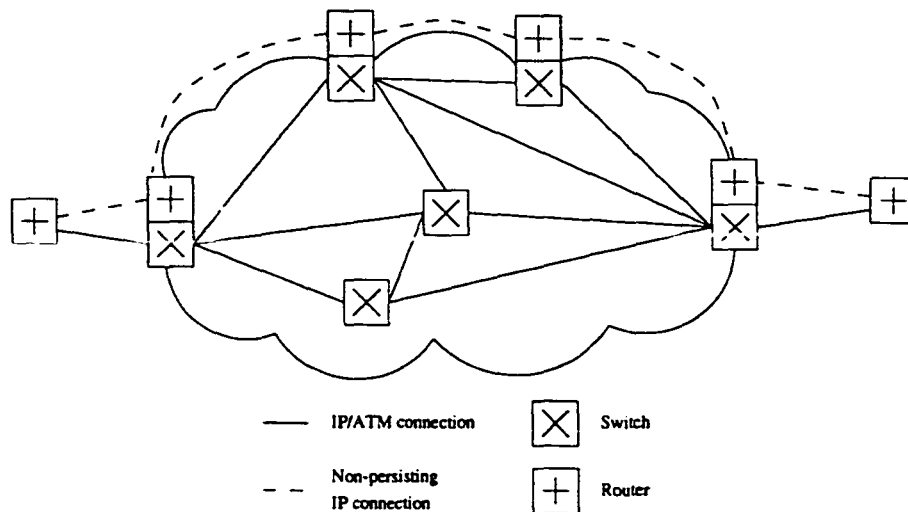


Figure 2.1: An IP/ATM architecture showing how the switches and routers handle different flows.

will refer to these proposals as IP/ATM switching. IP/ATM switching involves identifying persistent IP traffic flows that require some degree of QoS to be provided and guaranteed. These flows are identified by their source-destination IP address pair and possibly by their TCP socket address and mapped to a switched path. IP packets are then segmented into fixed size cells at the edge of the switching cloud and switched through to be reassembled into IP packets at the other end as shown in figure 2.1. The remaining IP traffic is routed through regular IP routers.

The statistical characteristics of the ATM cell traffic generated from segmenting an IP flow are different from those of native ATM traffic. Similarly, the characteristics of the segmented traffic from several multiplexed IP flows are also different from those of a traffic composed of several multiplexed ATM connections. This important point will be elaborated and taken into consideration in chapter 4 where

we discuss the system-level optimization approach.

2.2 A Generic Switching Architecture

A functional block model will be adopted to simplify the discussion of various design alternatives [28]. The switch functions will be divided among the following broad functional blocks, namely, input modules, output modules, cell switching fabric², connection admission control, and switch management. Figure 2.2 illustrates this switching model. The cell switch fabric is our main concern and will be discussed in detail later while the other blocks will be discussed briefly in this section.

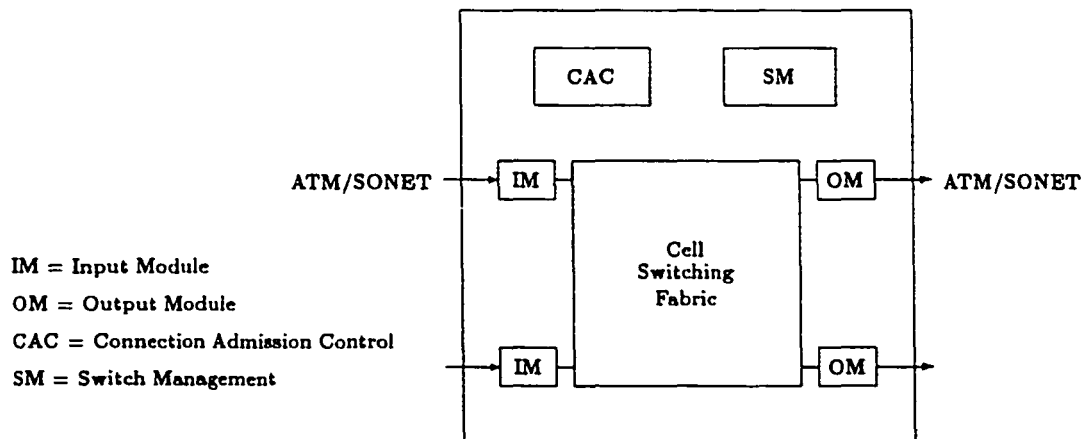


Figure 2.2: A generic model for ATM switches.

- *Switch Interface.* The switch interface includes both input and output modules. Input modules terminate the incoming signal, assuming a SONET sig-

²In this context, we are using the terms *cell* and *packet* interchangeably. Strictly, packets with a fixed size are called cells in some data communications contexts as in ATM systems.

nal, and extract the ATM cell stream. This involves signal conversion and recovery, processing SONET overhead, and cell delineation and rate decoupling. After that, for each ATM cell the header is checked for errors using the *Header Error Control*, HEC, field, the VPI/VCI values are validated and translated, the destination output port is determined, signaling cells are passed to CAC and Switch Management OAM cells to Switch Management, UPC/NPC are determined for each VPC/VCC and finally an internal tag containing internal routing and performance monitoring information for use only within the switch is added. Output modules prepare the ATM cell streams for physical transmission by removing and processing the internal tag, possible translation of VPI/VCI values. HEC field generation, possible mixing of cells from CAC and Switch Management with outgoing cell streams, cell rate decoupling, mapping cells to SONET payloads and generation of SONET overhead and finally by conversion of the digital bit stream to an optical signal.

- *Connection Admission Control CAC*. CAC establishes, modifies and terminates virtual path/channel connections. More specifically, it is responsible for high-layer signaling protocols, signaling ATM Adaptation Layer (AAL) functions to interpret or generate signaling cells, negotiating, renegotiating traffic contracts with users requesting VPCs/VCCs and allocating switch resources for them.
- *Switch Management*. The switch management module handles physical layer OAM, ATM layer OAM, security control for the switch database, usage measurements of the switch resources, management of configuration and traffic.

This area is difficult and still under development because management covers an extremely wide spectrum of activities.

- *Switching Fabrics.* They are primarily responsible for transferring cells between the other functional blocks, i.e., routing of data cells and possibly signaling and management cells as well. Other possible functions include cell buffering, traffic concentration and multiplexing, providing redundancy for fault tolerance, multicasting or broadcasting, cell scheduling based on delay priorities and congestion monitoring. The routing and buffering functions are the two major functions performed by the switching fabrics. In this section, we will give a few essential definitions that are more important for space-division switches than others, followed by a list of performance measures used to evaluate the switch. Some of the most common traffic models used to evaluate the switches are mentioned afterwards. Buffering schemes will be discussed in the following section, followed by a classification of the switching fabrics.

2.2.1 Definitions

- *Internal Blocking.* It occurs when two or more cells, destined to different output ports, contend over the same internal link. A switch is called non-blocking if it does not suffer from internal blocking.
- *Output Blocking.* It occurs when more than one cell request the same output port within the same time slot. A blocking switch suffers from both internal and output blocking. An output non-blocking switch is able to clear all

incoming cells in any given time slot to the buffers of the requested output ports before the next time slot.

- *Cell Sequencing.* A switch preserves cell sequencing if, for each input-output pair, it delivers incoming cells in the order by which they have arrived.
- *Self Routing.* A switching fabric is called a self routing fabric if the cell output address specifies a unique path between the input and output ports through the fabric. This is also known as digit-controlled routing.
- *Speed-Up.* Sometimes, it is necessary to speed up an ATM switching fabric to improve the performance and compensate for processing overheads. Fabric speed-up can be implemented in time or in space.
- *Multicasting.* B-ISDN must support multipoint communications in which more than two users participate in a connection. In an $N \times N$ switch, this refers to the situation where an incoming connection requests K output ports, where $1 < K < N$. When $K = 1$, this is point-to-point connection, or unicast, while $K = N$ refers to broadcasting.
- *Scalability.* A scalable switch has a modular design such that it can scale over a wide range of switching and buffering capacities using commonly available implementation technology.
- *Priority function.* It is the ability to differentiate among packets or cells according to priority information provided in them, and to give preferential treatment to higher priority packets.

2.3 Teletraffic Modeling

Before we discuss the existing switching architectures, an introduction to the analytical tools used to select the design parameters for these architectures and to evaluate their performance is in order. A realistic estimate of the traffic and signal activity exhibited in each of the building blocks of an architecture can not be realized accurately without some knowledge of the statistical characteristics of the network traffic coming into those blocks and leaving them. This is where *Teletraffic* comes in. Teletraffic is an engineering branch, ubiquitous in communications and networking, supported by fundamental mathematical research in both industry and academia.

Teletraffic engineering was used to study the statistical behavior on the call level only in the telephone era. In the emerging multimedia era, it is studying not only the number and duration of calls but also the statistical properties of the information flow during a call. Research results in this domain have an immediate practical significance for the engineering and performance evaluation of switches, multiplexers, flow-control algorithms, etc.

2.3.1 Traffic Source Models in Broadband Networks

Teletraffic engineering advances the development of models that try to capture the salient features of multimedia traffic using stochastic processes. Good source models should satisfy several criteria:

1. *proximity to real sources* in that they should render the relevant statistical properties of real traffic and should be easy to fit to real sources,

2. *generality* and appropriateness for modeling aggregate and/or output traffic,
3. *analytical tractability and accuracy* so that the behavior of a queueing system fed by such source can be studied, and
4. *ease of implementation* by means of both computer simulation or hardware-based traffic generators.

The following refers to the traffic model as seen by the input ports of the switch. The traffic model is described by two random processes. The first, and the most important one for our purposes, is the process that governs the arrival of cells in each time slot. The second process describes the distribution by which arriving cells choose their destination ports. Unless we mention otherwise, we will assume that an incoming cell chooses its destination port uniformly among all N output ports, and independently from all other requests, with probability $1/N$ [128]. Traffic source models can be classified into two classes: uniform and bursty.

- *The Uniform Traffic Model*

In this model, cells arrive at the input ports of the switch according to independent and identically distributed Bernoulli processes, each with parameter p , $0 < p \leq 1$. In other words, at an input port in a given time slot, a cell arrives with probability p , and there is no arriving cell with probability $1 - p$. Thus, p represents the input load or arrival rate at each input port of the switch. Asserting the assumption of uniformity for real-life situations may lead to optimistic evaluations of performance measures. However, a large number of studies on the performance evaluation of ATM switches assume

uniform traffic. The main reasons behind this trend are as follows:

1. This assumption makes the analytical evaluation more tractable especially for complex buffering strategies.
2. A distribution network can be used at the front end of the switch to randomize incoming traffic.
3. It was observed that the inherent smoothing, that takes place when bursty traffic is queued and then released at a given rate, makes the traffic less bursty. Furthermore, it was shown that subsequent stages of switching cause the traffic to become even less bursty, making the uniform traffic assumption closer to reality [43].

- *Bursty Traffic Models*

In broadband networks, cells tend to arrive in clusters rather than in smooth streams. In this sense, a periodic traffic stream does not capture this *burstiness* characteristic. Different source models depend on different descriptors to characterize the burstiness property [124] such as the ratio of the peak and mean traffic rates, the coefficient of variation of the traffic load ρ and the index of dispersion for counts, IDC, or for intervals, IDI. Others use the spectral properties of the traffic or the evolution of the traffic stream's entropy. In such models, all cells belonging to the *same* burst are assumed to address the same output port which is chosen uniformly among all N output ports, independently from all other bursts.

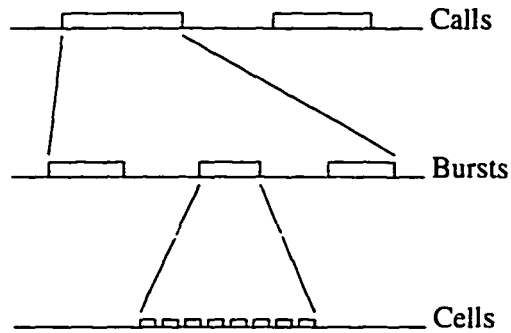


Figure 2.3: Time scale hierarchy in traffic modeling.

Traffic source models can be classified according to their time scale as shown in figure 2.3. Cell scale models are needed to study the mechanisms that handle individual cells, such as switch architectures, synchronization of real time services and flow control techniques. Burst scale models characterize the traffic by its instantaneous rate and appear naturally in *fluid flow* solution methods. Call scale models characterize the traffic by the holding time of arriving calls or service demands. Associated with each call is an *effective bandwidth* that describes the amount of bandwidth that should be allocated for this call [57,116]. They can also be classified according to their correlation structures. Short-range-dependent models have a correlation that is significant for relatively small lags only while long-range-dependent models have significant correlations even for large lags. Stochastic self-similar or fractal models are those that retain the same statistics over a range of scales. These are most evident in local area networks traffic [116]. Some of the major bursty traffic models are outlined below.

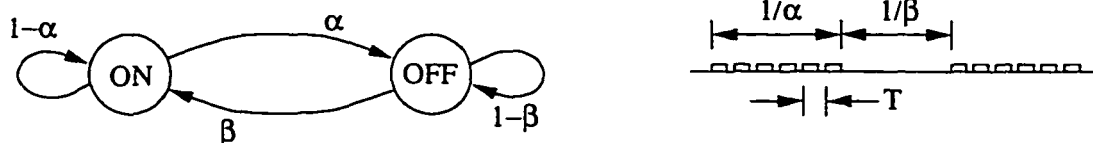


Figure 2.4: State transition diagram for an ON-OFF source and the corresponding average periods.

1. ON-OFF Source Models

Essentially, a general source model would take into account the entire past history of the traffic generation so as to determine the time for the next cell. However, such models would be very complicated to describe, to fit to real traffic and to analyze. Most models resort to limited interdependency and use the well known Markovian or memoryless property which assumes that the next state depends only on the current state of the source but not on its past history nor on the time already spent in the current state. The state space of the underlying stochastic process can then be described by a *discrete time Markov chain*, DTMC, or a *continuous time Markov chain*, CTMC, depending on whether the state transitions occur at discrete instants in time or at continuous time³.

The simplest and most popular model is the ON-OFF model shown in figure 2.4. In this model, cells are generated at the ON or active state only with fixed interarrival time and the average time spent in the ON or the OFF state is geometrically distributed if slotted, or exponentially distributed if continuous

³In what follows we will limit our discussions to discrete-time models because of the use of slotted time in ATM technology.

time, with means $1/\alpha$ and $1/\beta$ respectively. A discrete-time ON-OFF source has the transition matrix \mathbf{D} .

$$\mathbf{D} = \begin{pmatrix} 1 - \alpha & \alpha \\ \beta & 1 - \beta \end{pmatrix}. \quad (2.1)$$

The equilibrium distribution (π_0, π_1) is given by

$$\Pr(\text{state is ON}) = \pi_0 = \frac{\beta}{\alpha + \beta}, \quad (2.2)$$

$$\Pr(\text{state is OFF}) = \pi_1 = \frac{\alpha}{\alpha + \beta}. \quad (2.3)$$

If the average interarrival time in the ON state is T cell time slots, then the average load in the ON period is $\lambda_0 = 1/T$ and the overall average load produced by such source model is

$$\rho = \pi_0 \lambda_0 = \frac{\beta}{T(\alpha + \beta)}. \quad (2.4)$$

Other variations include the *switched Bernoulli process*, SBP, where the cell generation is defined as a Bernoulli process with a different parameter for each state and the *interrupted Bernoulli process*, IBP, where the cell generation is defined as a Bernoulli process with parameter λ in the ON state and as zero cells in the OFF state. The continuous versions have a Poisson process defining the cell generation and are called the *switched Poisson process*, SPP,

and the *interrupted Poisson process*, IPP.

2. *Markov Modulated Sources*

A Markov modulated source is a generalization of the ON-OFF source. It has a cell generation process that changes state according to a Markov chain. The states of this chain are called its *phase* to distinguish it from the state of the overall arrival process. The Markov chain is an embedded process at the instants of phase change. A general source would have m phases and would generate cells with a definite rate r_i when in phase i . The arrival process may therefore be called a Markov modulated rate process, MMRP. This type of sources has the following degrees of freedom [97]:

- (a) The number of phases of the Markov chain.
- (b) The transition probabilities or the modulator structure which is usually assumed to have only transitions to the nearest neighbors, which is known as a *birth-death process*, to simplify the analysis.
- (c) The process in each phase. In general, different phases may have completely different processes, however, this is rarely used. Cells may be generated according to a Poisson process with a different parameter for each phase and this would be a *Markov modulated Poisson process*, MMPP. In time slotted systems, a Bernoulli process would be used and the source is a *Markov modulated Bernoulli process*, MMBP. If a deterministic or periodic process is used, the source is a *Markov modulated Deterministic process*, MMDP. Batch arrivals are also possible. The SBP, IBP, SPP and IPP sources mentioned before are special cases with only two states.

- (d) The time evolution of the Markov chain. CTMC modulators produce exponentially distributed sojourn times in each phase. On the other hand, DTMC modulators make transitions at slotted instants in time and have geometrically distributed sojourn times for their phases.

A universal model that encompasses many special cases is Neuts' versatile *Markovian arrival process*, MAP [102], and its generalization to include batch arrivals, BMAP. The discrete-time version was given by Blondia [17] and called the D-BMAP model. This model will be explained in detail and utilized in chapter 4.

3. *Regression Models*

Regression models define explicitly the next random variable in the sequence by previous ones within a specified time window and a moving average of a white noise. Such models include the *autoregressive moving average*, ARMA, model and the *autoregressive integrated moving average*, ARIMA, model which are usually used to model VBR video and MPEG traffic. Another variation is the *transform-expand-sample*, TES, model which is based on non-linear regression with modulo-1 arithmetic. It aims to capture both autocorrelation and marginal distribution of empirical data. These models however are not amenable to analysis.

4. *Long-Range-Dependent and Self-Similar Processes*

A wide-sense stationary stochastic process X_t with a covariance $\text{Cov}\{X_t, X_{t+n}\}$

is said to be long-range-dependent if

$$\sum_{n=-\infty}^{\infty} \text{Cov}\{X_t X_{t+n}\} = \infty, \quad (2.5)$$

otherwise, it is called short-range-dependent. On the other hand, a process X_t is called strictly self-similar with a self-similarity parameter, or *Hurst parameter*, H if for any $\alpha > 0$, the processes $X_{\alpha t}$ and $\alpha^H X_t$ have the same finite-dimensional distributions. This was first observed in Ethernet LAN traffic [82]. Several models were proposed since then exhibiting these properties in a search to find a better fit to real traffic. Some of these models include the *fractional Brownian motion*, FBM, process and the *fractional autoregressive integrated moving average*, F-ARIMA, process which are mainly used to model VBR video traffic and sometimes used as limit traffic models to model a large number of ON-OFF sources. Other such models include the *chaotic map* models which make use of the possible fractal-like behavior of non-linear systems and the *quasi-self-similar MMRP* models where the transition probabilities of the modulating chain are chosen in such a way that correlation is introduced over a sufficiently wide range of time scales.

5. Bounded Traffic Models

A *bounded* traffic model of a traffic stream specifies only a few parameters instead of trying to give an accurate description of the connection's cell stream. It defines some deterministic or statistical bounds on the number of cells sent in a given time interval. This approach is obviously not optimal, however,

it is efficient in end-to-end studies and connection admission control studies. Such models include the *calculus of Cruz* model, the statistical *exponentially bounded burstiness*, EBB, model and the *stochastic bounding-interval-dependent*, S-BIND, model.

2.3.2 Modeling of Queueing Systems

A queueing system is concisely described by the *Kendall notation*. This is in the form $A/B/c/K$, where A describes the arrival process, B the service time distribution, c the number of servers, and K the system maximum capacity. If K is omitted, the capacity is infinite. The symbols for A or B are standard. Thus, $M/M/1$ is an infinite buffer system with one server, the packets arrive according to a Poisson process, and the service time is exponentially distributed, where M stands for Markovian [70]. For constant length packets such as ATM cells, $\cdot/D/1/K$ is often appropriate.

When the arrival process delivers work faster than the system capacity defined by the service time, queueing systems kick in to process cells one by one. A solution of the steady-state queue occupancy distribution allows us to derive useful design information such as the appropriate queue size for a certain cell loss probability. Another important feature of the queueing system is the order of serving the queued packets; this is called the *service discipline*. The most popular service discipline is *first come first serve*, FCFS, because of its implementation simplicity. However, guaranteeing the QoS while keeping the resource utilization high may require other service disciplines. Even non-work-conserving ones, where there might be times the

server is idle even though there are some packets still in the buffer, might be used if that would make the outgoing traffic less bursty for example.

2.3.3 Analytical Solution Methods

Once a traffic source model is setup and a switching architecture is identified, the queueing model can be established and then solved. The major problem to be solved is the determination of the buffer dimension or size that satisfies certain performance conditions. Usually, an infinite buffer is assumed in the mathematical model which is then solved to find the steady-state or equilibrium probability distribution of the buffer occupation to exceed a given level. Several techniques for finding this solution exist. The simplest of which is the Monte Carlo simulation which is usually used to get quick insights or to find some reference solution to compare analytical solutions to. However, simulating complex models to obtain very low probabilities is a very slow process. Some methods, such as the *rare events* method, try to extrapolate the results when the overall behavior is known in advance. Analytical techniques cover a broad range of models and complexities. Following is a brief discussion of some of the most important methods of these.

1. *Generating Function Transform*

This technique is based on the transformation of a *probability mass function*, pmf, in to the complex plane to get what is known as the *probability generating function*, PGF. Complex analysis techniques can be applied on the probability generating function before transforming it back to a probability distribution.

If a discrete probability distribution is given by $\Pr[A = k] = a_k, k = 0, 1, \dots,$

then its z -transform is defined as

$$A(z) = \sum_{k=0}^{\infty} a_k z^k, \quad (2.6)$$

and the inverse transform can be found from the relation

$$\frac{1}{n!} \left. \frac{d^n A(z)}{dz^n} \right|_{z=0} = a_n. \quad (2.7)$$

For broadband networks, it is necessary to obtain the whole distribution. Finding the inverse transform is straightforward for some generating functions but complicated for most of them. Other ways to find the inverse include finding the series expansion for the PGF and methods based on Fourier series. For sufficiently large n , partial fraction expansion is used and only the term involving the smallest zero of the denominator is inverted.

2. Fluid Flow Approximation

This approach is due to Anick, Mitra and Sondhi [3] in which the arrival process is modeled by a modulated Markov chain and the service time is assumed to be constant. This arrival process, $A(t)$, is modeled as a fluid with a rate, a_j , dependent on the CTMC modulator phase $j = 0 \dots N$. Hence, the rate of change of the buffer occupation, r_j , is given by

$$r_j = a_j - c. \quad (2.8)$$

Assuming a infinite buffer size, the steady-state overflow probability beyond a given level x follows

$$G(x) = \lim_{t \rightarrow \infty} \Pr[X(t) > x], \quad (2.9)$$

where $X(t)$ represents the buffer occupancy at time t . $F_j(x)$ is defined to be the probability that the buffer occupancy level is less than or equal to level x when the system is in equilibrium and the source modulator is in phase j . If $\mathbf{F}(x)$ is a row vector of $F_j(x)$ at all modulator phases, the steady-state overflow probability can be written as

$$G(x) = 1 - \Pr[X \leq x] = 1 - \mathbf{F}(x) \cdot \mathbf{1}, \quad (2.10)$$

where $\mathbf{1}$ is a unit column vector. After some manipulations, they showed that

$$\frac{d}{dx} \mathbf{F}(x) \cdot \mathbf{R} = \mathbf{F}(x) \cdot \mathbf{Q}, \quad (2.11)$$

where \mathbf{R} is the diagonal rate matrix and \mathbf{Q} is the infinitesimal generator matrix of the CTMC of the arrival process. The solution of this linear first-order differential equation is a linear combination of exponentials of the form $e^{z_i x}$ where z_i , $i = 0 \dots N$ are the eigenvalues of $\mathbf{Q}\mathbf{R}^{-1}$. In some special cases, this technique leads to elegant exact analytical solutions. However, it is an inherently approximate analysis of communication systems since it is based on the assumption of continuous cell generation.

3. *Matrix-Geometric Method*

The development of this method started with Neuts' [102] efforts to provide an alternative solution technique which avoided the sometimes difficult problem of numerically searching for the roots of a usually transcendental equation. The solution of the GI/M/1-type Markov chains was distinguished from that of the M/G/1-type. The former solution tend to be an elegant matrix generalization of the geometric distribution while the latter solution is usually more complicated. The transition probability matrix of such chains has a staircase structure known as an *upper Hessenberg* matrix. For modulated Markov chains, phase transitions are assumed to occur at the beginning of each time slot and a bivariate process of the queue occupancy and the modulator phases has to be used to have a Markovian chain. The probability transition matrix is then block partitioned into an *upper block Hessenberg* matrix with the modulator transition matrix taken into account. The key ingredient to the matrix analytic solution is a matrix functional equation, G , related to the fundamental period of the queue. The roots of the traditional analysis are the eigenvalues of the matrix G and they can still be easily computed even when some of them are close or identical. In the context of BMAP/G/1 queue, G is related to the busy period of the queue and indirectly to the behavior of the arrival process during successive idle periods of the queue. More details on this method will be presented in chapter 4.

2.3.4 Performance Metrics

The performance of a switching fabric is usually evaluated based on three measures: throughput, delay, and cell loss probability. Typically, a packet switching fabric, used in a system that guarantees a certain QoS, is required to support a high throughput, a small delay, and an extremely low cell loss probability.

- *Throughput (γ)*. It is defined as the average number of cells which are successfully delivered by the switch per time slot per input line. The metric γ_{\max} is the value of γ under maximum load conditions. While γ_{\max} is an important performance measure, it will not be directly felt by network users.
- *Switch Delay (D)*. It is defined as the average time, in time slots, a cell spends from the time it arrives at the input port, till the time it is successfully delivered to its requested output line. D includes the time spent in any input, internal, and/or output buffers. The end-to-end delay, which includes the delay of individual switching nodes, will be experienced by network users. Also, jitter should be very small in an ATM switch.
- *Cell Loss Probability (CLP or P_L)*. It is defined as the fraction of cells lost within the switch as a result of blocking or buffer overflows. Cell retransmission takes place on an end-to-end basis in ATM networks and because of the high speeds involved, P_L is considered a very important performance measure.

2.4 Buffering Schemes

Due to the statistical nature of the traffic, buffering in any packet switch is unavoidable. Consider a generic non-blocking space-division packet switch, without regard to its internal structure except for the assumption that it has no internal buffers. The basic schemes are input buffering, internal buffering and output buffering. Internal buffering is architecture-dependent, but generally speaking it is not desirable for one or more of the following reasons :

1. In multipath architectures, cells may be delivered out-of-sequence.
2. It complicates the internal design of the switching elements.
3. It complicates fault-diagnosis testing if a cut-through mechanism is implemented.
4. It may introduce random delays within the switch fabric causing undesirable cell delay variation or jitter.

Any two of the three basic buffering schemes can be jointly adopted in a switch to produce a mixed buffering scheme. Whenever mixed buffering is used, two different transfer modes can be adopted [110]:

- *Backpressure*. Through utilization of a suitable backward signaling, the number of cells actually switched to each downstream buffer, internal or output, is limited to the buffer's current storage capability. In this case, all the other *head-of-line*, HOL, cells remain stored in their respective upstream buffer, input or internal.

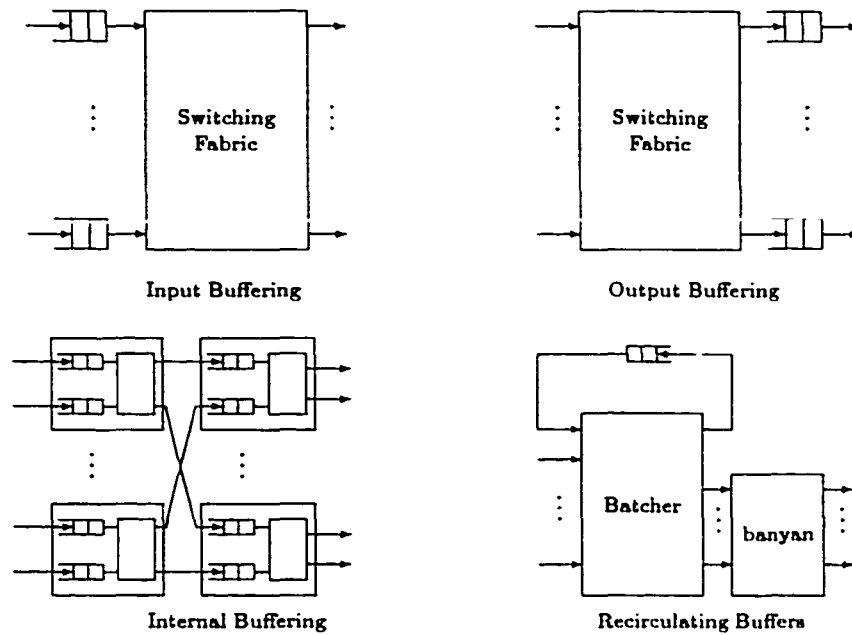


Figure 2.5: Examples of different buffering schemes.

- *Queue Loss.* Cell loss takes place in the downstream buffer for those HOL cells that have been transmitted by the upstream buffers, but cannot be stored in the addressed downstream buffers.

2.4.1 Output Buffering

Assuming an output non-blocking switch as shown in figure 2.5 where all cells arriving in a given time slot can be cleared to the output side within the slot, even if all cells arriving at the N inputs are destined to the same output port. This can be achieved by, say, speeding-up the switching fabric by a factor N . However, the input lines and output lines operate at the same speed and the output line can

serve only one cell in each time slot. The other cells have to be buffered at the output. If the average utilization of an output line is the same as that of an input line, then the system is stable. Assuming a uniform traffic, an infinite buffer size $B_{\text{out}} = \infty$ and $N = \infty$, the average queuing delay D_{avg} is the same as that of an M/D/1 queue [65]:

$$D_{\text{avg}} = \frac{p}{2(1-p)}, \quad 0 \leq p < 1 \quad (2.12)$$

Output buffering has the best throughput-delay performance under uniform traffic and it lends itself naturally to multicasting and broadcasting, however, it has its own complexities. The switch fabric must be speeded-up to transfer all incoming cells in any given time slot to the output side with that time slot and the memory used in the buffers must have a minimum bandwidth of $(N + 1)V$ bits per second, where V is the bit rate of the external lines. Memory speed can be reduced using bit-slicing techniques or output concentration [128].

2.4.2 Input Buffering

In an input buffering switch, arriving cells enter a *first-in first-out*, FIFO, buffer at the input port as shown in figure 2.5. In each time slot, the switch resolves contentions prior to switching. If all HOL cells are destined to distinct output ports, then all of them are switched. However, if K HOL cells, $1 < K \leq N$, are destined to the same output port, only one cell is switched to that port while the other cells wait for the next time slot. Several HOL arbitration schemes, also called contention resolution mechanisms or selection policies, have been proposed

and studied in the literature [5]. These include random selection, global FIFO, longest queue, oldest queue, round-robin and others.

Input buffering switches with FIFO queueing suffer from the *HOL blocking* which occurs when a cell is waiting for its turn to be switched to an output port and other cells are blocked behind it despite the fact that their destination ports are possibly idle. The relationship between the average queueing delay D_{avg} and the arrival rate p for an infinite input buffer $B_{\text{in}} = \infty$ and $N = \infty$ and random selection is according to the following equation:

$$D_{\text{avg}} = \frac{(2-p)(1-p)}{(0.586-p)(3.414-p)} - 1, \quad 0 \leq p < 0.586 \quad (2.13)$$

Consequently, the maximum throughput is limited to $2 - \sqrt{2} \approx 0.586$ regardless of the selection policy [65] which also represents the saturation value of p above which steady state queue sizes grow without limit. Removing the input buffers and dropping cells upon contention would increase the maximum throughput to ≈ 0.632 at the expense of high cell loss probabilities which are unacceptable for ATM switches [65]. HOL blocking can be reduced using structural techniques such as *switch expansion*, *windowing*, or *channel grouping* techniques [110].

2.4.3 Combined Input-Output Buffering

Within the past few years, great interest has been shown in input-output buffering switches built around non-blocking fabrics, either using the queue loss [26, 111] or the backpressure [5, 67, 105, 111] mechanism. Table 2.1 shows that many of the state-of-the-art designs use input-output buffering and the backpressure mecha-

nism. Some of the main observations about input-output buffering non-blocking switches include the following [64, 111]:

- For a fixed total buffer budget, there exists an optimal placement of buffers among input and output ports to minimize P_L .
- The backpressure mechanism requires less $B_{\text{in}} + B_{\text{out}}$ to achieve a given P_L under a very wide range of load values.
- With backpressure, the effect of B_{out} is more dominant on the performance than that of B_{in} .
- Buffering cells at the input ports results in a significant reduction in the fabric speed-up needed to attain a given P_L at a given load value.
- Under Poisson arrivals, a higher γ_{max} is achieved with fixed-length packets compared to that achieved with packets having exponentially-distributed lengths.

2.5 Classification of Switching Fabrics

Traditionally switching has been defined to encompass either space switching or time switching or combinations of both techniques. The classification adopted here is slightly different in the sense that it divides the design approaches under four broad categories, namely, shared memory, shared medium, fully interconnected and space division. For simplicity, the ensuing discussion will assume a switch with N input ports, N output ports, and all port speeds equal to V cells/sec.

2.5.1 Shared Memory Approach

Figure 2.6 illustrates the basic structure of a shared memory switch [28]. In this structure, incoming cells are converted from serial to parallel form, and written sequentially to a dual port Random Access Memory. A memory controller decides the sequence in which cells are read out of the memory, based on the cell headers with internal routing tags. Outgoing cells are demultiplexed to the outputs and converted from parallel to serial form. This approach is an output queueing ap-

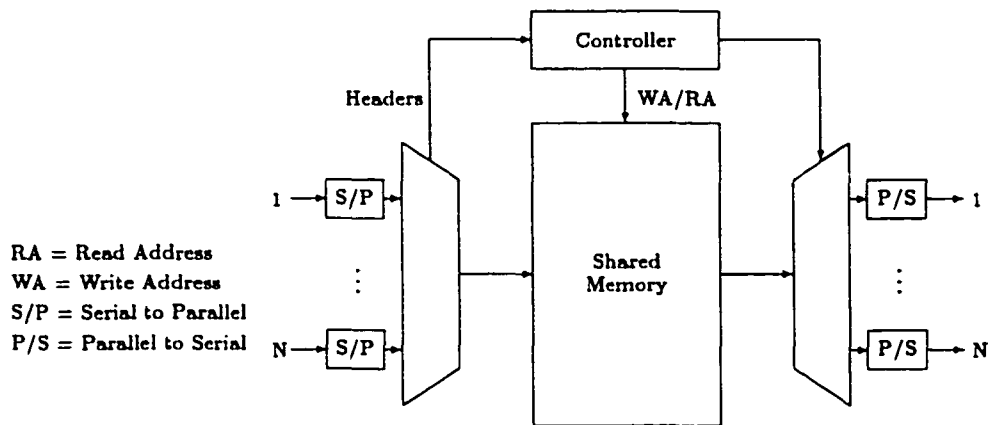


Figure 2.6: Basic structure of a shared memory switch.

proach, where all the output buffers physically belong to a common buffer pool. The approach is attractive because it achieves 100% throughput under heavy load. The buffer sharing minimizes the amount of buffering needed to achieve a specified cell loss rate. This is because if a large burst of traffic is directed to one output port, the shared memory can absorb as much as possible of it. CNET's Prelude switch was one of the earliest prototypes of this technique, which employed slotted opera-

tion with packet queueing. Hitachi's shared buffer switch and AT&T's GCNS-2000 are famous examples of this scheme [28]. The approach, however, suffers from a few drawbacks. The shared memory must operate $2N$ times faster than the port speed because cells must be read and written one at a time. As the access time of memory is physically limited, the approach is not very scalable. The product of the number of ports and port speed, $2NV$, is limited. In addition, the centralized memory controller must process cell headers and routing tags at the same rate as the memory. This is difficult for multiple priority classes, complicated cell scheduling, multicasting and broadcasting.

2.5.2 Shared Medium Approach

Cells may be routed through a shared medium, like a ring, bus or dual bus. Time-division multiplexed buses are a popular example of this approach and figure 2.7 illustrates their structure. Arriving cells are sequentially broadcast on the shared bus in a round-robin manner. At each output, address filters pass the appropriate cells to the output buffers, based on their routing tag. The bus speed must be at least NV cells/sec to eliminate input queueing. The outputs are modular, which makes address filters and output buffers easy to implement. Also the broadcast-and-select nature of the approach makes multicasting and broadcasting straightforward. As a result, many such switches have been implemented, such as IBM's Packetized Automated Routing Integrated System (PARIS) and *planET*, NEC's ATM Output Buffer Modular Switch (ATOM), and Fore Systems' ForeRunner ASX-100 to mention a few [28]. However, because the address filters and output buffers must

operate at the shared medium speed, which is N times faster than the port speed, this places a physical limitation on the scalability of the approach. In addition, unlike the shared memory approach, output buffers are not shared, which requires more total amount of buffers for the same cell loss rate.

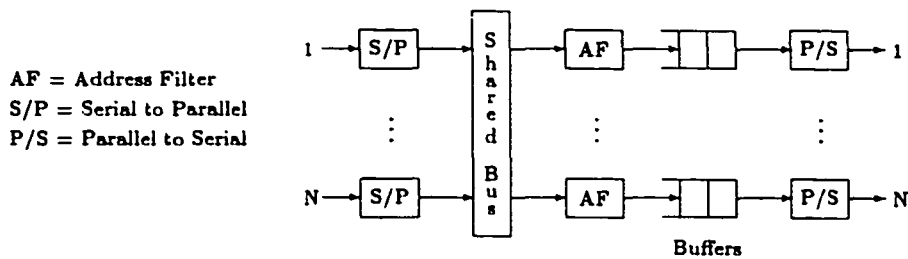


Figure 2.7: A shared bus switch.

2.5.3 Fully Interconnected Approach

In this approach, independent paths exist between all N squared possible pairs of inputs and outputs. Hence, arriving cells are broadcast on separate buses to all outputs and address filters pass the appropriate cells to the output queues. This architecture is illustrated in figure 2.8. This design has many advantages. As before, all queueing occurs at the outputs. In addition, multicasting and broadcasting are natural, like in the shared medium approach. Address filters and output buffers are simple to implement and only need to operate at the port speed. Since all of the hardware operates at the same speed, the approach is scalable to any size and speed. Unfortunately, the quadratic growth of buffers limits the number of output ports for practical reasons. However, the port speed is not limited except

by the physical limitation on the speed of the address filters and output buffers. The *knockout switch* developed by AT&T [140] was an early prototype where the amount of buffers was reduced at the cost of higher cell loss. Instead of N buffers at each output, it was proposed to use only a fixed number of buffers L for a total of $N \times L$ buffers. This technique was based on the observation that it is unlikely that more than L cells will arrive for any output at the same time. It was argued that selecting an L value of 8 was sufficient for achieving a cell loss rate of 10^{-6} under uniform random traffic conditions for large values of N .

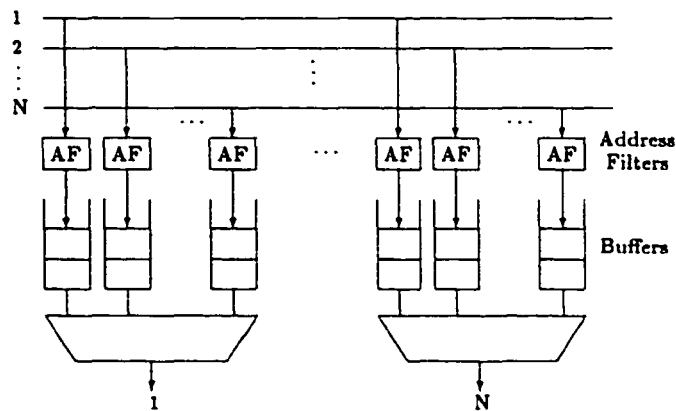


Figure 2.8: A fully interconnected switch.

2.5.4 Space Division Approach

The *crossbar* switch is the simplest example of a matrix-like space division fabric that physically interconnects any of the N inputs to any of the N outputs. It can be used to achieve very high aggregate rates using input/output buffering and a bidirectional arbitration algorithm. *Multistage interconnection networks*, MINs, which

are more tree-like structures, were developed to reduce the N^2 crosspoints needed for circuit switching, multiprocessor interconnection and, more recently, packet switching. MINs come in different forms; such as Clos networks, Bencš networks and banyan networks. Banyan networks have many variations; such as Baseline, Omega, Indirect Binary n -Cube, Shuffle and Delta networks. These networks were shown to be topologically equivalent and most ATM switching researchers tend to freely interchange their names, also justified by the fact that they have the same performance under uniform traffic [5].

In general, to construct an $N \times N$ banyan network, the n^{th} stage uses the n^{th} bit of the output address to route the cell. The banyan will consist of $n = \log_2 N$ stages, each consisting of $N/2$ switching elements as in figure 2.9. A MIN is said to be *self-routing* when the output address completely specifies the route through the network. The banyan network technique is popular because switching is performed by simple switching elements, cells are routed in parallel, all elements operate at the same speed and there is no additional restrictions on the size N or the speed V , large switches can be easily constructed modularly and recursively. Many proposals used more than one banyan network for the fabric. They were used in parallel planes, in tandem and pipelined. The throughput performance of a banyan network can be improved by replacing each internal link by more than one; this is the *dilated* banyan network [5, 128].

A banyan or an Omega network becomes non-blocking if incoming cells are ordered according to their output port addresses, and concentrated, provided that output conflicts do not exist [57]. This is the basic idea behind all sort-banyan based

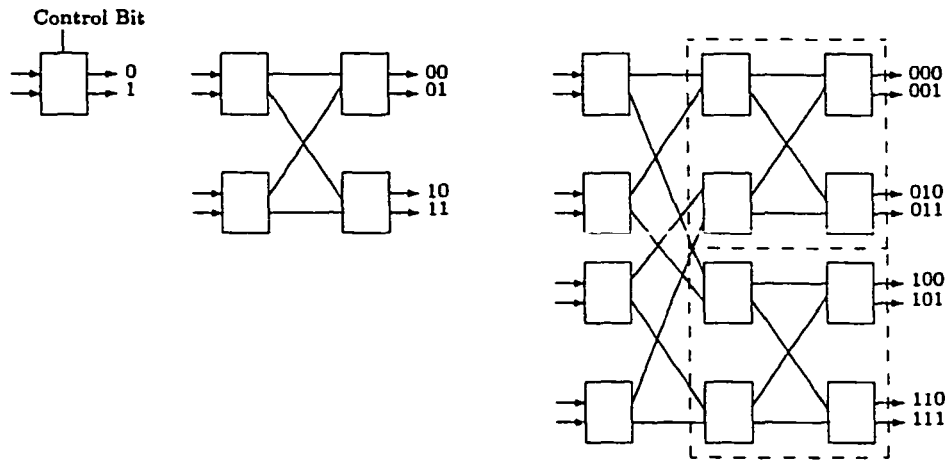


Figure 2.9: A switching element, a 4×4 banyan and an 8×8 banyan switching fabric.

ATM switches. The Batcher network which is based on the bitonic sorting principle, can sort an arbitrary set of cells according to their output port addresses. Thus, the combination of a Batcher network and a banyan or Omega network solves the internal blocking problem. The *starlite* switch sort-banyan based switching fabric was the first such fabric to be proposed in the literature. The *sunshine* switch combines a Batcher network, a trap network to resolve output conflicts by selecting the K highest priority cells for each output, a concentrator which is another Batcher network to recirculate unused cells for queueing and finally up to K cells per output port are switched within a given time slot, each through a different banyan network. It was also shown to have extremely low P_L for circuit switching. The *starlite* switch is the same as the *sunshine* switch with $K = 1$. Many other variations were proposed such as the input-output buffered *duplex* switch, Lee's modular switch, the output buffered *Shared Concentration Output Queueing* switch [5, 128].

2.6 Summary

In this chapter, an overview of the main categories of ATM electronic switching architectures was presented along with an introduction to the analytical tools and measures used to evaluate and compare these architectures. It is obvious that some of these architectures have gained more research interest because of the availability of the technology required to implement them while others, such as the shared memory architecture, were neglected because of the technology constraints in spite of their known advantages. Hence, more research is required to evaluate and compare the shared memory architectures with respect to different integrated services traffic environments. It is also worth noting that real-life situations do not necessarily agree with the traffic uniformity assumption. Also, it is important to extend the comparison beyond performance, and consider other factors such as implementation complexity, cost, fault tolerance and reliability.

Table 2.1: Survey of the state-of-the-art ATM fabrics in the recently published literature

Year ^a [Ref.]	Affiliation	Fabric Type	Size $I \times O$	Ports Rates	Buffering Scheme	Queue Management	Multicast Support ^b	Priority	Considering Scalability	Performance Evaluation ^c	Implemen. & Technology
1997 [31]	Bell Labs., Lucent, USA	ATLANTA: Memory/space- division/memory	8 x 8 Building blocks	5 Gbits/s aggregate	Selective backpres- sure input- output		U/M/B		Upto 25 Gbits/s aggregate		CMOS
1997 [67]	ASSICOM Project, European Union	ATLAS I: Multistage crossbars	16 x 16	622 Mbit/s	Multilane backpres- sure input- output		U/M/B		20 Gbit/s aggregate	$\gamma_{max} = 0.96$, $D_{avg} = 57$	0.35 μm CMOS
1997 [138]	Mitsubishi, Japan	Funnel or concentrating structure	32 x 8 with 1 chip-set or 32 x 32 with 4 chip-sets	622 Mbits/s	Shared buffering	Searchable- address FIFO	U/M/B	Delay & loss	Yes	$P_L = 10^{-6}$	Chip-set of 2 chips using 0.5 μm CMOS
1997 [52]	Univ. of Toronto, ON	RAM-based			Shared out- put buffer- ing	per-VC Scheduling			Building block		
1997 [115]	Siemens, Germany	MainStrt Xpress: Funnel struc- ture	32 x 16	20 Gbit/s aggregate	Backpressure input- output	per-VC Sta- tistical mux- ing			Upto 1 Tbit/s aggregate		ASICs
1997 [106]	Boğaziçi Univ., Turkey	PIPN: Parallel intercon- nected parallel banyan net- works	256 x 256	155 Mbit/s	Input- output buffering	FIFO				$\gamma_{max} = 0.327$	0.1 μm CMOS
1997 [44]	NTT, Japan	TORUS: Multi- stage circular interconnec- tion	4 x 2	2.4-10 Gbits/s	Input- output buffering				Upto 1 Tbits/s aggregate	$P_L = 10^{-6}$	0.8 μm Si- bipolar
1997 [26]	Polytechnic Univ., NY	ABACUS: Grouping crossbar core and output switching modules	32 x 32		QL input- output buffering		U/M/B	Many types	Upto 1024 x 1024	$\gamma_{max} = 0.88$, $P_L = 10^{-6}$, $D_{avg} = 58.9$	0.8 μm CMOS Triple metal
1997 [70]	Univ. of Toronto, ON	PINIUM: Knockout variation	32 x 16 Build- ing blocks	155 Mbit/s	Output buffering		U/M/B	Sorting	Upto 160 Gbit/s aggregate	$P_L = 10^{-6}$	
1997 [29]	Nanyang Technical Univ., Singapore	Multistage sorting and expansion		150 Mbit/s	Output buffering	Multichannel Round Robin			Yes	$P_L = 10^{-9}$	

^a More recent proposals are concentrating on adding more features to the fabric and selecting the design parameters rather than proposing new architectures.

^b U/M/B stands for Unicast/Multicast/Broadcast.

^c These performance evaluations did not use the same load, traffic and architectural conditions.

Table 2.1: Survey of the state-of-the-art ATM fabrics in the recently published literature (cont.)

Year ^a [Ref.]	Affiliation	Fabric Type	Size $I \times O$	Ports Rates	Buffering Scheme	Queue Management	Multicast Support ^b	Priority	Considering Scalability	Performance Evaluation ^c	Implemen. & Technology
1997 [137]	Oki Electric, Japan	Batcher banyan	8 x 8	2.6 Gbits/s							0.5 μm GaAs MESFET in a multichip module
1997 [96]	UCSD and Rockwell, CA	Demux/Mux crossbar	12 x 12	0.12 Tbits/s aggregate							AlGaAs/GaAs HBT
1997 [104]	GARDEN Project: European Union	Demux/Mux Crossbar	2 x 2	10 Gbits/s	Internal buffering						0.3 μm HEMT
1996 [21]	UCSB, CA	Thunder and Lightning: Crossbar	4 x 4	10 Gbits/s	Internal shared buffering	FIFO			Upto 100 Gbits/s aggregate		8 full-custom HBT chips
1996 [68]	Univ. of British Columbia, BC	Truncated fat-banyan tree	16 x 16		Internal buffering	FIFO			Yes	$P_L = 10^{-8}$	0.8 μm Bi-CMOS
1996 [109]	Watson Research Center, IBM, NY	Shift register ring	16 x 16	1 Gbit/s	Shared output buffering		U/M/B		Yes	$P_L = 10^{-8}$	0.5 μm CMOS
1996 [80]	Royal Institute of Technology, Sweden	Banyan network	2 x 2	10 Gbit/s	Internal buffering						Vitesse GaAs MESFET
1995 [42]	AT&T, NJ	Growable knockout structure	8 x 8	2.5 Gbit/s	Output buffering				Upto 160 Gbit/s aggregate		0.5 μm CMOS
1995 [4]	Queens's Univ., ON	MS4: Multi single-stage-shuffling			Output buffering				Yes	Arbitrary small P_L	
1995 [87]	UIUC, IL	iPOINT: Parallel shift register rings	4 x 4	800 Mbit/s aggregate	Input buffering				Upto 128 Gbit/s aggregate		PPGA
1995 [118]	TriQuint, OR	Crossbar	16 x 16	40 Gbit/s aggregate							GaAs MESFET

^a More recent proposals are concentrating on adding more features to the fabric and selecting the design parameters rather than proposing new architectures.

^b U/M/B stands for Unicast/Multicast/Broadcast.

^c These performance evaluations did not use the same load, traffic and architectural conditions.

Chapter 3

Shared Memory Switching Fabric

3.1 Introduction

In this chapter, system requirements for switching fabrics on-board satellites are identified and discussed. To meet those requirements, a shared memory switching fabric architecture is proposed. This proposal is preceded by an overview of the main categories of shared memory architectures. Moreover, this chapter discusses how this proposed architecture makes use of the shared memory advantages and how it overcomes its disadvantages. Results from this chapter have appeared in [131,132].

3.2 Requirements for On-Board Satellite Systems

In order for satellite networks to offer competitive services, issues related to multiple access, protocol enhancement and on-board processing have to be addressed. In

this work, on-board processing is the main concern. Typical system requirements must be identified so that we can proceed to select the appropriate architecture and implementation. Typical requirements used in determining a suitable switching architecture in terrestrial networks are capacity and port rates, in addition to performance criteria such as throughput, cell loss rate and average cell transfer delay. For satellite systems, however, additional requirements exist. These requirements include those imposed by the limited resources on-board, by the harsh environment and by the need to adapt to new services. The requirements due to the limited resources can be summarized in the following two points.

- Satellites have a limited payload in terms of size and mass. This requires minimizing the size of on-board processing systems including switching architectures. This in turn translates into a smaller silicon and printed boards area, fewer chips and simpler and smaller packaging and cooling systems.
- Limited on-board energy resources translates into the requirement of low-power dissipation in VLSI architectures. This can be achieved by using power reduction techniques on both the logical and architectural levels. Low-power dissipation also reduces the cooling requirements and reduces the need for advanced chip and system packaging techniques that allow for enough thermal dissipation while meeting the payload requirements.

These requirements would normally be addressed on a design-by-design basis and in an ad-hoc manner that involves designers expertise at different levels. In this work, they will be addressed by proposing a very simple architecture that can be implemented in as few chips as possible and using some of these ad-hoc methods as

will be shown throughout this chapter and chapter 5. Furthermore, a generalized framework is proposed in chapter 4 to formulate this problem as an optimization problem and to solve it using numerical techniques in order to provide adequate performance and at the same time satisfy the above physical requirements.

Satellites operate in a hostile environment. Aside from the mechanical vibrations that occur during launching the satellites which must be accommodated by proper mechanical design, radiation in the space environment is a major factor to consider. It has two primary effects on electronics in space. One is *total dose*, the accumulation of radiation over time, which results in permanent degradation of device performance including shifts in turn-on or threshold voltages, increases in operating and stand-by currents and changes in signal propagation delays. Trapped electrons and protons, which comprise Van Allen belts that surround the earth above the atmosphere, contribute the bulk of the total dose damage. Heavy ions contribute to single-event upsets, the second main effect of radiations on electronics in space [12]. Single-event upsets are generally soft errors, which means that although the data in the upset cell is permanently lost, the cell can be still rewritten with new data with no change in its operating characteristics. Single-event latch-ups, on the other hand, can cause both a soft and a hard failure. The soft failure can be repaired by powering the unit off and on again. However, latch-up may also cause parasitic currents large enough to burn out the junctions and permanently destroy the integrated circuit. The requirements imposed by such conditions can thus be summarized as follows.

- The use of radiation-hardened technologies for applications such as satellite switches is required. Radiation hardening is a fabrication process issue and will not be discussed any further. It is sufficient to mention that rad-hard integrated circuits can now be manufactured on dedicated wafer fabrication lines. Also available are innovative commercial self-contained processes that are being used recently to provide for the thousands of satellites going into low earth-orbits in the next few years including even some FPGA technologies [35] and some special shielding packages with enough mass to reduce the radiation inside to tolerable levels [12].
- The satellite and its payload usually have long operating lifetimes during which repairs can rarely or almost never be done. The system must exhibit a very high reliability and all single-point failures must be avoided. Subsequently, fault tolerance should be provided by introducing error and fault detection and redundancy. Some architectures are simpler than others to make them fault tolerant, based on their switching mechanism. Redundant switching paths and spare components are better for space division switches while error control codes can be used to detect or correct and isolate faults in memory based switches [46]. Fault tolerance can also provide a way to relax the rad-hard technology constraints through component swapping and can provide as well as share some of the system self-testing capabilities. On the system level, redundancy in the satellite constellation as a whole will also allow for rapid system recovery in the event of a loss of a whole satellite and the switch-over to another redundant one. Fault tolerance in the proposed

architecture will be discussed in section 3.4.3.

Other requirements that are also essential for efficient architectures on-board satellites include:

- Another consequence of the long operating lifetime is that the system should be remotely upgradeable. Advances and changes in QoS provision, scheduling, buffer management and priority handling algorithms as well as other communication protocols such as multiple access, CAC and OAM protocols should be implemented easily through remote software upgrades. This means that programmability should be of great concern while designing all control architectures. Chapter 5 addresses this to some extent by designing a flexible buffer management unit capable of supporting different scheduling and priority handling algorithms.
- Scalability and modularity are not of great concern for satellite switching fabrics since they have different design drivers. Specifically, they need to be simplified and optimized. Also, the satellites are generally irrecoverable and later upgrades can not be economically done specially for large constellations. However, the satellite system as a whole need to be designed with enough capacity to support the expected exploding amount of traffic. Therefore, a scalable switching fabric need to be designed and installed in all satellites from the beginning based on the most aggressive expectations for the network capacity. This will be discussed in section 3.4.2 as well as in chapter 4.

- Delay is an important issue to consider for satellite switching systems with respect to performance requirements. Depending on the orbit, the switching delay will have a varying effect on the overall delay, propagation and switching. For example, the one-way propagation delay for a geosynchronous orbit is about 125 msec. The on-board switching delay, including queuing delay, is negligible compared to the propagation delay. However, the cell loss probability becomes increasingly important, 10^{-10} or less is a typical value, because of this delay. Any retransmissions will incur large delay because of the inherent latency of automatic repeat request, ARQ, systems. To avoid large retransmission delays, due to cell loss within the switch, it may be necessary to introduce larger buffers than those typically in terrestrial switches. Predictive methods have to be used to limit on-board congestion instead of feedback or external back-pressure methods [2].

3.3 Switching Fabric Architectures

In order to meet the requirements discussed above along with the performance requirements of the network, the switching fabric architecture proposed is based on a special shared memory architecture. It also makes use of newly available technologies such as very high speed memories and advanced semiconductor technologies as well as special design methodologies such as low-power VLSI design techniques.

3.3.1 Shared Memory Architectures

It was shown in chapter 2 that the shared memory approach is not preferred because of the high memory access rates and the high complexity of the centralized controller. Very few proposals [21, 138] of those in table 2.1 are based on shared memory. However, the shared memory approach has the advantages of higher memory utilization and easier queue management along with multicasting and priority capabilities as summarized in table 3.1. Also, it can be considered an output buffering scheme which guarantees the switch throughput. Moreover, it is also known that the shared memory approach has the least cell loss rate of all buffering schemes for the same buffer size [53].

Table 3.1: The main features of the different fabric design approaches.

	Shared Memory	Shared Medium	Space-Division
Fabric Speed ¹	$2NV$	NV	V
Fabric Constraints	memory speed	bus speed	size
Controller Complexity	high, centralized	moderate, centralized or distributed	low, distributed
Memory Utilization	high	low	low
Multicasting Capability	easy	easy	difficult
Priority Classes Capability	easy	difficult	difficult

¹ N is the number of input or output ports, and V is the port rate.

Few shared memory architectures have been proposed in the past and were neglected due to technology limits. Now, advances in memory technology [39, 59, 66] moved past the barrier of memory bandwidth as will be discussed in section 3.4.1 and some of these architectures deserve to be revisited, such as:

- Linked-List Based Shared Memory Switch [74].
- Shared-Buffer Direct-Access Switch, SBDA [72].
- Hybrid Shared and Dedicated Output Buffer Switch [81].
- CAM-Based Shared Buffer Switch [119].

These architectures are compared in table 3.2 and their basic designs are shown in figures 3.1 to 3.4.

Table 3.2: Comparison of basic shared memory ATM switching architectures.

	Linked-List Based Switch	SBDA Switch	Hybrid Shared Switch	CAM-Based Switch
Representative Design	[74]	[72]	[81]	[119]
Cell Storage	RAM	RAM	RAM	CAM
Queue Maintenance	address lists	linked-list tags	FIFO address buffers	associated tags
Processing	sequential	parallel	sequential	compound
Idle Addresses Storage	IAF extra memory block	separate list of tags	IAF extra memory block	CAM valid bit
Queue Length Checking	additional counters	additional counters	additional counters	comparison of sequence numbers
Broadcast Control	separate queue, additional hardware	extra queue, no hardware	two separate blocks of memory, additional hardware	CAM broadcast bit, no hardware
Priority Control	separate queue and hardware per port	modification of ATM cells	separate FIFO for each priority per port	single CAM priority bit
Cell Access Time ¹	$3T_{wr} + 3T_{rd}$	depends on implementation issues	$3T_{wr} + T_{rd}$	$2T_s + T_{wr} + T_{rd}$

¹ T_{rd} and T_{wr} are the RAM read and write access times respectively while T_s is the CAM searching time.

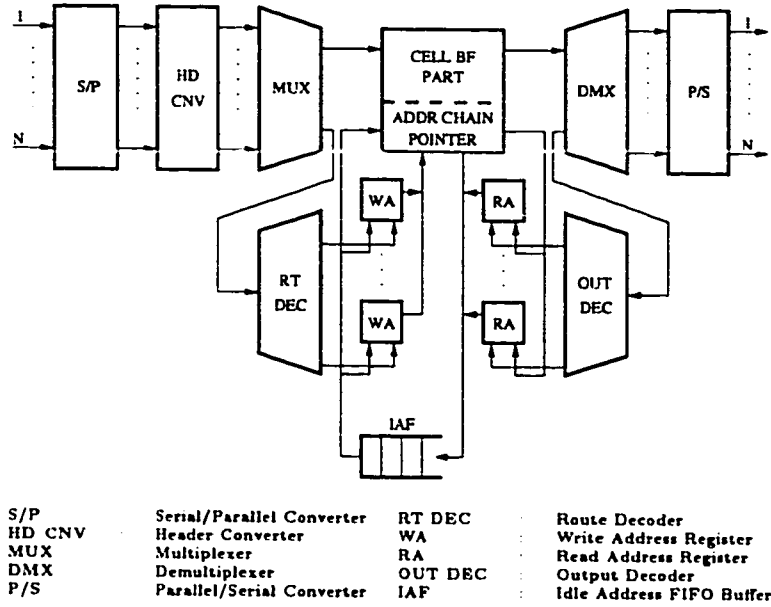


Figure 3.1: Linked-list based ATM switch architecture.

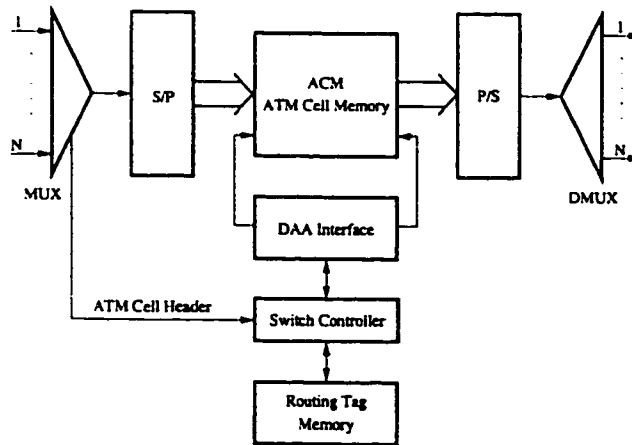


Figure 3.2: SBDA ATM switch architecture.

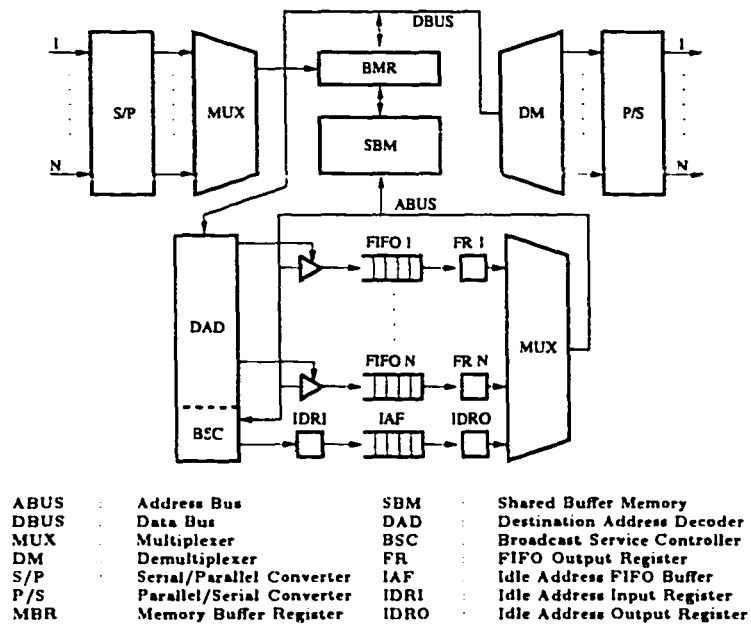


Figure 3.3: Hybrid shared and dedicated output buffer architecture.

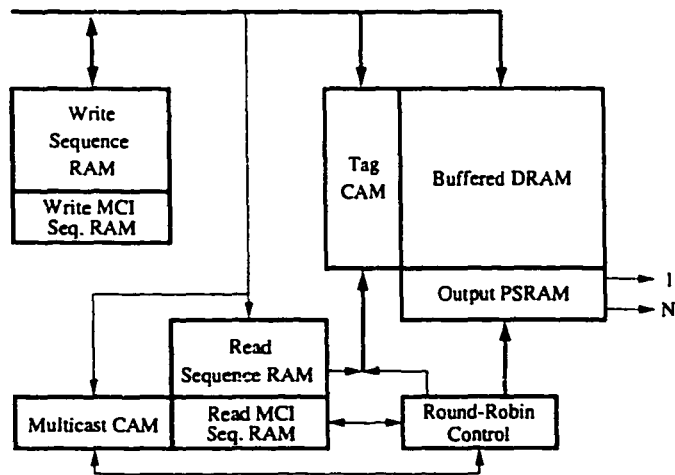


Figure 3.4: CAM-based ATM switch architecture.

3.3.2 Buffer Sharing Effect

An important factor that motivated the use of shared memory architectures is the buffer sharing effect which reduces the size of memory needed in such fabrics to achieve certain cell loss probabilities as compared to the memory size required in other architectures. Broadband networks are required to support large traffic loads under stringent QoS and performance requirements and this is mostly translated into larger buffer requirements. For a fabric in a core switch, such as a satellite switch, we will use the reasonable assumption that the traffic routed through it has already undergone a lot of statistical multiplexing in the buffers of preceding nodes and a random uniform traffic model can be a good approximation. Shared memory buffering was first analyzed under random uniform traffic in [40, 53]. This analysis is used here merely to illustrate the buffer sharing effect. Proper sizing of the buffers will be done using complex traffic models later.

Following the approach in [53] for an $N \times N$ switching fabric under random traffic with an offered load of p , the random variable A is defined as the number of cell arrivals destined for a certain output in a given time slot,

$$a_k \triangleq \Pr[A = k] = \binom{N}{k} (p/N)^k (1 - p/N)^{N-k} \quad k = 0, 1, \dots, N \quad (3.1)$$

which, for $N = \infty$ becomes

$$a_k \triangleq \Pr[A = k] = \frac{p^k e^{-p}}{k!} \quad k = 0, 1, \dots, N = \infty \quad (3.2)$$

Letting Q_m denote the number of cells for the logical queue of the output under consideration at the end of the m^{th} time slot, and A_m denote the number of cell

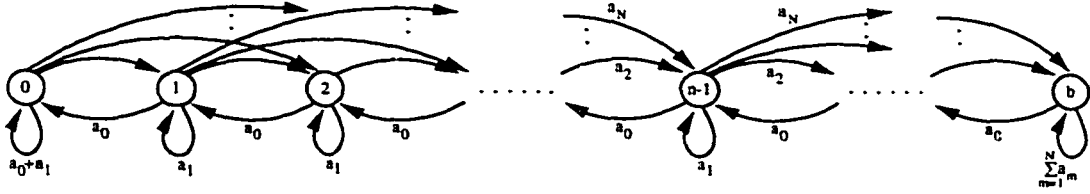


Figure 3.5: State-transition diagram of an output buffer occupancy.

arrivals during the m^{th} time slot, assuming each logical queue can hold up to b cells, then

$$Q_m = \min \{ \max(0, Q_{m-1} + A_m - 1), b \} \quad (3.3)$$

When $Q_{m-1} = 0$ and $A_m > 0$, one of the arriving cells is immediately transmitted during the m^{th} time slot. For $N = \infty$ and $b = \infty$, the queue size Q_m can be modeled by an M/D/1 queue and for finite N and b it can be modeled by a finite-state, discrete time Markov chain with state transition probabilities $P_{ij} \triangleq \Pr[Q_m = j | Q_{m-1} = i]$ given by

$$P_{ij} = \begin{cases} a_0 + a_1 & i = 0, j = 0 \\ a_0 & 1 \leq i \leq b, j = i - 1 \\ a_{j-i+1} & 1 \leq j \leq b - 1, 0 \leq i \leq j \\ \sum_{m=j-i+1}^N a_m & j = b, 0 \leq i \leq j \\ 0 & \text{otherwise} \end{cases} \quad (3.4)$$

The steady-state queue size can be obtained directly from the Markov chain balance

equations to yield the following equation for the very first state

$$q_1 \triangleq \Pr[Q = 1] = \frac{(1 - a_0 - a_1)}{a_0} q_0 \quad (3.5)$$

and by studying state $n - 1$, it yields

$$q_n \triangleq \Pr[Q = n] = \frac{(1 - a_1)}{a_0} q_{n-1} - \sum_{k=2}^n \frac{a_k}{a_0} q_{n-k} \quad 2 \leq n \leq b \quad (3.6)$$

where

$$q_0 \triangleq \Pr[Q = 0] = \frac{1}{1 + \sum_{n=1}^b q_n/q_0} \quad (3.7)$$

Now, if the logical buffers were to share a single physical buffer of size Nb cells and if Q_m^i denotes the number of cells destined for output i in the buffer at the end of the m^{th} time slot, then $\sum_{i=1}^N Q_m^i$ is the total number of cells in the shared buffer at the end of the m^{th} time slot. If the buffer size is infinite, $Nb = \infty$, then

$$Q_m^i = \max \{0, Q_{m-1}^i + A_m^i - 1\} \quad (3.8)$$

where A_m^i is the number of cells addressed to output i that arrive during the m^{th} time slot. With a finite buffer size, cell arrivals destined for some outputs may fill the shared buffer at the expense of other arrivals in the same time slot. The resulting buffer overflow invalidates the above equation, however, it is a good approximation in the region of interest, i.e., the low cell loss probability region, below 10^{-6} . Assuming that the traffic destined for each output port is independent from that destined to the other output ports, the steady-state number of cells in the buffer, $\sum_{i=1}^N Q^i$, can be modeled as the N fold convolution of N M/D/1 queues.

With the assumption of an infinite buffer size, the cell loss probability can then be approximated by $\Pr[\sum_{i=1}^N Q^i \geq Nb]$. It can be shown that the mean value of the shared queue size is $Np^2/2(1-p)$.

A simpler approach to find the cell loss probability is based on the use of *moment-generating functions*. For a random variable x , the moment-generating function, $G_X(z)$, is defined as the expectation of z^x :

$$G_X(z) \equiv E(z^x) = \sum_{x=0}^{\infty} z^x \Pr[X = x] \quad (3.9)$$

and it has the following properties:

$$G_X(1) = 1, \quad (3.10)$$

$$G_X(0) = \Pr[X = 0], \quad (3.11)$$

$$E(x) = \left. \frac{dG_X(z)}{dz} \right|_{z=1} = \sum_{x=0}^{\infty} x \Pr[X = x] \quad (3.12)$$

and if x_1, x_2, \dots, x_N are N independent random variables and y is their sum, $y = \sum_{j=1}^N x_j$, then

$$G_Y(z) = E(z^y) = \prod_{j=1}^N E(z^{x_j}) = \prod_{j=1}^N G_{X_j}(z) \quad (3.13)$$

and using the same recursion used before to describe an M/D/1 queue

$$Q_m = \max\{0, Q_{m-1} + A_m - 1\} \quad (3.14)$$

If we let

$$H_m = Q_{m-1} + A_m \quad (3.15)$$

then

$$Q_m = \begin{cases} H_m - 1 & H_m \geq 1 \\ 0 & H_m = 0, \end{cases} \quad (3.16)$$

$$G_Q(z) = \sum_{q=0}^{\infty} z^q \Pr[Q = q] \quad (3.17)$$

and introducing $\Pr[H = h]$ in place of $\Pr[Q = q]$

$$G_Q(z) = \Pr[H = 0] + \Pr[H = 1]z + \Pr[H = 2]z^2 + \Pr[H = 3]z^3 + \dots \quad (3.18)$$

$$G_Q(z) = \Pr[H = 0] + \frac{G_H(z) - \Pr[H = 0]}{z} \quad (3.19)$$

and using equations (3.15) and (3.16) at steady state, one can write

$$G_H(z) = G_Q(z)G_A(z) \quad (3.20)$$

$$\therefore G_Q(z) = \frac{\Pr[H = 0](z - 1)}{z - G_A(z)} \quad (3.21)$$

and using L'Hospital's rule

$$G_Q(1) = 1 = \frac{\Pr\{H = 0\}}{1 - G'_A(1)} \quad (3.22)$$

Since, the arrivals have a binomial distribution for the uniform random traffic

$$G_A(z) = \left(1 - \frac{p}{N} + \frac{p}{N}z\right)^N \quad (3.23)$$

and $G'_A(1) = E(a) = p = \rho$, ρ being the utilization of the outgoing link. Finally, the M/D/1 queue can be described using the moment-generating function:

$$G_Q(z) = \frac{(1-p)(z-1)}{z - \left(1 - \frac{p}{N} + \frac{p}{N}z\right)^N} \quad (3.24)$$

and using the same assumptions as before, the shared buffer, Q_s , can be approximated by an N fold convolution of N M/D/1 queues, from which the probability of the buffer having n cells can be computed using

$$\Pr\{Q_s = n\} = \frac{1}{n!} \left. \frac{d^n G_{Q_s}(z)}{dz^n} \right|_{z=0} \quad (3.25)$$

and from that the cell loss probability can be calculated. The delay can also be calculated using Little's law. The cell loss probability is shown against the buffer size for different switching fabric sizes under random uniform traffic with an offered load of $\rho = 0.8$ in figure 3.6-a for output buffering and in figure 3.6-b for shared buffering. It is obvious from these results that the buffer size required per an output port is much less for the shared buffer case. Moreover, from the position of the $N = \infty$ curve relative to the other ones in both cases, it can be observed that as the number of output ports increases, the buffer size per port for shared

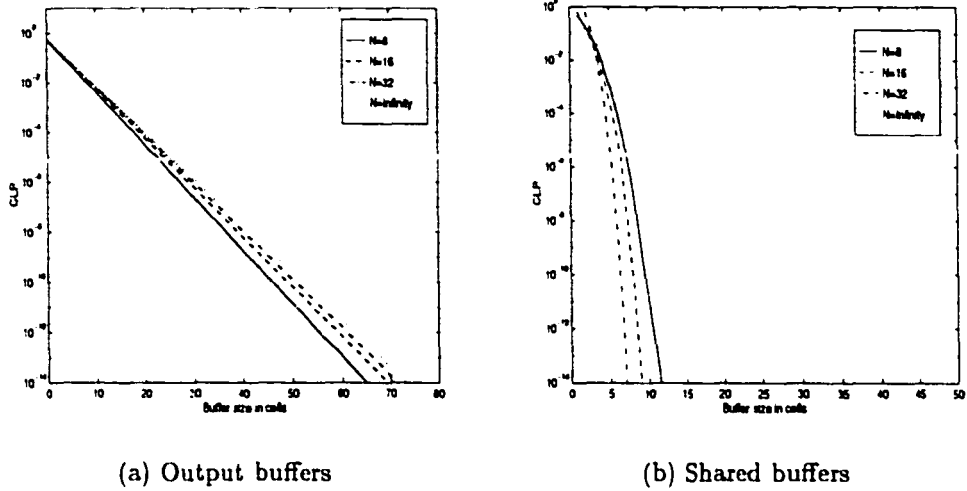


Figure 3.6: Analytical results for (a) output and (b) shared buffers with random uniform offered load $\rho = 0.8$.

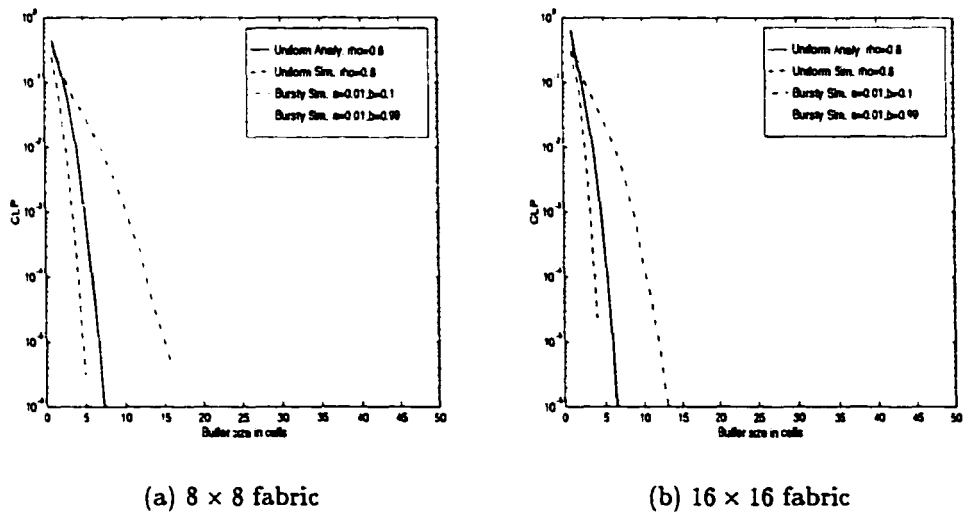


Figure 3.7: Analytical and simulation results for different switching fabrics using shared memory.

buffering gets smaller whereas it gets larger for output buffering.

A parametric model has been built for a shared memory switching fabric and it has been used to simulate its performance for random uniform traffic for different parameters. Figure 3.7 shows the simulation results of an 8×8 and a 16×16 switch with a uniform load of $\rho = 0.8$ overlaid with the results from the analytical model discussed above. Results for bursty traffic models, ON-OFF sources, are also shown for comparison. The results are shown up to a cell loss probability of 10^{-6} because it would take a long time to simulate beyond that point. If the simulation results are extrapolated, it can be shown that the appropriate size of the shared buffer with a reasonable safety margin should be about 19 cells per output port, or 152 cells for an 8×8 fabric under random uniform traffic with an offered load of 0.8 in order to achieve a very low cell loss probability of about 10^{-14} for satellite applications. This would only require a 64 kbits memory module. Building VHDL models and mapping them to a specific technology should not affect the performance and the same results should be obtained as long as the circuits comply with the timing constraints such that the cell slot time is as required.

3.4 Proposed Architecture

The shared memory architecture proposed addresses the disadvantages of shared memory architectures in terms of the need for a high access rate memory technology and it supports scalability and fault tolerance. This switching fabric architecture is shown in figure 3.8. In general, it uses an $M \times N$ fabric as a basic unit. Shared memory blocks and bit-slices are used to solve the access rate problem. A Clos

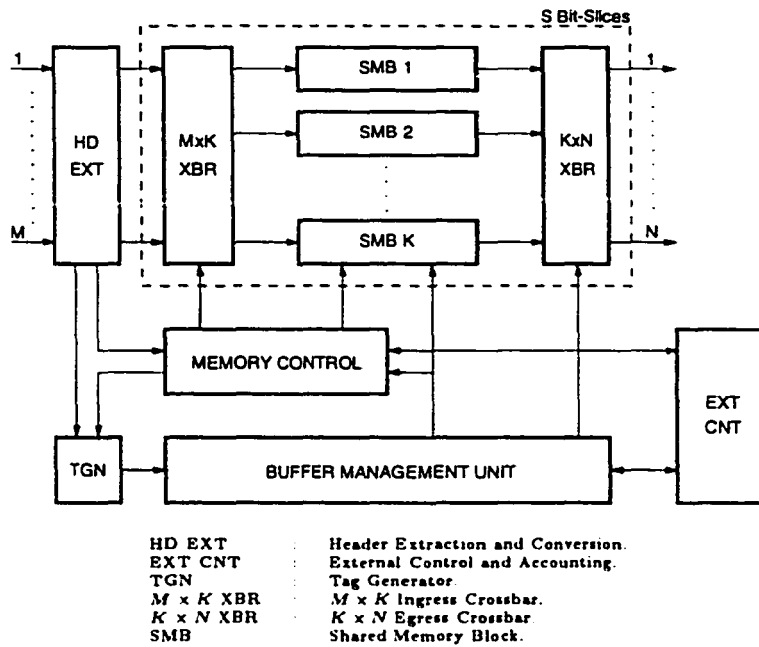


Figure 3.8: Proposed shared memory switching fabric architecture.

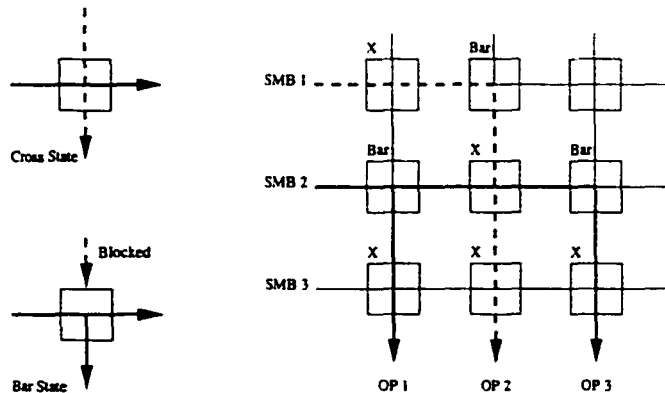


Figure 3.9: An example of multicasting from memory block SMB2 to output ports OP1 and OP3.

network is used to build scalable fabrics. Fault tolerance falls naturally from both shared memory and Clos networks. These issues are described hereafter, while a queue management unit solves the controller complexity problem and will be discussed in chapter 5.

3.4.1 Memory Bandwidth Issues

The shared memory of the switching fabric is arranged in K blocks accessible from all inputs and outputs through simple crossbar switches that allow for multicasting and broadcasting to all their ports as shown in figure 3.9. Instead of having an access rate of $N + M$ access operations, read or write, per cell time, each block can have an access rate of only N operations per cell time. This is because only outputs may have to access the same block simultaneously while the inputs can be controlled to write to different idle memory blocks. Memory access rate can now be independent of the number of input ports. However, the number of memory blocks, K , has to be selected properly according to the number of input ports and the RAM access rate as follows:

$$M + N \leq NK \quad (3.26)$$

The main core of the fabric is implemented in bit-slices to achieve the required throughput. For example, if the switch has 32 inputs and 8 outputs, at least 5 memory blocks will be needed and each of them should have an access rate of 8 operations per cell time.

Figure 3.10 shows the latest trends in dynamic memory access rates and densities

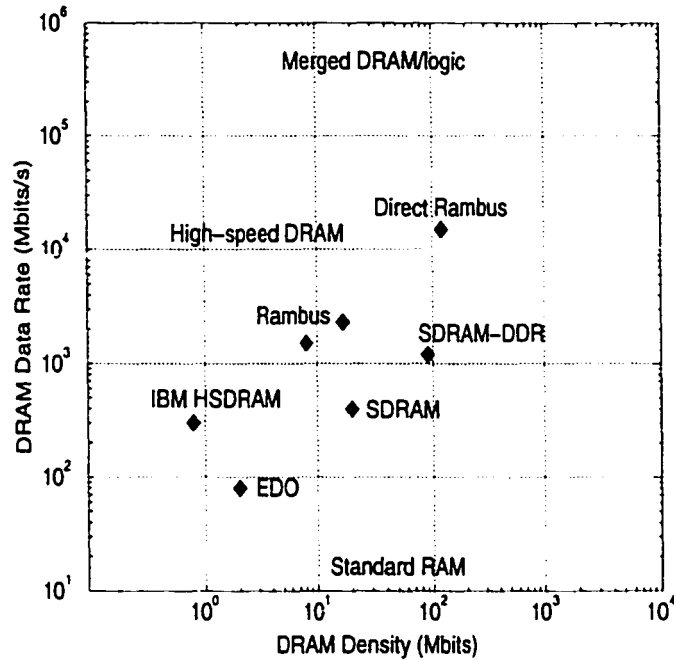


Figure 3.10: The relation between the DRAM density and its peak data rate for recent technologies.

[66]. A 16 Mbit SDRAM-DDR with 16 I/O pins operating at 100 MHz clock can provide 3.2 Gbits/s. An improved version of the Direct Rambus memory can provide 1.6 Gbytes/s which is 12.8 Gbits/s [37] operating at 250 MHz. Moreover, static memory can provide even higher data rates although at a higher cost. Now, if the memory blocks use the SDRAM-DDR technology which can provide up to 3.2 Gbits/s using a clock of 100 MHz, they can support all ports for a link rate of up to OC-48 or 2.4 Gbits/s if they are used in bit-slices. The aggregate rate required from the memory of an 8×8 switch is 19.2 Gbits/s so we need 6 bit-slices. Since the cell size¹ is 424 bits, the word length for each bit-slice would be 71 bits.

¹The tag overhead is stored in the queue management unit.

Similarly, to support OC-192 or 9.6 Gbits/s links, 24 bit-slices are needed, each with a word length of 18 bits. In this way, the speed bottleneck is moved from the memory to the controllers. Bit-slices are implemented as identical chips while the controllers are implemented separately. Note that these high speeds would impact the design of the transponders, air interface and multiple access technique used by the satellite [2].

3.4.2 Design Scalability

Several multi-stage networks were proposed to address the issue of scalability and implement large switching fabrics with large numbers of input and output ports. Most notable of these proposals are networks that use 2- or 3-stage interconnection networks [54, 79, 138] where the stages on the inputs side distribute the cells while the final stage on the output side buffers and dispatches the cells, such structures are usually referred to as *growable* architectures. One of the most popular choices is the Beneš network which was used in some architectures because it has a lower complexity at the cost of not being strictly non-blocking [50]. Another popular choice is the 3-stage non-blocking Clos network shown in figure 3.11.

We will refer to the network shown as $\mathcal{C}(i, j, n_1, n_2, n_3)$ where the first, second and third stages consist of n_1 identical $i \times n_2$ fabric modules, n_2 of identical $n_1 \times n_3$ fabric modules and n_3 identical $n_2 \times j$ fabric modules respectively. The network is said to be *non-blocking* if all the one-to-one connections are compatible. That is, cells can be switched over all the point-to-point configurations between the network inputs and its outputs in a certain cell time slot. A Clos network can be either *strictly non-blocking*, SNB, *rearrangeably non-blocking*, RNB, or *wide-*

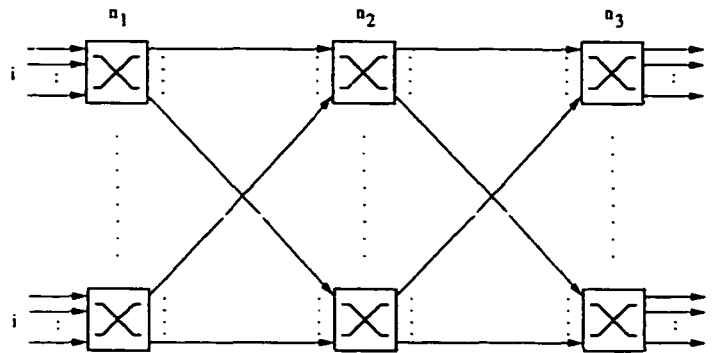


Figure 3.11: General configuration of $C(i, j, n_1, n_2, n_3)$ Clos network.

sense non-blocking, WNB [139]. This classification has a significant impact with regard to supporting multicast traffic and simplifying the CAC algorithm [22]. In an SNB network, a multicast connection request between available ports can always be satisfied without the need to rearrange the existing connections. Another advantage is that distributed algorithms are available for SNB networks which enhances the efficiency and fault tolerance of the overall fabric. On the other hand, RNB networks may need to rearrange the existing connections and this will not be practical in large networks with lots of virtual connections. WNB networks have certain routing algorithms to allow it to satisfy connection requests regardless of the establishment sequence of those requests, however, the algorithms are not distributive and would reduce the fault tolerance of the fabric.

The disadvantage of the SNB configuration, though, is that the number of the middle stage modules has to satisfy the following condition

$$n_2 \geq i + j - 1, \quad (3.27)$$

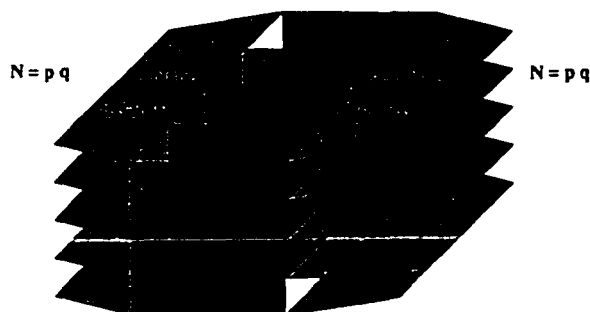


Figure 3.12: Recursive approach to the SNB Clos network.

while the RNB configuration would only have to satisfy the relaxed condition

$$n_2 \geq \max(i, j). \quad (3.28)$$

Figure 3.12 shows how we use an SNB Clos network to build a scalable switching fabric in a recursive manner. In this implementation, each plane represents a board where the different slices of a switching module are assembled. The recursive planes simplify the interconnects between them and enhance the modularity of the design. The figure shown implements a $C(p, p, q, 2p - 1, q)$ where both the numbers of input and output ports are equal to $N = pq$. If all the switching modules were designed using only crosspoints, we would need $2p(2p - 1)q + q^2(2p - 1)$ as compared to $N^2 = p^2q^2$ crosspoints in a full crossbar design. This is around 80% savings for a fabric with $p = q = 30$ or $N = 900$ ports.

However, a fabric with crossbar switching modules only does not have buffering capabilities that allow it to cope with traffic contention and scheduling requirements. This is where the shared memory switching modules described in the previ-

ous section are used. Another important point to consider is that ATM requires the delivery of cells in sequence. In order to maintain cell sequence using the proposed configuration, the second stage of the Clos network can not have any buffers and the switching modules in that stage are purely crossbar networks. Since the crossbar switching modules cost much less silicon area and power compared to the shared memory buffered switching modules, a larger number of modules can be used in the second stage and the use of SNB Clos networks has lower cost overhead and is justified. An advantage of the SNB Clos network is that routing across the stages can be done distributively using the algorithm proposed in [91]. Another important advantage of using the Clos network is that fault tolerance falls naturally as will be described in the next section.

3.4.3 Fault Tolerance and Reliability

Fault tolerance should be considered on several levels in this architecture. On the slice level, the crosspoints connecting the input and output ports to the shared memory blocks can have spare columns and rows of crosspoints along with some reconfiguration strategy. The number of the spare rows and columns depends on the reliability requirements expected since it is directly related to the number of faults that can occur without getting the whole slice out of service. Reconfiguration strategies available include *rippling replacement* where a faulty element is replaced with its neighbor in the same row, *fault stealing* where a faulty element is replaced by one of its neighbors on a row or a column and *repair-most algorithm* where complete rows or columns are replaced by spare ones [62, 130]. The details are beyond the scope of our work, however, our choice of the fault stealing strategy is

based on the fact that it does not require any configurations for input and output port connections and can accommodate more faults with a number of spare units that is midway between the numbers of spare units required by the other methods. On the other hand, redundancy is also introduced to the shared memory blocks, K , so as to replace any faulty blocks by one of the redundant ones. Detection of faulty memory blocks can be achieved using error detection codes [8]. Faults in memory are usually modeled as random or clustered faults. Clustered faults have a very low probability of occurrence in normal applications. A SEC-DED, single error correction-double error detection, code is usually enough to provide a very high reliability in normal applications. Temporary single bit-flip errors are combated using simple parity checks. Other ad hoc techniques are used to detect permanent addressing, coupling and periphery logic faults. In the space environment and depending on the radiation levels expected at the application orbit, more powerful error correcting codes. ECC, can be used and this will increase the number of bits needed to store an ATM cell. A factor R_{ECC} will be used to represent the redundancy in memory bits per slice and will be assumed to be a specified constant for every specific applications. When the memory controller of a certain slice detects inconvertible errors beyond a certain threshold for a certain interval in a shared memory block, this block is declared faulty and replaced by one of the redundant blocks [18, 88].

On the other hand, the Clos network can provide fault tolerance by introducing n_s redundant slices in each module and n_m redundant modules in each stage [16, 75]. For a stage of n modules and S slices per module, the overhead is $\frac{n_m}{n} + \frac{n_s}{S} + \frac{n_m n_s}{n S}$.

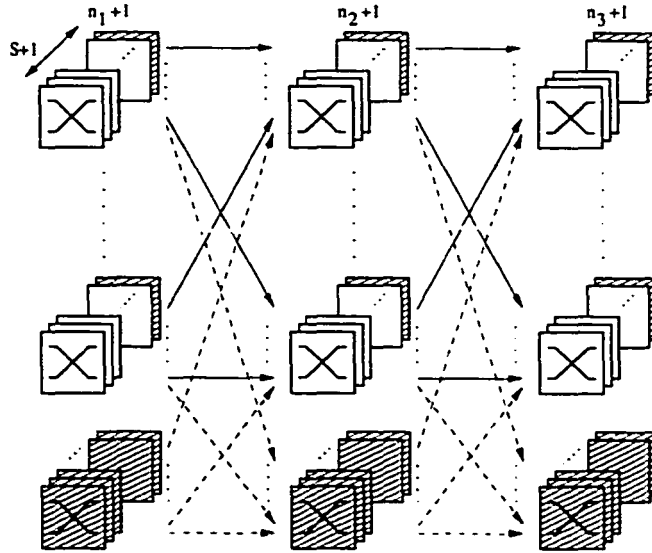


Figure 3.13: General configuration of $\mathcal{C}(i, j, n_1, n_2, n_3)$ Clos network with a single redundancy at slice and module levels.

Proper choice of n_m and n_s can reduce the overhead much below the 100% overhead of completely duplicated configuration. At the same time, redundant slices can be accommodated on the same module board or plane, reducing the overhead in boards. Figure 3.13 shows a $\mathcal{C}(i, j, n_1, n_2, n_3)$ Clos network with only one redundant module per stage and one redundant slice per module. Depending on the layout, technology and type of slice, the slice reliability can be estimated as $R_{\text{bufd}}(t)$ for the shared memory buffered architecture and $R_{\text{xbar}}(t)$ for the crossbars in the middle stage. Using that estimate, the reliability of an S -slice module without any spare slices is given by

$$R_{\text{mod}_0} = R_{\text{slc}}^S, \quad (3.29)$$

where R_{slc} is equal to either R_{bufd} or R_{xbar} depending on the type of slices used in that module. Consequently, the reliability of an S -slice module with n_s spare slices is given by

$$R_{\text{mod}} = \sum_{k=S}^{S+n_s} \binom{S+n_s}{k} (1 - R_{\text{slc}})^{S+n_s-k} R_{\text{slc}}^k, \quad (3.30)$$

and it can be normalized by dividing by R_{mod_0} . Similarly, the reliability of the i -th Clos stage, or the i -th stage in any other multistage interconnect network for that matter, with n_i modules and n_m spare ones can be found from

$$R_{\text{stage}} = \sum_{k=n_i}^{n_i+n_m} \binom{n_i+n_m}{k} (1 - R_{\text{mod}})^{n_i+n_m-k} R_{\text{mod}}^k \quad \forall i. \quad (3.31)$$

where R_{mod} is computed for the proper type of modules used in stage i and the normalization is achieved by dividing by $R_{\text{mod}}^{n_i}$. Finally, the normalized reliability of the whole network is the product of the reliabilities of all stages. The *mean time to failure*, MTTF, can also be found by integrating the reliability over all time.

Commutation of the spare units should be done automatically on the fly and without involving the input or output ports. Figure 3.14 shows the most common techniques used for that [107]. These techniques can be used to kick a spare slice in a module, a spare module in a stage or even a spare shared memory block within a slice. Using the slices within a module for illustration, the most straightforward technique is to route the S slice inputs to the spare unit using an $S : 1$ multiplexer and the S slice outputs from the spare unit to the output ports using a $1 : S$ demultiplexer. The problem with this technique is obviously the large fan-in and -out at the spare unit and the complexity of the interconnect and its routing.

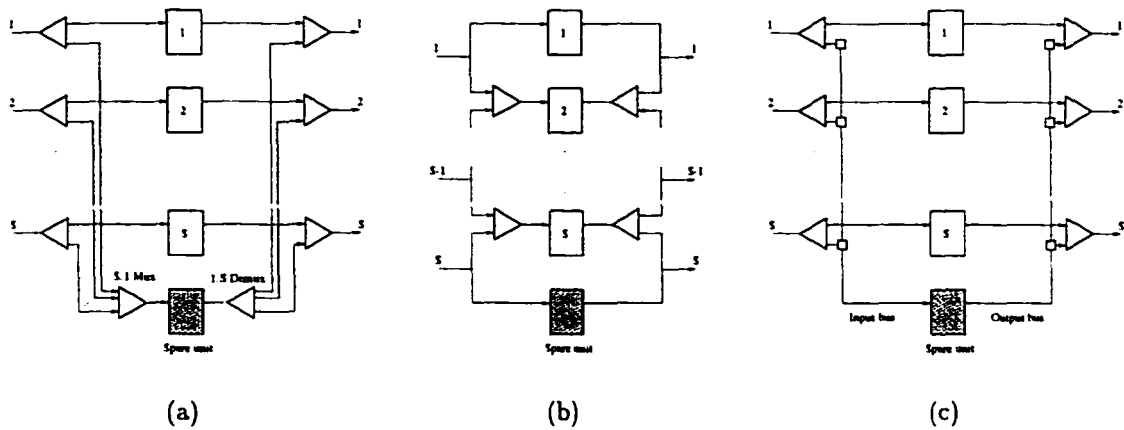


Figure 3.14: Common commutation techniques (a) mux/demux, (b) ladder and (c) bus.

Another technique is the ladder network, as shown in figure 3.14-b, which requires less complex interconnect but can not handle an arbitrary number of spare units since it only allows for nearest neighbor connections. Another drawback is that all units after the faulty one will have to shift their duties to the next one which can be time consuming and impractical in some cases. The technique chosen in this switching fabric is the bus technique which avoids the fan-in and fan-out problems, reduces complexity and gives complete flexibility in positioning and using the spare units. The problem with this technique is that the longer the bus is, the more delay it might introduce, however, this problem is alleviated by the fact that these buses are localized through out the different units.

The spares operate in a hot-standby mode. Spare slices are commuted to replace faulty ones in a module. If all spare slices are exhausted, the module is declared faulty and a spare module in that stage is commuted to replace it. Commutation of

the spares does not involve the input or output interfaces. If all redundant modules in the second stage are exhausted, the network will still be able to work although with a severely affected performance and it will not be non-blocking anymore. An important point to consider is that when a buffered module fails, some of the cells buffered inside it can still be recovered and in some cases most of them; such as in the case when the input interfaces of a buffered module fail. Dropping all the cells buffered in a module while only a few of them are corrupted can be disastrous in terms of QoS. To avoid this, the input side of a spare is commuted as soon as the fault is detected. The switching fabric attempts to flush out the cells in the faulty unit and when the memory controller declares it empty, the output side of the spare is commuted in its turn. In this way, corrupted cells are detected on an end-to-end basis and are retransmitted while good cells are salvaged. Non-buffering modules such as the pure crossbars used in the second stage have only one cell per connection at a time and their outputs can be commuted just one time slot after their inputs are commuted.

3.5 Summary

In this chapter, a shared memory architecture was proposed because of its many advantages. Most importantly is the buffer sharing effect that reduces the buffer size needed for a certain cell loss probability as compared to other architectures as well as the full throughput achievable by the logical output queueing in such architectures. An architecture that uses separate shared memory blocks to construct a shared memory pool in bit-slices is proposed. These blocks are accessed from the input

and output ports using crossbars to alleviate the bandwidth problems of the shared memory architectures. Bit-slicing also serves to reduce the memory bandwidth requirements and achieve a degree of fault tolerance. Other aspects were also discussed in terms of scalability, fault tolerance and reliability.

Chapter 4

Power Optimization Methodology

4.1 Introduction

Recent advances in VLSI technology such as scaling the clock speeds and higher memory bandwidth have made the design of packet switching fabrics with very high aggregate rates possible. However, such designs often produce very hot chips and low-power techniques are only employed at the circuit-level and attain little improvement. On the other hand, new applications are emerging with very scarce energy sources, special chip and system packaging requirements and complex cooling and heat dissipation processes. This situation is evident in the *global broadband networks* where the switches are used on-board low-earth-orbit satellites, LEOs. On-board switches would have to dissipate as little power as possible and would require special packaging to prevent the ill effects of radiation and ions on the electronic circuits. This type of packaging would also make cooling and heat dissipation a very difficult task. Consequently, all of those factors have to be considered during early design stages and the design for low-power dissipation should be at the

system-level where the largest possible improvement can be achieved.

Designing switching fabrics at the system-level has always been based on optimizing certain criteria related to performance metrics such as throughput, cell loss probability and mean cell delay [5, 128]. Only a handful of studies in the literature attempted to consider practical and physical requirements in switch design as well. Shaikh *et al.* compared the counts of the crosspoints and the pin-limited chips in two switching fabrics; namely the Shufflenet and Banyan networks [121]. In [34], Coppo *et al.* proposed a methodology for evaluation and optimization of Clos networks, used as ATM switching fabrics, based on component count and interconnection cost as physical cost requirements for a given connection blocking probability. Zegura presented a comparison of different switching fabrics based on the count of pin-limited chips needed by each fabric [141]. Shi *et al.* explored the optimization of the Knockout switch hardware cost given certain constraints on the quality of service [122]. Most recently, Schultz studied the performance limits of shared memory architectures using experimental results and technology trend curves [120]. Moreover, almost all of those studies used certain traffic profiles or a certain set of given quality of service parameters that would make the analysis of the physical requirements simpler and more tractable. In this chapter, a framework for system-level design of packet switching fabrics is proposed. This framework uses a traffic-based approach to determine the transition activity¹ for digital signals in a shared memory IP/ATM switching fabric. Results from this chapter have appeared

¹*Transition activity* refers to the probability of signal transitions between logic 0 and 1 which results in dynamic power dissipation in the system. It is usually referred to as *switching activity* but we will not use this convention to avoid confusing it with *switching* the packets through the fabric.

in [133, 135].

4.2 Low-Power System-Level Design Techniques

The issue of low-power design continues to gain more attention because of many reasons, some are obvious while others are subtle. Battery life is a well understood advantage for low-power consumption, however, reliability, performance, cost and time to market all benefit from lower power as well. Reliability and performance are temperature dependent. It is well known that long term reliability, in terms of *mean time between failures* MTBF, drops by approximately 50% for every 10 degrees rise in junction temperature. In addition, transistor currents, and hence performance, drop by approximately 3% for every 10 degrees rise, providing significant motivation to minimize power, which in turn minimizes junction temperature [11,41]. Cost is another key factor because of the more sophisticated and costly packaging solutions required to maintain a desired junction temperature. This affects the integrated circuit packaging as well as the system packaging as in the case of requiring larger fans to increase the air flow.

4.2.1 Power Measurements

Several power measurements exist. Maximum or peak power, P_{max} , figures are used in the design process to avoid circuit failure due to electro-migration or voltage drops on power and ground busses and to meet the worst case noise margins. Root-mean-squared power, P_{RMS} , is also used for sizing to meet the electro-migration rules which are usually sensitive to steady-state and periodic current flow. Hence,

RMS power or a variant of it is used to verify the design compliance with the current density rules. Instantaneous peak currents, sometimes referred to as $\frac{di}{dt}$ currents, are used to determine the allowable parasitic inductance presented to the device which is essential to minimize the ground bounce effects. Lastly, time-averaged power, or simply the average power, P_{avg} , represents a single value for power consumption obtained by integrating power on a cycle-by-cycle basis over a number of clock cycles. Average power is used to calculate battery life and junction temperature, and is therefore the most widely used type of power calculation. This is the type of measurement that we are mostly interested in and will be used throughout this work.

4.2.2 Power Sources

An integrated circuit total power consumption is comprised of two different categories. The primary distinguishing factor between the two is whether the power is frequency dependent or not. Frequency independent power is defined to be the product of the power supply voltage and a frequency independent current, which itself has two components: leakage current and static current. Leakage power, $P_{leakage}$, which occur in all MOS devices, is due to currents from sub-threshold transistor operation and reverse bias diode leakage. These currents are parasitic effects and are usually ignored except in those devices that have extremely long standby times or operate at extremely low voltages. Static power, P_{static} , is due to static currents which occur in normal operation also but are due to transistors being continuously operated in their saturation regions and arise in designs that employ analog techniques or resistive pull-ups. Static currents are on the order of micro- to

milli-amperes, however, they can be safely ignored if analog techniques are avoided. Correctly designed CMOS circuits do not have static current, however, it is useful in detecting faulty ones. Frequency dependent power also has a couple of components, short circuit power and dynamic capacitive load power. Short circuit power, P_{short} , represents the short circuit current that flows from the supply to the ground when the NMOS and PMOS transistor stacks switch their states. This current can be reduced by making sure that the signals have fast enough transitions. On the other hand, the dynamic capacitive load power, $P_{dynamic}$, represents the currents required to charge the fanout load capacitance. Dynamic power is normally given for all nodes in the system by the following equation

$$P_{dynamic} = \left(\sum_{v_i} \alpha_{0 \rightarrow 1, i} C_i V_i \right) V_{dd} f_{clk}, \quad (4.1)$$

where, for each node i , $\alpha_{0 \rightarrow 1, i}$ is its activity factor or the average number of power consuming transitions from 0 to 1 per clock period, C_i is its capacitance which is the total contribution of the gate, the diffusion and the interconnect capacitances and V_i is its voltage swing which is usually the same as the supply voltage V_{dd} while f_{clk} is the clock frequency.

The total average power dissipation in digital systems can then be expressed as the sum of all four components.

$$P_{avg} = P_{leakage} + P_{static} + P_{short} + P_{dynamic} \quad (4.2)$$

In an ASIC, or library-based, design methodology the frequency dependent currents

are combined into *cell*² and *load* currents. The cell currents include the short circuit current as well as the current required to charge the parasitic capacitance internal to the cell, while the load current represents the current required to charge the load capacitance external to the cell. It might also include the effect of the leakage currents.

4.2.3 Power Estimation and Reduction Techniques

To fully manage and optimize the power dissipation and at the same time reduce the design time, a level-by-level power analysis and estimation approach is needed. The higher the level, the easier it is to explore the design space and the more the impact of power optimization [90]. A classical feedback design flow is compared to a generic level-by-level low-power design flow in figure 4.1. In the classical approach, power information is not available until the gate or transistor level design is obtained. If the power specifications are not met, the design flow feeds back to higher levels where power optimization can be carried on. On the other hand, the level-by-level approach tries to provide power information earlier at the higher levels to replace the large feedback loop with smaller more efficient ones. Design efforts at higher levels usually show orders of magnitude savings in power while lower level gate and circuit optimizations typically offer improvement by a factor of two or less [85]. High-level power modeling and optimization techniques for digital systems encompass three different levels; namely, the behavioral, register transfer (RT), and software levels.

²The term *cell* in this context should not be confused with the ATM fixed-size packet. It refers to a physical entity corresponding to a gate or a collection of gates in a certain technology-specific library.

We will discuss the power modeling and estimation proposals for these levels then turn our attention to the corresponding power optimization approaches.

High-Level Power Modeling and Estimation

On the behavioral level, there are the information-theoretic models which try to capture a power estimate using entropy measures of signal activity either at the bit-level [92] or the word-level [99]. However, there is no universal information-theoretic measure for all designs and the approximation of the average activity by a certain measure, such as its entropy or informational energy, requires an appropriate choice for each design. Moreover, an assumption is made in such approaches as to how the entropy reduces from one logic level to the next. Exponential and quadratic reductions were assumed in [92] and [99] respectively, which is not true for general circuits and architectures. A similar point can be made with regards to the total capacitance which is estimated by quick mapping to universal gates or by information-theoretic models relating the gate complexity to the input and output entropy difference. Another approach is the complexity-based models which rely on the assumption that the complexity of a circuit can be estimated by the number of *equivalent gates* which is generated by analytical prediction functions [98] or using regression analysis performed on a large number of circuits [100]. Another technique obtains higher accuracy for controller circuits by introducing empirical parameters determined using least squares fit to real data [76]. Again, parameter tuning is needed for the prediction functions used with each specific design in this type of methods and still their accuracy is low except for the case where the parameters are obtained using real data fitting. Synthesis-based models are also used to select

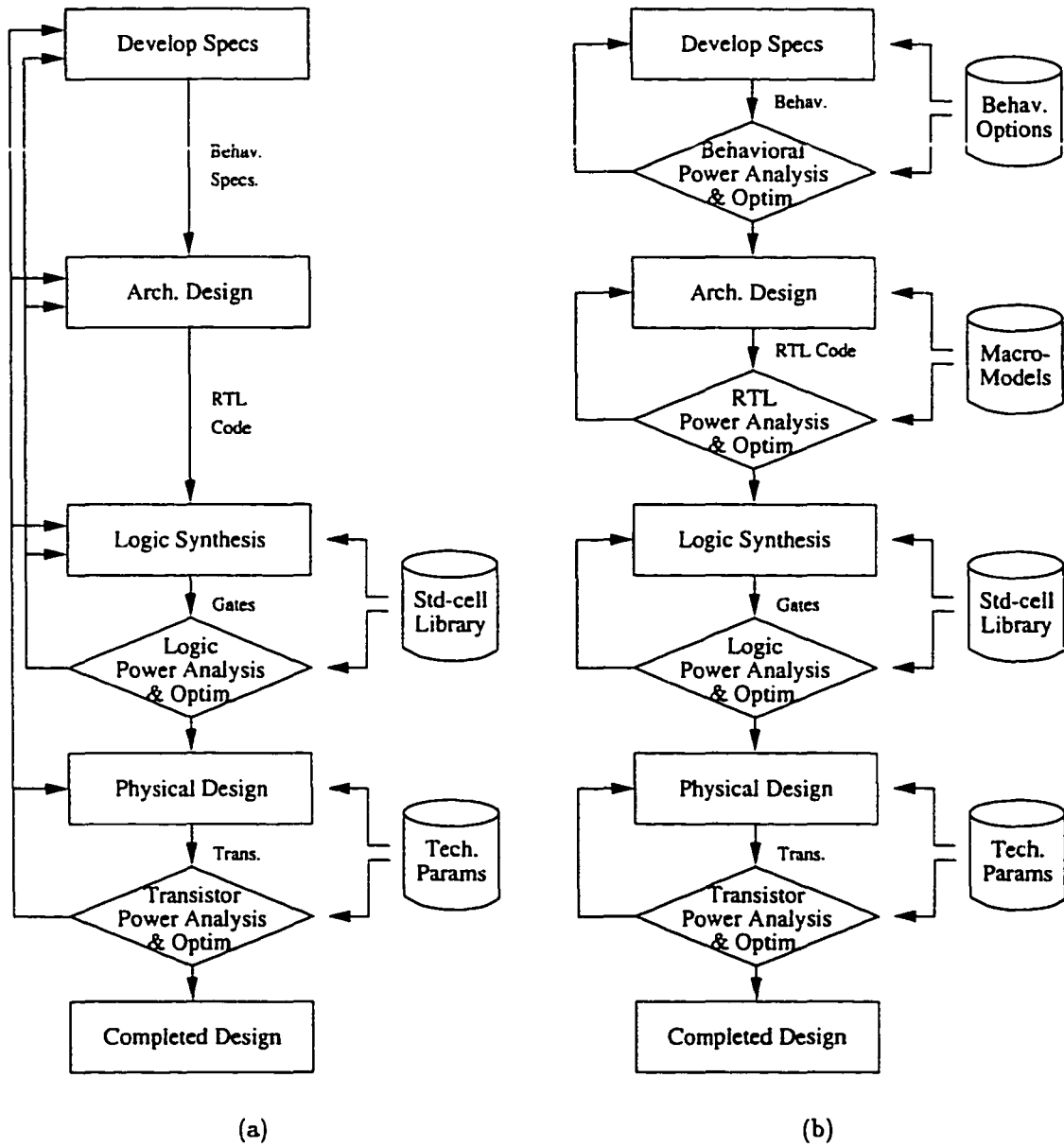


Figure 4.1: Low-power design flow: (a) classical feedback flow, (b) level-by-level flow.

from different behavioral options, such as memory organizations, pipelining issues, synchronization schemes, bus architectures and controller designs. Different options are quickly synthesized and relevant data statistics are captured by static profiling based on stochastic analysis [24] or by dynamic profiling based on simulations under typical input streams [71].

Power modeling and estimation on the RT level uses mostly regression-based switched-capacitance models for circuit models commonly known as *power macro-models*. These models have the following typical procedure to use:

1. All components in the high-level library are characterized under pseudorandom data and fitting a multivariable regression curve to the power dissipation using a least mean square error fit [13].
2. The macro-model variables are extracted from the design for which the power is to be estimated either by static analysis or simulation.
3. These values are plugged in as parameters into the macro-model equations from the high-level library prepared in the first step.

This category of models includes the *power factor approximation* technique [112] which does not account for data dependency, *dual bit type* model which is based on DSP signal statistics [77] which was also extended to include input/output activity in [51], parametric models which express the power as a function of a set of parameters related to the implementation style and internal architecture of the different components [86]. Macro-model techniques provide an estimate of the average power consumption over a large number of clock cycles and are called *cumulative*

power macro-models. They will be discussed further in section 4.3.1. *Cycle-accurate* power macro-models are used when the average power is not sufficient, such as in circuit reliability or noise analysis or in physical design optimization. These models cluster the closely related data patterns in separate clusters, e.g., using Hamming distance. Input patterns are mapped to those clusters and a lookup table that characterizes the power for each cluster is then used to derive the total power [94]. Another scheme uses statistical sampling of a training set of input patterns and regression analysis to select the most power-critical variables for the model [113]. *Census* macro-modeling uses a power co-simulator invoked by the RT level simulator every cycle. To improve efficiency, *sampler* macro-modeling invokes the power co-simulator at n random cycles defined before the simulation. Typically, the sampler improves efficiency by a large factor with an average error of 1% [55]. These techniques are all sensitive to the training data set and might be biased by it. However, the sample variance of the ratio of gate-level power to macro-model equation power tends to be much smaller than that of the gate-level power itself. Based on that, *Adaptive* macro-modeling invokes a gate-level simulator in a small number of cycles to reduce the model bias [90].

Software level power modeling is of little relevance here but it is mainly based on power macro-modeling [90]. These models either estimate the average switching capacitance upon activation of a CPU module or estimate the switching activity on the different buses. Other models include instruction level power models which take into account both the instruction and circuit state energy cost. Some models synthesize different programs that are shorter than the ones that will run on the

processor but would produce similar power traces. A power profile of those shorter programs is used to estimate the power for larger and more complex ones.

High-Level Power Optimization

Power optimization approaches have also been proposed on the same three levels. The behavioral techniques focus on optimizing the synthesis tools to take the power dissipation into account during the different stages of behavioral transformations, operation scheduling, resource allocation, multiple supply-voltage scheduling, bus encoding and control-logic synthesis and optimization. Techniques on the RT level overlap with those on the gate level and sometimes there is no definitive distinction between them. Nevertheless, these techniques are mainly based on precomputations, clock gating, guarded evaluation, detection of hold conditions, glitch minimization and retiming. Software optimization basically includes two main categories of techniques; namely, optimizing instruction scheduling and code generation to reduce the number of transitions occurring and minimizing memory access costs. Since most of these techniques are already integrated in the available design tools, we will not discuss them any further but rather use them through these tools. An excellent introduction to these techniques can be found in [14]. Moreover, most digital systems are event driven. Power management policies can be used to shut down those parts of the system that are not active. These policies can be implemented at almost all of the abstraction levels down to the circuit level. Some of these policies are already available through the design tools and some need provisions to be included in the cell libraries or the logic and control design.

4.2.4 Low-Power Approach

Our approach is two-fold. To build parametric modules for each of the different components used in the switching architecture such that these modules make use of all of the available power optimization techniques in the design tools at both high and low levels of abstraction. The second part is to develop a system level optimization problem that is based on a suitable power estimation model to choose appropriate values for the parameters of each of the modules used such that the overall design minimizes the power dissipation and satisfies all other constraints imposed.

The synthesis tools available already use some of the high-level power optimization techniques mentioned before. Moreover, some low-level techniques [11] were also used to optimize the power for each of the parametric modules and mostly using the automated design tools. These can be summarized in the following techniques.

- *Power minimization by frequency reduction.* It is obvious that decreasing the clock frequency reduces the dynamic power linearly. However, the time needed to finish a certain task increases, which has negative consequences, while the overall energy dissipated to complete the task does not change. Battery-powered systems such as those on-board satellites, however, benefit from the lower power and the associated lower current discharge. It was shown recently [93] that the total amount of energy provided by batteries, C , is dependent on their rate of discharge according to the empirical *Peukert's*

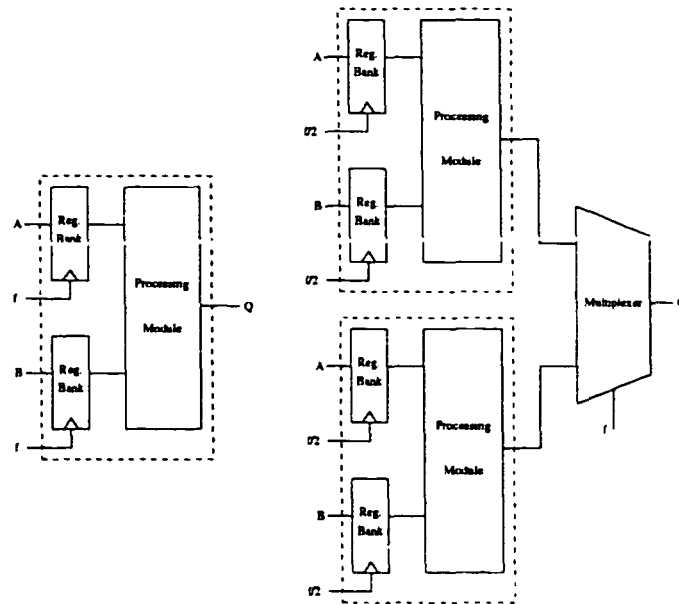


Figure 4.2: Reducing the power by frequency reduction and parallelization.

formula

$$C = \frac{\chi}{I^\alpha} \quad (4.3)$$

where χ is a characteristic of the particular type of battery technology used, I is the average discharge current and α is a technology dependent fitting factor. So, reducing the frequency reduces the power and the discharge current and consequently increases the total amount of energy provided by the battery. In order to make use of frequency reduction and avoid reducing the system throughput, architecture level techniques such as parallelization, see figure 4.2, are used within the building blocks or parametric modules whenever possible provided that the area increase is minimal. With the frequency

reduced, each of the parallel blocks have to satisfy a smaller worst case delay and the voltage of the power supply can be reduced. The supply voltage reduction together with the frequency reduction can overcome the effect of the capacitance doubling due to parallelization and the total effect would be to reduce the power considerably.

- *Power minimization by supply voltage reduction.* The quadratic relation between dynamic power and supply voltage makes this the most attractive option. However, this option is limited by the available technology limitations. For a certain standard cell library, there is a certain minimum nominal supply voltage and a maximum clock frequency for which the cells are optimized. Modifying the library is out of the scope of this work, however, we should strive to use the best available library in terms of the lowest supply voltage. In CMOS, an inverter delay, T_d , has the following trend

$$T_d \propto \frac{V_{dd}}{(V_{dd} - V_t)^2} \quad (4.4)$$

where V_t is the threshold voltage. This relation assumes the transistor current to fit the quadratic model and that during transients the transistor controlling the load capacitance is in saturation [23]. If we had control over the library at circuit level, parallelization and pipelining transformations could be used to allow for supply voltage reduction and overcome the delay penalty and provide the same throughput. Similarly, low-swing internal signals can be used to reduce power dissipation which is the practice, for example, in static memory design. In deep sub-micron technologies, however, the devices are

velocity saturated³ and this quadratic model does not apply. Consequently, device limitations can restrict the architectural options too and it will become increasingly difficult to move to lower voltages because of reduced noise margins and deterioration of device characteristics as V_{dd} approaches V_t .

- *Power optimization by capacitance reduction.* Dynamic power has a linear dependence on capacitance. The capacitive load of a CMOS gate consists mainly of three components,

$$C_{out} = C_{fo} + C_w + C_p \quad (4.5)$$

where C_{fo} is the gate capacitance of the inputs of the fan-out gates, C_w is the interconnect or wires and C_p is the parasitic junction, gate-to-source and gate-to-drain capacitance of the transistors in the gate. In state-of-the-art technologies, approximately 50% of C_{out} is due to C_{fo} , 40% of C_w and 10% of C_p . C_w is becoming more dominant in deep sub-micron technologies and it is already dominant for data busses and global control wires. C_p on the other hand causes the least concern since it is well characterized, small compared to the other components and constant in a first order model. Reducing C_{out} not only decreases power but also reduces area and delay. Thus, automated layout tools, and whenever possible manual intervention, were used to minimize the routing area and the active silicon area and consequently the capacitance.

From the power point of view, only the most active capacitance need to be

³A transistor is velocity saturated if no improvement in transit time of the electrons through the conductive channel can be obtained by increasing the drain-source voltage.

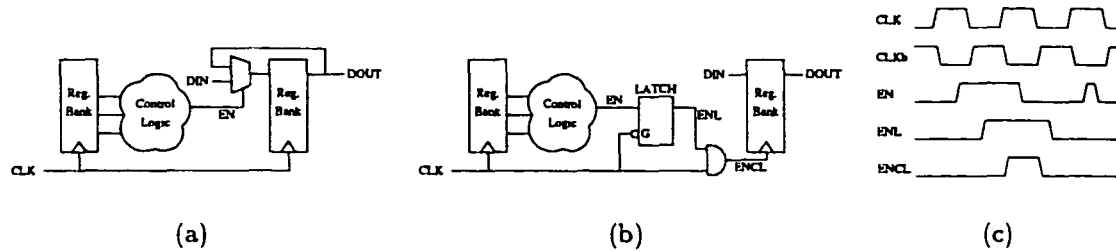


Figure 4.3: Register banks with (a) synchronous load enable or (b) clock gating and (c) its timing diagram.

reduced. On the other hand, a slight increase of capacitance may reduce the power considerably as in the case of parallelization.

- *Power optimization by transition activity reduction.* Reducing the transition activity can be done in several ways. One of which is to reduce the *useless* activity using power management techniques based on shutting off the power supply of unused parts of the systems or stopping the clock from reaching those parts by clock gating techniques. In standard cell based designs, it may be difficult to shut off the power supply without manually modifying the physical layout. Clock gating on the other hand can be used as illustrated in figure 4.3. Another way is to optimize the circuit and logic design using available tools to remove spurious transitions and glitches due to unequal propagation delays which also represent useless activity. One powerful method is to use asynchronous or self-timed digital design techniques [63,69,103] such as micro-pipelines whenever possible.
- *Power optimization using circuit-level techniques* These techniques are mostly

employed during the design of the standard cell library and as far as this work is concerned it boils down to using the most optimized standard cell library. Nevertheless, these techniques include transistor sizing especially for clock drivers [56] and transistor reordering within the cell [33]. Other tool supported techniques are at the physical layout level and include low-power placement, clock tree optimization and wire sizing [33].

4.3 System-Level Power Optimization of the Shared Memory Architecture

In this section, we will consider the shared memory architecture for an $M \times N$ switching fabric, where M and N are the numbers of input and output ports respectively, which was described in detail in section 3.4 and illustrated in figure 3.8. The link rates were all assumed to be V cells per second. In that architecture, a physical memory pool is used by all logical queues, hence the term “shared”. Logical queues are implemented using address linked lists and managed by a separate memory controller. This shared memory pool is divided into K blocks accessible from all inputs and outputs. This allows each block to have an access rate of only N operations per cell time independent of the number of input ports. The number of memory blocks, K , has to be selected properly [131, 138] according to equation (3.26) restated here for completeness

$$M + N \leq NK. \quad (4.6)$$

The fabric is also implemented in S parallel bit-slices to support even higher throughputs and provide fault tolerance.

IP/ATM switching has been recently proposed to switch IP traffic, connection-less variable size packets, over ATM networks, using connection-oriented fixed size cells, to provide faster service with prespecified classes of quality of service, QoS, rather than the best effort service provided by TCP/IP networks. In most implementations of this scheme, certain IP packet flows, i.e., persisting connections, are chosen for switching over the ATM network rather than being routed through regular store-and-forward routers of the TCP/IP network. Each packet of those flows is divided into as many fixed size cells as needed and a certain bit in the ATM cell header is used to indicate the end-of-packet cell. Those cells can then be switched through the ATM network as usual. For IP/ATM switching, the design of the switching fabric is exactly the same as that for ATM switches in terms of its logical and physical structure. They only differ in the traffic sources and how their statistics affect the design.

4.3.1 Power Analysis and Modeling

As discussed above, the average power, P_{avg} , dissipated in digital systems can be divided into four components; namely, dynamic, short-circuit, leakage and static. However, the short-circuit and static power components are normally negligible in CMOS logic gates. Also, the leakage current, and power, is negligible in advanced fabrication technologies. Hence, the dominant power component is the dynamic one. From equation (4.2), the average power for all nodes, i , in the system can be

expressed as follows

$$P_{avg} \approx P_{dynamic} = \left(\sum_{V_i} \alpha_{0 \rightarrow 1,i} C_i V_i \right) V_{dd} f_{clk}. \quad (4.7)$$

However, it is not practical to evaluate this power at all nodes in large scale systems and a faster high-level technique is needed.

Architectural-Level Power Analysis

For such large digital designs, states and dynamic events of the architectural components are too many to use gate- or transistor-level abstractions for the power analysis. Architectural-level abstraction is the level of choice for power analysis during system-level design. This abstraction level is based on a library of architectural building blocks or components implemented as parametric modules. These parametric modules are characterized using a power macro-model based on extracted coefficients and power estimates from accurate lower abstraction levels such as the gate- or transistor-levels.

As mentioned before, the most appropriate models for high-level power modeling in this case are the cumulative macro-models since we have systems built of several components that differ in their architectures and need to be modeled separately. More importantly, cumulative macro-models capture the system behavior over a long time interval and based on the input statistics which is more appropriate in the case of highly stochastic network traffic that can not be completely described using simple measures such as entropy. These cumulative power macro-models encompass several approaches. One approach is based on *gate equivalent counts* [86,98] where the complexity of the architectural components is specified in terms of the average

number of reference logic gates. The component power is then the number of the reference gates times the power of the reference gate derived from a transistor- or circuit-level analysis. The power due to the interconnects is estimated using variants of Rent's rule for local and intermediate interconnects and H-tree for the clock network [7]. The power for a certain memory architecture is used to model the power of the on-chip memory and improve the estimate of the overall power. The main problem with this approach is that it does not take the transition activity of the real input data into account but rather uses a fixed value for it. Another approach, the *power factor approximation* [112], is based on pre-analyzed models for a library of functional modules. The power for each module is determined using a gate-level abstraction and assuming independent *Uniform White Noise*, UWN, data streams at the inputs. The power of the overall architecture is estimated by identifying the blocks needed to build the system given enough system-level specifications and summing the power of all blocks. Although, this approach accounts for the transition activity, the assumption of independent UWN data is not always valid. In [78], Landman *et al.* proposed the *dual bit type*, DBT, data model. It was shown that for a stream of two's complement samples, typical in DSP applications, bits can be classified into two types. The first type includes the least significant bits whose transition activity can be approximated by a UWN model. Bits of this type will be called the UWN bits. On the other hand, the most significant bits, which represent the sign, may have a relatively low or high transition activity because of a positive or negative temporal correlation between the data samples respectively. These bits represent the second type and will be called the correlated bits. The power

estimation in this approach is based on a black-box effective switching capacitance concept. For the bits with UWN activity, the effective capacitance is computed for each and every component using low-level simulation⁴ to find the average power, P_o , at reference supply voltage, V_o , and reference frequency, f_o , and using a UWN excitation.

$$C_{uwn} = \frac{P_o}{V_o^2 f_o}. \quad (4.8)$$

This effective capacitance is then used to estimate the power for a typical bit of the the UWN bits at different frequencies and supply voltages. For the correlated bits, the effective capacitance can take one of four values according to the bit transition between successive data samples: C_t , $t \in \{00, 01, 11, 10\}$. These values are computed in a similar way as C_{uwn} with the appropriate excitation. If p_t , $t \in \{00, 01, 11, 10\}$ were the probabilities of the corresponding bit transitions in the input data stream, then the effective switching capacitance for all bits can be computed using

$$C_T = L_u C_{uwn} + L_c \left(\sum_{t \in \begin{pmatrix} 00, 01, \\ 11, 10 \end{pmatrix}} p_t C_t \right), \quad (4.9)$$

where L_u and L_c are the numbers of UWN and correlated bits respectively. Hence, the total power is given by

$$P = C_T V^2 f. \quad (4.10)$$

⁴Power annotated gate-level simulation, IRSIM or POWERMILL switch-level simulation or SPICE circuit-level simulation whenever possible.

Although this technique is limited by the assumption of a specific numeric data representation scheme and by its dependence on the input transitions only, it can be improved to be robust and accurate enough for system-level power analysis. Several successive studies have addressed some of these limitations. They demonstrated the importance of accounting for the transitions at the internal nodes and the use of least mean square regression models to obtain better power estimates [20,51,55,94].

Most recently, Bogliolo and Benini [19] proposed another model, the BB model, that is characterization-free for the dynamic power but then uses gate-level simulations to characterize the delay-sensitive second-order power contributions using a regression model. In effect, this is merely modeling the error of the characterization-free part and not necessarily the glitches or short-circuit power. A simpler approach would be to use regression for the whole model. It also assumes that the dynamic power of a certain node is proportional to the transition activity derived from the module Boolean expressions with the node capacitance as the proportionality constant. However, the assumption of the capacitance being independent of the input streams is not always true. For example, the capacitance seen at the inputs of a complex gate differs with the input pattern depending on which transistors are on. The model, however, illustrates other important facts. It shows that not all of the internal node capacitances need to be considered. These node capacitances can be sampled to speed up the characterization with only minor loss of accuracy [19,55]. It also shows that a uniform random sampling gives the best results. This technique will be used to modify the DBT model as will be described below. Moreover, all of these schemes still need to be generalized to accommodate different types of

data representation such as those for network traffic streams commonly found at the input ports of the switching fabrics.

Bit-Level Analysis

Before studying the transition activity for each bit in an ATM cell, we need to point out the fact that all of the components used in the switching fabric discussed have no activity coupling between adjacent bits and it is safe to assume independent distributions for the transition activity of the cell bits. Thus, it is necessary to study the activity factor at the bit-level for all of those components when they are excited with real network traffic streams. This was done for several types of ATM traffic streams for all 424 bits of the ATM cell. Single streams at the UNI as well as multiplexed streams at the NNI were used. These included CBR, VBR and ABR traffic processed using AAL1, AAL2 and AAL5 layer respectively. Moreover, multiplexed ATM traffic from all three classes, single and multiplexed IP packet streams over ATM cells were also analyzed. Typical results are shown in figure 4.4.

From these results, it is obvious that a generalized form of the DBT model [78] is needed. Depending on the traffic mix, the bits of the ATM cell are decomposed into activity regions. Bit regions with activity factor approaching 0.25 are assumed to have UWN activity. Each of the remaining correlated regions is similar to the sign bits in DSP applications except that each region has different transition probabilities. The probabilities of bit transitions t estimated for each region r are $p_{t,r}$, $t \in \{00, 01, 11, 10\}$. L_u and $L_{c,r}$ are the total number of bits in the UWN regions and the number of bits in each correlated region r . Consequently, our proposed model has generalized bit types and the effective switching capacitance is

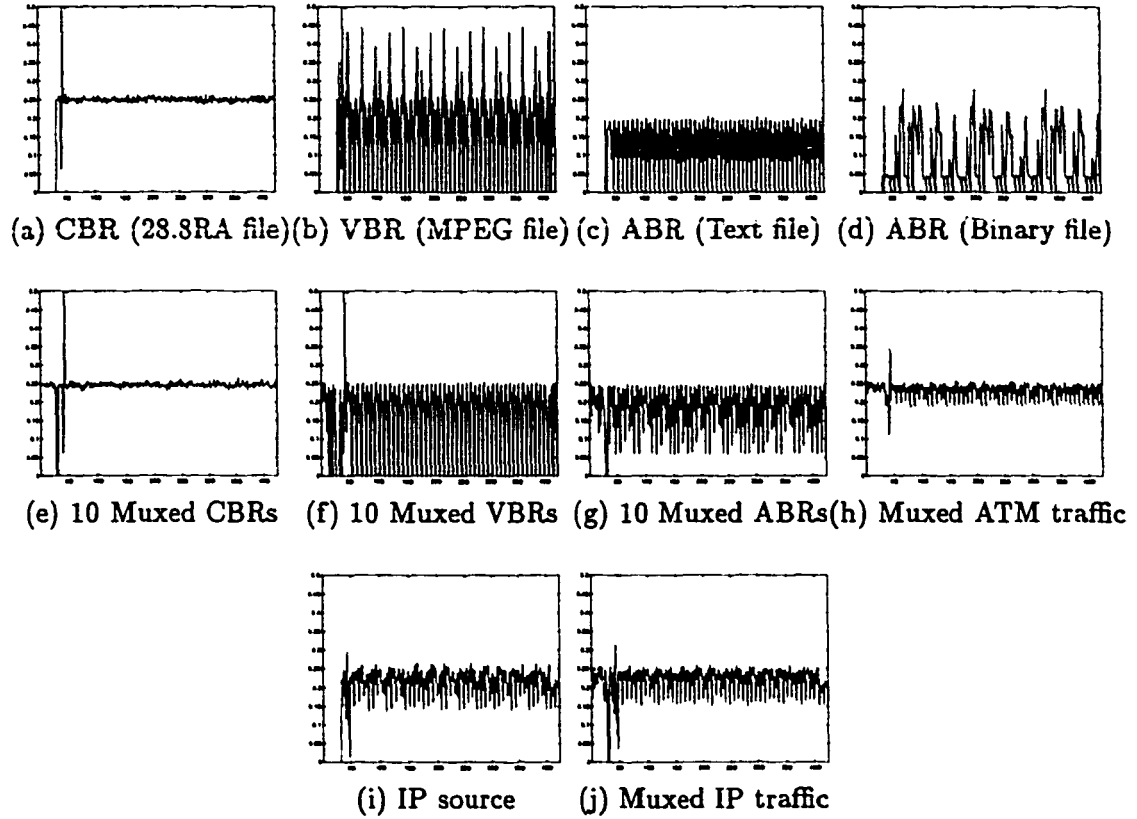


Figure 4.4: Transition activity factor vs. bit number in an ATM cell for various traffic streams.

given by

$$C_T = \frac{L_u}{L/S} [C_{\text{eff}} \cdot L_f] + \sum_{\forall r} \frac{L_{c,r}}{L/S} \left[\sum_{t \in \begin{pmatrix} 00,01, \\ 11,10 \end{pmatrix}} p_{t,r} [C_t \cdot L_f] \right], \quad (4.11)$$

where C_{eff} and C_t represent the effective capacitance vector of a bit-slice of the design for UWN bits and for the correlated bits with transition t respectively and L_f represents the bit-slice complexity vector. To account for all bits, the weights

$L_u/(L/S)$ and $L_{c,r}/(L/S)$ are included, where L is the cell width in all bit-slices and is normally equal to 424 and S is the number of bit-slices.

Cell-Level Analysis

The capacitance switched is influenced by the arrival rate of cells, ρ , at the inputs of each module. This can be derived from the D-BMAP model which will be discussed in detail in section 4.3.2. It is also influenced by the module type and dimensions including the buffer length, B , in cells which will also be derived in section 4.3.2. The switched capacitance for the static memory blocks has the form

$$C_{\text{mem}} = (c_3 \frac{MBL}{K} \frac{L}{S} + c_2 \frac{MB}{K} + c_1 \frac{L}{S} + c_0) K \rho, \quad (4.12)$$

where those capacitive terms are computed for the memory bit array, the decoding circuitry, the sense amplifiers and other periphery circuits, and the control circuitry of the memory respectively. The coefficients include the effect of the layout and the fabrication technology parameters. Any power management technique used to deactivate unused blocks will have to be considered in the equation through a probabilistic weight. Similarly, the switched capacitance for the crossbars is given by

$$C_{\text{xbar}} = c_5(p_x M + p_b N) K \frac{L}{S} \rho + c_4 \rho, \quad (4.13)$$

where p_x and p_b are the probabilities that a crossbar element is in a cross or a bar state respectively given a cell arrival and are constants specific to the distribution of source-destination address pairs in the traffic at the ingress crossbar and to the scheduling algorithm at the egress crossbar. The second term in equation (4.13) is due to the control circuitry of the crossbars. The capacitive coefficients, C_{eff} or C_t , and the complexity parameters, L_f , can then be written in vector form as follows

$$\mathbf{C} = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 & c_4 & c_5 \end{bmatrix}, \quad (4.14)$$

$$\mathbf{L}_f = \rho \begin{bmatrix} K & K \frac{L}{S} & MB & MB \frac{L}{S} & 1 & (p_x M + p_b N) K \frac{L}{S} \end{bmatrix}^T. \quad (4.15)$$

Extracting the Capacitive Coefficients

To extract the capacitive coefficients several steps have to be followed. An input stream or pattern has to be generated for a certain value of input temporal correlation for every parametric module we have. This pattern is then used to simulate the modules using a switch-level or transistor-level simulator with capacitance measurement capabilities to estimate the average capacitance. The use of switch-level or transistor-level simulation takes into consideration the glitching power because of the inherent inclusion of a delay model in these simulations. In [19,55], it was shown that, for a given module, the energy for a particular transition from one input to the next is proportional to the energy estimated using only a uniform random sample

of internal nodes. During this simulation step, the internal nodes are also sampled to extract their corresponding capacitances which depends on the input transition propagated through the circuit. The data generated in this step represents equation (4.14). However, this data must be fit to the regression model that represents the complexity of the module. To do this, the simulation is repeated for each parametric module over a wide range of parameters and the resulting capacitances are fit using the least square fitting technique. The whole process is then repeated using different correlations for the input patterns to extract separate capacitance coefficients for those different correlations, i.e., C_t , $t \in \{uwn, 00, 01, 11, 10\}$. In short, this model uses general bit types for the different transition activity regions and models them using sampled capacitances in a regression power macro-model. Hence, we will refer to it as the *General Sampled-Bit Regression* model, GSBR.

4.3.2 Performance Modeling

The Traffic Source Model

Before we can discuss the dynamic power optimization, a model for the traffic source has to be established in order to be able to derive the transition activity for different nodes throughout the architecture. To do that, the *Discrete-time Batch Markovian Arrival Process*, D-BMAP [17], which is a very flexible model that can be used to model both native ATM traffic classes and IP traffic or an aggregate of them, is chosen. Formally, a D-BMAP model can be defined as a two-dimensional discrete-time Markov process $\{A(k), P(k) : k \geq 0\}$ on the state

space $\{(i, j) : i \geq 0, 1 \leq j \leq m\}$, where m is the number of states in the irreducible modulating Markov chain, with the transition matrix

$$\mathbf{H} = \begin{pmatrix} \mathbf{D}_0 & \mathbf{D}_1 & \mathbf{D}_2 & \mathbf{D}_3 & \dots \\ 0 & \mathbf{D}_0 & \mathbf{D}_1 & \mathbf{D}_2 & \dots \\ 0 & 0 & \mathbf{D}_0 & \mathbf{D}_1 & \dots \\ \vdots & \vdots & \ddots & \ddots & \dots \\ \vdots & \vdots & \ddots & \ddots & \dots \end{pmatrix}. \quad (4.16)$$

The two sets of random variables $\{A(k) : k \geq 0\}$ and $\{P(k) : k \geq 0\}$ represent the number of cell arrivals on a time slot-per-slot basis, and the state, or the phase, of the underlying Markov chain respectively. Transitions between subsequent states are governed by the transition matrices $\mathbf{D}_n, n \geq 0$, whose elements $(d_n)_{u,v}, 1 \leq u, v \leq m$, describe the probability that n cells are generated and that the modulating Markov chain is in state v during a slot, given that it was in state u in the previous slot. The matrix $\mathbf{D} = \sum_{n=0}^{\infty} \mathbf{D}_n$ is the transition matrix of the underlying Markov chain, with a stationary probability vector π satisfying

$$\pi = \pi \mathbf{D}, \quad \pi \mathbf{e} = 1, \quad (4.17)$$

where \mathbf{e} is a column vector of appropriate size with all elements equal to 1.

The D-BMAP model has the advantage of representing a point process which can be used as an input process independently of the queueing system unlike other models such as the fluid approximation method. It also leads to accurate conservative approximations for a large range of system parameters, whereas the fluid

approximation is valid only for systems with very large buffer sizes and the M/D/1 approximation is valid only for systems with small buffer sizes. Also, the superposition of two D-BMAP processes with m_1 and m_2 states and maximum batch sizes of n_1 and n_2 is also a D-BMAP process with $m_1 \times m_2$ states and $n_1 + n_2$ maximum batch size. This property is very helpful in modeling several different sources separately and aggregating them into one model. Finally, the output process of the system, which is the input process of the next network element, e.g., a multiplexer or a switch, can be modeled again as a Markovian arrival process of the same class as the input process. On the other hand, the number of the states of the Markov chain may grow rapidly, and the use of the special structure of the transition matrix is essential to keep the method computationally tractable for realistic problems. In this work, the statistics of both ATM traffic and IP flow traces are used to derive the transition matrices of the D-BMAP model using a variation of the matching procedure proposed in [123]. The statistics of each traffic source are matched to a Markov modulated Bernoulli process, MMBP, which is the special case of a two state D-BMAP model. This choice mitigates the problem of rapid state growth to a large extent. Finally, the traffic source models are superimposed into one D-BMAP model.

The Switch Queueing Model

To model the shared memory architecture, a single output multiplexer is considered along with its associated logical queue. The input process is modeled as a D-BMAP process as discussed above. The overall system can then be modeled as a D-BMAP/D/1/ B queueing system where the service time is equal to one time slot

and the buffering capacity is finite and is equal to B ATM cells. This model can be solved at steady-state to obtain the occupancy probability of the queue. Assuming that the traffic destined to different output ports is independent, the occupancy probability of the shared buffer for a switch with N output ports can be obtained from the N -fold convolution of the occupancy probability for the queue of a single port multiplexer. This probability is the key to our formulations throughout the rest of this chapter.

Several methods to derive the steady-state distribution of the buffer occupancy have been proposed in the literature. An efficient numerical method to compute the steady-state probability and the cell loss probability is described in [17] and is used here. This method is based on the Matrix-Geometric solution approach used to solve M/G/1 queueing systems. However, the numerical solution is iterative by nature and no closed form solution can be obtained. For example, a specific value for the cell loss probability, CLP, may be used to determine a proper buffer size per port. Formulating the system-level parameter selection as an optimization problem is now complicated by this iterative solution and will require very long solution times. In order to avoid that, a family of curves can be fit to the CLP solution and used in the optimization problem. The CLP is computed using the Matrix-Geometric method iteratively for different switch sizes as a family of curves, such as that shown in figure 4.5. Extrapolated simulation results for the case of $N = 8$ are also shown to agree quite well with the analytical results in this figure. Real IP flows were used to develop the D-BMAP model used to generate these results. It is obvious that these curves are almost linear on a logarithmic scale within the

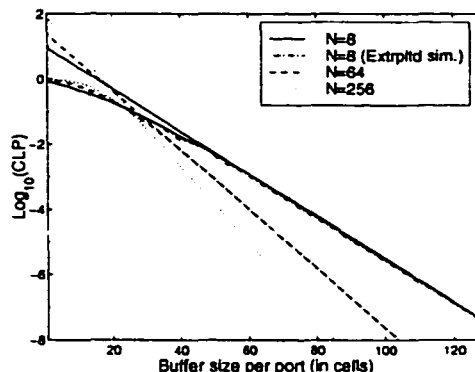


Figure 4.5: Sample results from the solution of the D-BMAP modeled traffic and their linear approximations on a logarithmic scale.

range of interest of the buffer size, B , and a linear equation can be fitted to them as follows

$$\log_{10}(CLP) \geq f_1(x, \mathbf{H})B + f_2(x, \mathbf{H}), \quad (4.18)$$

$$x = \log_2 N, \quad x \in \mathbb{Z}, \quad x \geq 1. \quad (4.19)$$

The choice of system-level parameters does not affect the traffic model and for every design problem the aggregate traffic pattern expected is assumed to be known. Hence, the transition matrix \mathbf{H} is dependent only on the types of traffic streams and their intensities and is constant for every specific design problem. Consequently, the coefficient functions are reduced to $f_1(x)$ and $f_2(x)$. These two functions can then be approximated by using the least squares regression method to fit them to

appropriate equations of the form

$$f_i(x) \approx t_{i,0} + t_{i,1}x + t_{i,2}x^2 + \dots + t_{i,l}x^l, \quad i \in 1, 2 \quad (4.20)$$

where $t_{i,j}$ are dependent on the traffic model only and l is an appropriate order to provide enough accuracy and facilitate the optimization process later. All of the traffic models developed for this work used $l \leq 3$ and provided accurate approximations.

Matrix-Geometric Solution Algorithm

In recent years, several methods have been proposed to solve models of M/G/1 type and quasi-birth-death, QBD, models. Li *et al.* [83,84] proposed a generalized folding algorithm [1]. However, the D-BMAP/D/1 is simple enough and will be solved off-line which makes the matrix-geometric solution algorithm a better candidate for this application. The matrix-geometric solution was first proposed [101,102] as a generalized analytic technique for the embedded Markov chains of the M/G/1 and GI/M/1 queueing models. An excellent introduction to its application to the batch Markovian arrival process can be found in [89]. The algorithm used here is an extension of that to the discrete case following [61,114]. It follows the following steps.

1. Start with a stochastic or zero matrix $\mathbf{R}(0)$ and iterate to get \mathbf{R} , also known as the *passage time matrix*:

$$\mathbf{R}(k) = \sum_{n=0}^{N_{trnc}} \mathbf{D}_n [\mathbf{R}(k-1)]^n, \quad (4.21)$$

where N_{trnc} is the batch size at which we can truncate computations without loss of accuracy and k is the iteration number. The iteration process ends when

$$\max_{i,j} \left| R_{i,j}(k) - \left\{ \sum_{n=0}^{N_{trnc}} \mathbf{D}_n [\mathbf{R}(k)]^n \right\}_{i,j} \right| \leq \epsilon \quad \forall i, j, \quad (4.22)$$

where the subscripts i, j denote the element in row i and column j of the corresponding matrix and ϵ is a very small number around 10^{-10} . Since \mathbf{D}_n is assumed to be 0 for $n > N_{trnc}$, the matrix polynomial $\sum_{n=0}^{N_{trnc}} \mathbf{D}_n \mathbf{R}^n$ can be efficiently evaluated using Horner's rule [101] and at each iteration $\mathbf{R}(k)$ needs to be normalized to be a stochastic matrix.

2. Let $\mathbf{x}(i)$ be a vector of m elements, one for each state of the enumerated states of the D-BMAP model. If this vector represents the probability of having i cells in the buffer or the occupancy probability of the buffer, compute the vector $\mathbf{x}(0)$ as follows:

$$\bar{\mathbf{D}}_i = \sum_{n=0}^{N_{trnc}-i} \mathbf{D}_{i+n} \mathbf{R}^n, \quad (4.23)$$

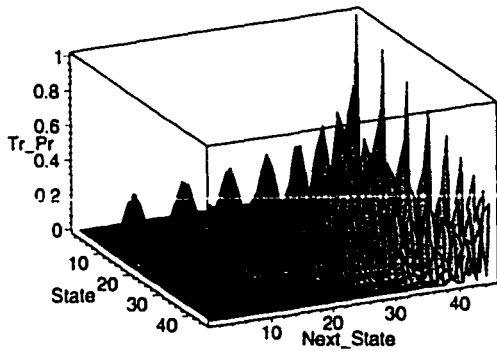
$$\mathbf{D}^* = \sum_{n=1}^{N_{trnc}} n \mathbf{D}_n, \quad (4.24)$$

$$\mathbf{Z} = \mathbf{D}_0 + \bar{\mathbf{D}}_1 (\mathbf{I} - \bar{\mathbf{D}}_1)^{-1} \mathbf{D}_0, \quad (4.25)$$

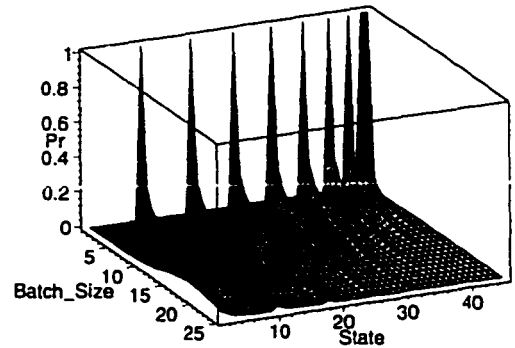
$$\mathbf{T} = \bar{\mathbf{D}}_1 \mathbf{R}, \quad (4.26)$$

$$\rho = \pi \mathbf{D}^* \mathbf{e} \quad \text{and} \quad (4.27)$$

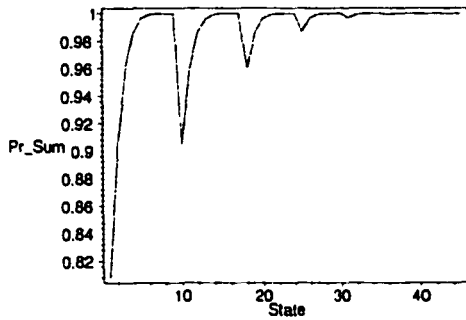
$$\mathbf{x}(0) = \frac{1}{d} \mathbf{z}, \quad (4.28)$$



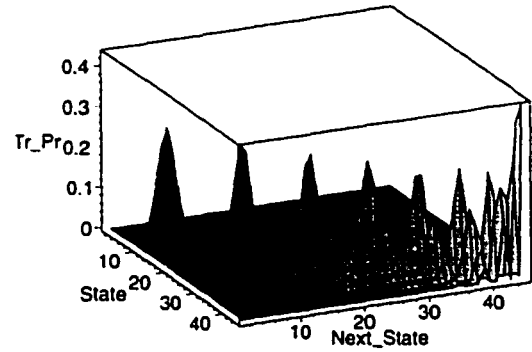
(a) D matrix



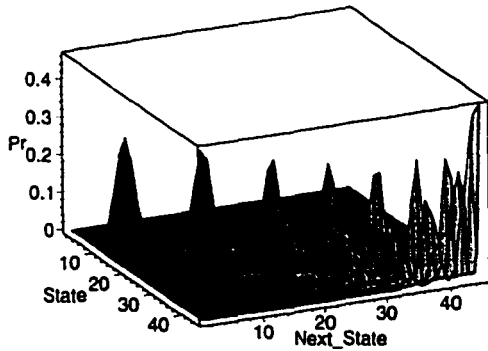
(b) Batch-size matrix



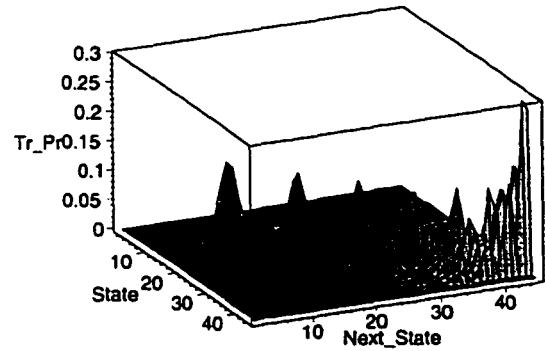
(c) Truncated sum at $n = 24$



(d) D_0 matrix

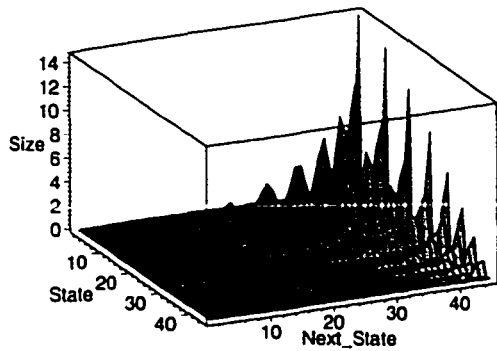


(e) Passage time matrix R

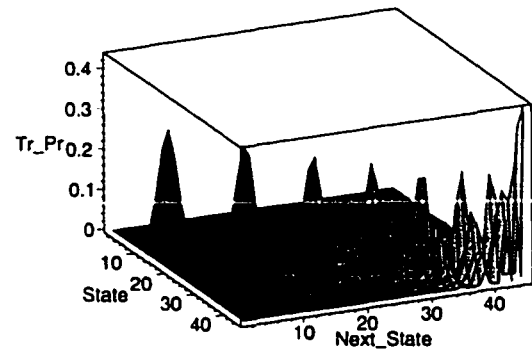


(f) \bar{D}_1 matrix

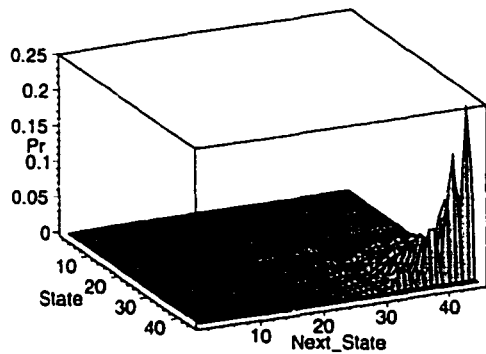
Figure 4.6: Sample results of the matrix-geometric algorithm solving a D-BMAP model of 8 identical bursty sources with $\rho \approx 1.48$.



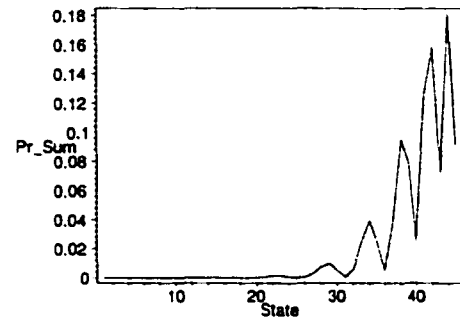
(g) D^* matrix



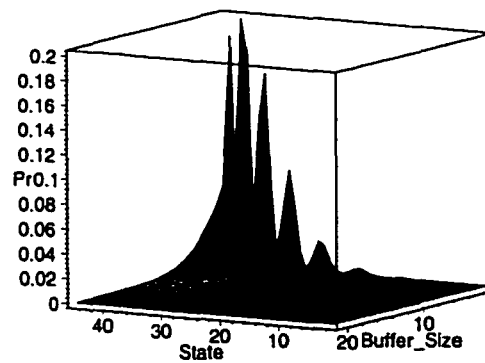
(h) Z matrix



(i) T matrix



(j) π vector



(k) Buffer occupancy x

Figure 4.6: Sample results of the matrix-geometric algorithm solving a D-BMAP model of 8 identical bursty sources with $\rho \approx 1.48$. (cont.)

where \mathbf{z} is the invariant probability vector of \mathbf{Z} satisfying $\mathbf{zZ} = \mathbf{z}$, $\mathbf{z}\mathbf{e} = 1$, and

$$d = 1 + \frac{1}{1-\rho} \mathbf{z}[\mathbf{I} + (\mathbf{D} - \mathbf{D}_0 - \mathbf{T})(\mathbf{I} - \mathbf{D} + \mathbf{e}\pi)^{-1}]\mathbf{D}^*\mathbf{e}. \quad (4.29)$$

3. Compute $\mathbf{x}(i)$ for $i \geq 1$ by Ramaswami's recurrence [89]:

$$\mathbf{x}(i) = \left[\mathbf{x}(0)\bar{\mathbf{D}}_i + \sum_{n=1}^{\min(i-1, i-N_{trnc}+1)} \mathbf{x}(n)\bar{\mathbf{D}}_{i-n+1} \right] (\mathbf{I} - \bar{\mathbf{D}}_1)^{-1}. \quad (4.30)$$

The traffic intensity or cell arrival rate given in equation (4.27) is again equivalent to that given by

$$\rho = \pi \left(\sum_{n=0}^{\infty} n\mathbf{D}_n \right) \mathbf{e}, \quad (4.31)$$

under the assumption that $\mathbf{D}_n = \mathbf{0}$ for $n > N_{trnc}$. Some experimental results for this algorithm are shown in figure 4.6. This analysis is done for a D-BMAP model of 8 identical bursty sources with $\rho \approx 1.48$. It has $\frac{(8+1)(8+2)}{2} = 45$ enumerated states representing which and how many sources are on. The buffer occupancy derived and shown in figure 4.6-k is then used to derive different performance constraints for different designs.

4.3.3 Other System-Level Requirements

Area Requirements

Heuristic formulas similar to those used in [120] were derived using parameterized deep sub-micron layouts for the area of the crossbar elements, A_{xbar} , and the static memory cells, A_{cells} . On the other hand, parametric VHDL models and layouts

for primitive logic gates were used to model other decoding and periphery circuits, A_{prphry} .

$$A_{\text{xbar}} = a_0\lambda^2 + a_1\lambda^2(M + N)K\frac{L}{S}, \quad (4.32)$$

$$A_{\text{cells}} = a_2\lambda^2 MB\frac{L}{S}, \quad (4.33)$$

$$A_{\text{prphry}} = \lambda^2 K \left(a_3 + a_4 \frac{MB}{K} + a_5 \frac{L}{S} \right), \quad (4.34)$$

where λ is the characteristic length for the technology to be used. Interconnect area was accounted for in the above equations by using layout measurements of interconnect length or estimates from Rent's rule. The above procedure was repeated for the three technologies available and the scaling factors, $a_i, i = 0 \dots 5$, were found using the least squares method. Consequently, the total area of a single bit-slice is given by

$$A = \lambda^2 \left((a_0 + a_3K) + (a_4 + a_2\frac{L}{S})MB + (a_5K + a_1(M + N)K)\frac{L}{S} \right). \quad (4.35)$$

Again, this area should be constrained for every chip depending on the yield, packaging type and the active area of the die. An empirical relation similar to that in [120] will be used here

$$A < 1.35 \times 10^{-4} \lambda^{-0.85}. \quad (4.36)$$

I/O Pin Constraints

In a switching fabric design, many constraints can be imposed due to the I/O pins used [120]. The pin count can limit the number of chips used and can even impose a

certain type of packaging. On the other hand, the pin count itself can be limited by the chip area and the I/O pad size and pitch. Pin count can also limit the aggregate throughput due to the pin speed limits. A constraint that is more related to this work is the I/O power due to the pads associated with those pins. The switched capacitance for the pads would be given by

$$C_{\text{PADS}} = (c_6 M \frac{L}{S} \rho + c_7 N \frac{L}{S} \rho_o + c_8), \quad (4.37)$$

where ρ_o is the cells departure rate from the fabric. It was shown in [17] that the departure process of a D-BMAP/D/1/B queue is a D-MAP process which is a special case of the D-BMAP process. The departure process can be derived from the queuing model and the departure rate ρ_o can then be calculated. The first two terms in equation (4.37) represent the switching capacitance for the input and output pads respectively. The last term represents the remaining control pads. These capacitive coefficients and complexity parameters can be appended to the vectors in equations (4.14) and (4.15) to add the power dissipation of the pins to the fabric power expression.

Other Performance Constraints

The aggregate traffic rate, or throughput, expected from an $M \times N$ fabric with a link rate of V cells/sec is limited by the number of bit-slices used and by the access rate, or bandwidth, of the memory used, BW_{mem} , according to the following relation

$$S \geq \frac{NVL}{BW_{\text{mem}}}. \quad (4.38)$$

Depending on the application that the switching fabric is being designed for, equations governing the cell transfer delay, CTD, or the cell delay variation, CDV, can be formulated similar to those for CLP and included as constraints for the optimization problem if necessary.

4.3.4 Error Sensitivity of the Model

To check the robustness and accuracy of this model, the different components along with 2×2 , 4×4 and 8×8 test fabrics were built in a $0.8\mu\text{m}$ BiCMOS, $0.5\mu\text{m}$ CMOS and $0.35\mu\text{m}$ CMOS technologies. The test fabrics were merely for test purposes and each had a buffer space that guarantees a cell loss probability less than or equal to 10^{-5} , which is a bit relaxed, for all the D-BMAP models used. To develop those D-BMAP models, multiplexed IP streams captured from the core of the network, or the NNI, were divided into two sets and used to derive the traffic models $IP_{c,1}$ and $IP_{c,2}$. Similarly, multiplexed ATM streams captured from the core were used to develop models $ATM_{c,1}$ and $ATM_{c,2}$ while those captured at the edge, or the UNI, were used to develop $ATM_{e,1}$.

Using the $IP_{c,1}$ model and the three different fabrication technologies, the power dissipation estimated using the proposed power model was compared to the power evaluated using switch-level simulations [56]. Figure 4.7-a shows that the relative errors did not exceed 8.4% for any of the tested components. These results show that the power model is robust and accurate enough to be used in system-level design. Moreover, the errors seem to be decreasing with the size of the test fabrics which makes the model more suitable as larger parameter values are explored. Also, those same test components implemented using the $0.35\mu\text{m}$ CMOS technology

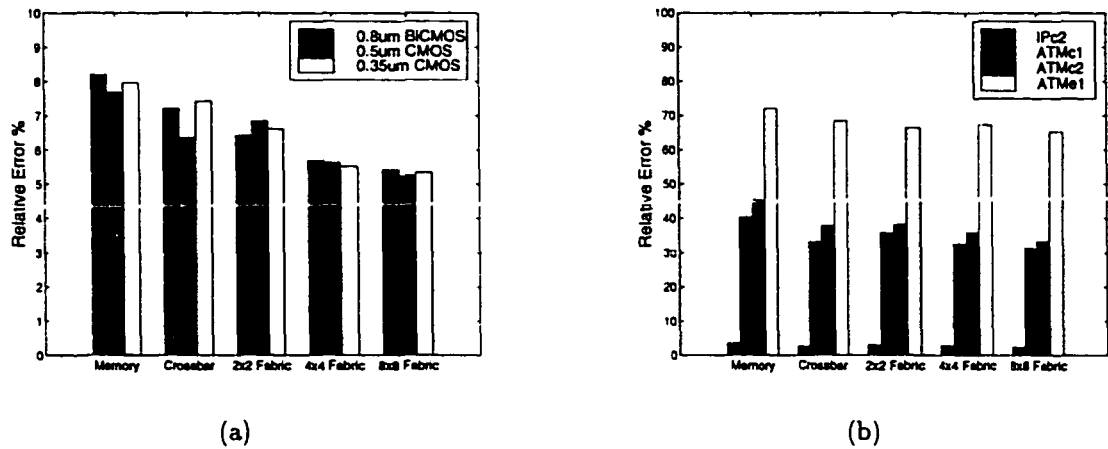


Figure 4.7: Relative errors for power dissipation estimates (a) fixing the $IP_{c,1}$ traffic model and (b) fixing the $0.35\mu\text{m}$ CMOS technology.

were used to compare the power dissipation estimated for the $IP_{c,1}$ model to that estimated for the other traffic models. The relative errors shown in figure 4.7-b indicate that the power model of the fabric is indeed dependent on its input traffic. This is evident from the large errors associated with traffic types other than the IP traffic used to estimate the power dissipation. Also, the figure shows that the only acceptable relative error is the one for the $IP_{c,2}$ model. These results assert the accuracy of the D-BMAP model in modeling specific aggregates of traffic and its suitability for use with the power model.

In order to compare the performance of the proposed model relative to the other macro-models, the power estimates are compared for the same $IP_{c,1}$ traffic model and the $0.5\mu\text{m}$ CMOS technology using the proposed model (GSBR), the dual bit type model (DBT) and the Bogliolo and Benini model (BB) as shown in figure 4.8.

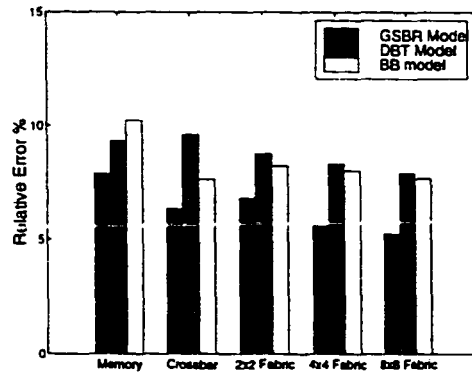


Figure 4.8: Relative errors for power dissipation estimates for different power macro-models (using the $IP_{c,1}$ traffic model and the $0.5\mu\text{m}$ CMOS technology).

The errors compared are again the relative errors of the estimated power using these models to the power evaluated using switch-level simulations [56]. The results show that the proposed model improves the results considerably. An interesting point is that the DBT model fares better for memory than the BB model because the DBT model takes into consideration the detailed blocks of the memory while the BB model assumes the dependency on the input transitions is only manifested in the logic expressions which is not completely true for memories and it also ignores the dependency of the capacitance on the input transitions. The proposed model uses the best of the two. It acknowledges the dependency of the capacitance on the input transitions, as in the DBT model, and improves its modeling using the least mean squares regression. It recognizes the different types of bit transition models rather than only two types. It also uses the sampling technique as in the BB model to speed up the characterization and maintain high accuracy.

Figure 4.9 shows the relative errors for the area estimated using the heuristic

model defined by equation (4.35). The errors are within 19.6% which is acceptable considering the fact that the layout and the physical design rules may differ widely between different technologies. If more design rules and technology parameters were included, the formulation would be too complicated and the accuracy gained would not be significant for a system-level constraint.

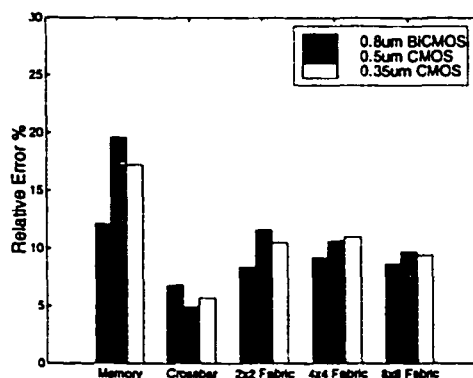


Figure 4.9: Relative errors for the area estimates compared to the actual layout area for different fabrication technologies.

4.3.5 Optimization for Low-Power

The proposed framework formulates and solves the design problem as an optimization problem. The main objective of this framework would be to minimize the power dissipation within the fabric. Other applications may regard the chip area or some other requirement as the objective function. Multi-objective optimization can also be used to minimize the power dissipation along with other requirements. In global satellite networks, a limited number of ports is needed and the shared memory switching fabric can be used as a stand-alone fabric. M is commonly taken to

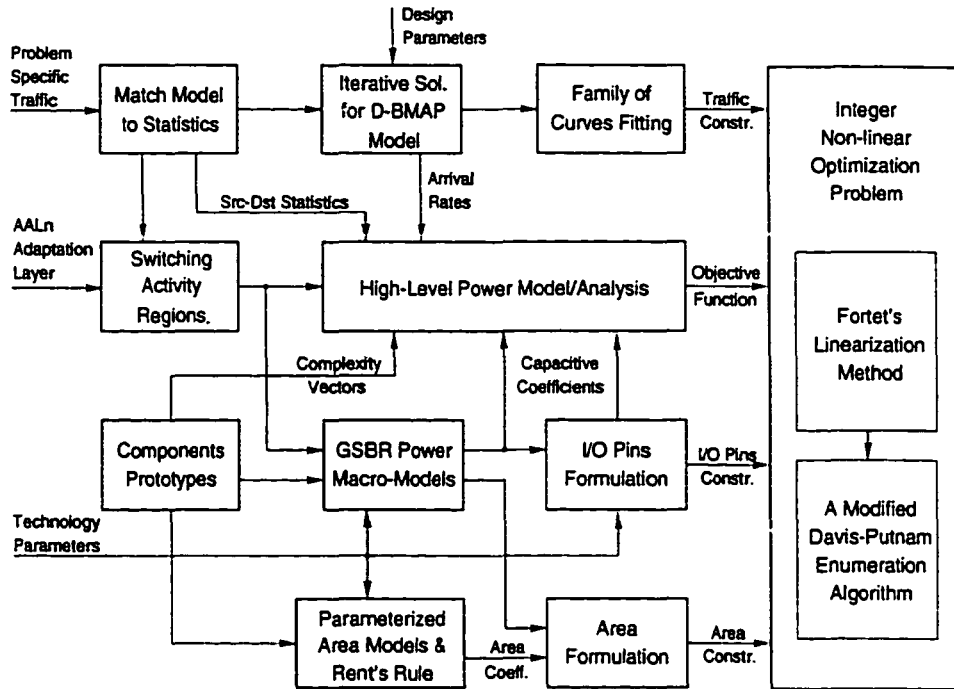


Figure 4.10: Switching fabric design in satellite applications at the system-level formulated as an integer non-linear optimization problem.

be equal to N to simplify the design problem. Consequently, several equations are simplified. The constraint on K in equation (4.6) will be

$$K \geq 2. \tag{4.39}$$

Similarly, the complexity vector, the total area and the switched capacitance for the pads in equations (4.15), (4.35) and (4.37), respectively, can be rewritten as

$$L_f = \rho \left[K \quad K \frac{L}{S} \quad NB \quad NB \frac{L}{S} \quad 1 \quad 2(p_x + p_b)NK \frac{L}{S} \right]^T, \tag{4.40}$$

$$A = \lambda^2 \left((a_0 + a_3 K) + (a_4 + a_2 \frac{L}{S}) NB + (a_5 K + 2a_1 NK) \frac{L}{S} \right) \quad (4.41)$$

$$\text{and } C_{\text{PADS}} = (c_6 N \frac{L}{S} \rho + c_7 N \frac{L}{S} \rho_o + c_8). \quad (4.42)$$

The design problem is formulated as an integer non-linear optimization problem since the design parameters, i.e., N, B, K and S , are all integers. Figure 4.10 summarizes how the models are derived and how they interact within the proposed framework. The objective is to minimize equation (4.10), with C_T defined by equations (4.11), (4.14), (4.40) and (4.42). The different constraints define the solution domain for the optimization problem. Equations (4.18) and (4.19) define the performance constraints while equations (4.38) and (4.39) set lower bounds for the numbers of slices and shared memory blocks respectively. The area constraints are defined by equations (4.36) and (4.41).

This formulation could also be used to derive a figure of merit and perform several useful comparisons. The following equation represents a figure of merit, F_M , that favors architectures or designs that minimize the power dissipation and silicon area and at the same time achieve a low cell loss probability. It is defined as

$$F_M = \frac{-c \cdot [\log_{10}(CLP)]^\alpha}{P \cdot A^\beta}, \quad (4.43)$$

where c is a normalization factor, $\log_{10}(CLP)$, P and A are similar to the definitions given in equations (4.18), (4.10) and (4.41) respectively. The factors $0 < \alpha \leq 1$ and $0 < \beta \leq 1$ are used to give appropriate weights based on the significance of the constraints. A system designer can tune these factors depending on the in-

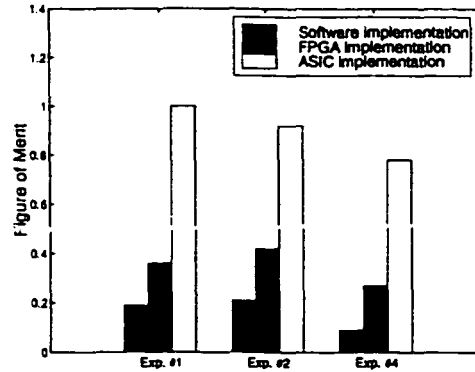


Figure 4.11: Normalized figure of merit for the proposed architecture implemented using different technologies and parameter.

tended application. This figure of merit can be used to compare one architecture implemented in different types of technologies, one architecture implemented using different parameter values, different architectures implemented using the same technology and parameters or could even be used to compare different modeling and optimization techniques. However, comparing different architectures or different techniques would require a detailed study, modeling and implementation of those architectures or those techniques. In figure 4.11, the architecture proposed is compared using different parameter sets from experiments 1, 2 and 4 of table 4.1 and different implementation technologies; namely, software, FPGA and ASIC. The power estimate for the software implementation used a power macro-model that is based on instruction power profiles [90]. To be able to compare all the different cases, a common normalization factor was used for all of them rather than having a normalization factor for each experiment or for each implementation technology. A value of $\alpha = 1/2$ is used to give the cell loss probability less weight than the power

dissipated and a value of $\beta = 1/3$ gives the silicon area the least weight of all three parameters. Software implementations are the worst ones since they are assuming general purpose processors that are not optimized for this application. Moreover, software implementations do not scale well and will not be able to handle the traffic load as the link rates increase.

Integer Non-Linear Optimization Algorithm

Integer non-linear optimization problems are hard to solve. However, this problem can be solved in two steps. The first step uses Fortet's linearization method to linearize the problem introducing new variables but maintaining a polynomial size for the problem. Then, an implicit enumeration algorithm based on Davis-Putnam enumeration algorithm, which is a logic-based branch-and-cut optimization algorithm, is used to find the optimal solution [9, 10]. This method requires the enumeration of all allowed values of the variables as separate Boolean variables. Although this requires an additional step in the problem formulation, it allows for speeding up the optimization algorithm by enumerating only those variable values that are valid feasible solutions and ignoring all other invalid values, i.e., the algorithm will not search through invalid design space regions, if there are any, to save time. Moreover, the accuracy of the solution is affected by the linearization process, however, it is still accurate enough for system-level design. This algorithm is also fast enough to explore many regions of the design space and to make informed system-level decisions.

Numerical Results

Table 4.1 lists the results for several fabric optimization experiments with different traffic models and fabrication technologies. The power dissipations obtained in these experiments are considerably below the values reported in [120] for comparable fabrics. Also, variations in the problem constraints were used, such as adding a lower bound of 32 for the number of ports in experiment number 6 and limiting the pin count per chip to 300 in experiment number 7. It should also be noted that optimizing the components of the fabric at lower levels of abstraction may further reduce the power dissipation.

As mentioned before, typical cell loss probabilities for satellite applications are 10^{-10} or less because retransmissions due to errors or cell loss can not be tolerated due to the large propagation delay. Consequently, validating the D-BMAP traffic model using simulations is almost impossible for low cell loss probabilities because the simulation time grows exponentially as the absolute value of the exponent of the cell loss probability grows linearly. The best we can do is to simulate the system under that traffic model up to a certain feasible value of the cell loss probability and then extrapolate the results for lower cell loss probabilities. This was shown in figure 4.5 and the extrapolated simulation results agreed quite well with the analytical results. Another useful approach that can help to validate the overall effect of the traffic model on the design framework is to test for the sensitivity of the system-level framework to variations in the D-BMAP model parameters used to develop the performance constraints. To this end, we have conducted experiments number 8 through 11 in table 4.1 which were basically repetitions of experiments

2 and 4 except that the parameters of the traffic model were perturbed by a factor of +5% in experiments 8 and 10 and by a factor of -5% in experiments 9 and 11 compared to the actual values in experiments 2 and 4 respectively. The results vary somewhat in terms of the estimated power dissipation values. However, they show a high degree of robustness in terms of the fabric complexity parameters which indicates that the overall framework is not largely susceptible to variations in the traffic model. Therefore, the validation of the traffic model is not that critical to the accuracy and robustness of the overall framework.

Table 4.1: Experimental results for low-power fabric design.

Exp. #	Traffic Model	CLP	Tech. λ	Aggregate Mem. Rate	Link Rate	N	B	K	S	Power Est. (W)
1	IP _{c,1}	10 ⁻¹⁰	0.8 μ m	3.2Gb/s	2.4Gb/s	8	1024	4	10	11.67
2	IP _{c,2}	10 ⁻¹⁰	0.5 μ m	3.2Gb/s	2.4Gb/s	16	1024	4	14	9.98
3	ATM _{c,1}	10 ⁻¹⁰	0.5 μ m	3.2Gb/s	9.6Gb/s	4	2048	6	12	21.82
4	ATM _{c,2}	10 ⁻¹²	0.35 μ m	3.2Gb/s	2.4Gb/s	16	2048	6	12	11.47
5	ATM _{c,2}	10 ⁻¹²	0.35 μ m	4.8Gb/s	2.4Gb/s	32	2048	8	18	24.66
6	ATM _{c,2}	10 ⁻¹⁰	0.35 μ m	3.2Gb/s	622Mb/s	128	2048	12	24	22.84
7	ATM _{c,2}	10 ⁻¹⁰	0.35 μ m	3.2Gb/s	622Mb/s	32	1024	2	8	2.71
8	IP _{c,2}	10 ⁻¹⁰ ^a	0.5 μ m	3.2Gb/s	2.4Gb/s	16	1024	5	14	10.76
9	ATM _{c,2}	10 ⁻¹² ^a	0.35 μ m	3.2Gb/s	2.4Gb/s	16	2048	7	12	13.16
10	IP _{c,2}	10 ⁻¹⁰ ^b	0.5 μ m	3.2Gb/s	2.4Gb/s	16	1024	4	14	9.21
11	ATM _{c,2}	10 ⁻¹² ^b	0.35 μ m	3.2Gb/s	2.4Gb/s	16	2048	6	12	9.89

^a The traffic model used to derive this constraint had its parameters perturbed by a factor of +5%.

^b The traffic model used to derive this constraint had its parameters perturbed by a factor of -5%.

4.3.6 Scalable Architecture Model

Different methods can be considered to extend this framework to accommodate larger scalable switching fabrics. In this section, we will use a strictly non-blocking Clos network, SNB $\mathcal{C}(i, j, n_1, n_2, n_3)$, and we further assume equal number of mod-

ules in both the first and third stages, $n_1 = n_3$, and equal number of inputs to the first stage and outputs of the third stage, $i = j$. One method to extend the optimization framework would be to reformulate all of the detailed equations of the power macro-model and the heuristic equations for the silicon area to take into account all the modules in the Clos network. However, this formulation will add more variables, equations and iterations to many parts of the framework requiring excessively longer times to find different solutions and to explore the design space which defeats the whole purpose of this work. To avoid that, the same framework for single shared memory $M \times N$ fabric modules is used with minor modifications. Note that both M and N are still treated as optimization parameters. First, we assume that the required number of input or output ports, \mathcal{N} , for the scalable fabric is known and constant. With the assumption that $i = j = M$, the number of $M \times N$ modules in the first stage and $N \times M$ modules in the third stage can be found as

$$n_1 = n_3 = \frac{\mathcal{N}}{M} \quad (4.44)$$

The number of the crossbar modules in the second stage is simply equal to the number of the output ports of one shared memory module, N . Next, we need to add another constraint to account for the non-blocking condition as given by equation (3.27) for SNB Clos networks or by equation (3.28) for RNB Clos networks. With the above assumptions, these two conditions reduce to the following

$$N \geq 2M - 1, \quad (4.45)$$

and

$$N \geq M. \quad (4.46)$$

Minor modifications can also be done at this stage to accommodate the requirements of fault tolerance by changing the complexity variables to reflect the redundancies needed.

The formulation of the performance constraints in this case is also a bit different. Figure 4.12 shows the different traffic models through the stages of a Clos network. A D-BMAP model is used to model the aggregate sources for a certain logical queue at the first stage. CLP and other performance measures can be extracted from the distribution of the buffer occupancy in the shared memory pool which is then function of N but not M . Similarly, the traffic coming out of a first stage output can be modeled using a D-MAP model. This traffic goes through the crossbars and is aggregated into a D-BMAP model as it feeds into the logical queues of the third stage. This time, the performance measures are function of M rather than N . Curves similar to those obtained in figure 4.5 are used here to derive the performance constraints. For example, a specification given for the overall CLP of the network would have to be larger than the values derived for shared memory modules both with N and M output ports and weighted by the numbers of modules in the first and third stages, n_1 and n_3 , respectively.

A test case was used to evaluate the error of the estimated power using that method as compared to that using a detailed formulation model. The parameters used were $M = 8$, $B = 1024$, $K = 4$ and $S = 10$ and using a link rate of 2.4Gb/s

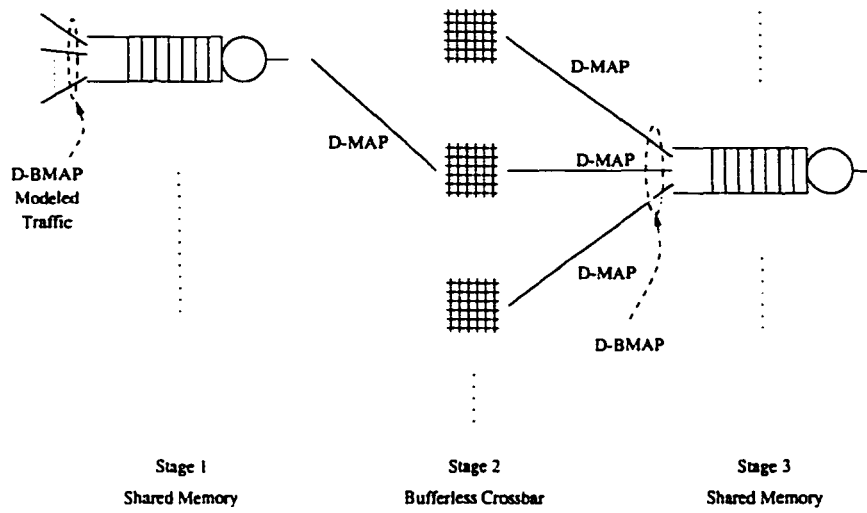


Figure 4.12: The traffic models used to define the performance constraints through the stages of a Clos network.

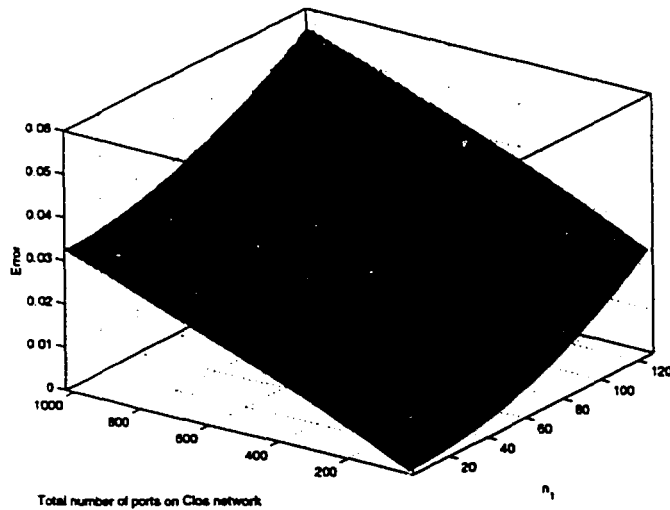


Figure 4.13: Relative error for the power estimates using the shared memory modules framework as compared to a detailed formulation framework (The error surface is fit to the sampled points).

and a $0.5\mu\text{m}$ CMOS technology. Figure 4.13 shows a surface fit to the relative error at sampled points and that surface is within approximately 6% for fabrics with \mathcal{N} up to 1024. This error is small enough to justify the use of this method over the computationally expensive detailed formulation. Moreover, this error can be reduced considerably by taking into account the $N \times n_1^2$ crosspoints of the second stage.

Table 4.2 lists the results for several scalable fabric optimization experiments following the method described above. The time consumed in arriving at any of these results is comparable to that in shared memory module experiments and the framework can still be used to explore the design space for scalable networks.

Table 4.2: Experimental results for scalable low-power fabric design using $\lambda = 0.35\mu\text{m}$, $\text{CLP} = 10^{-10}$ and $\mathcal{N} = 512$.

Exp. #	Traffic Model	Aggregate Mem. Rate	Link Rate	$n_1 = n_3$	$i = j = M$	$n_2 = N$	B	K	S
1	$\text{IP}_{c,2}$	3.2Gb/s	2.4Gb/s	32	16	32	2048	6	28
2	$\text{ATM}_{c,1}$	3.2Gb/s	2.4Gb/s	32	16	32	4096	10	26
3	$\text{ATM}_{c,1}$	4.8Gb/s	2.4Gb/s	64	8	16	4096	16	12
4	$\text{ATM}_{c,2}$	3.2Gb/s	622Mb/s	8	64	128	2048	14	26
5	$\text{ATM}_{c,2}$	3.2Gb/s	622Mb/s	16	32	64	1024	6	14

4.4 Summary

Low-power system-level design of switching fabrics is becoming a necessity with the emergence of on-board satellite switching and advanced VLSI technologies. This work has proposed a framework for system-level design optimization that provides for quick and efficient exploration of the design space without compromising the

design quality. Its power lies in the incorporation of the D-BMAP traffic model with a high-level power macro-model and in matching it to real traffic statistics. The results obtained had remarkable power reductions compared to other methods that do not include traffic models and statistics. This also emphasizes the importance of moving the power optimization process from the circuit-level to the system-level.

Chapter 5

Scheduling and Buffer Management

5.1 Introduction

Scheduling traffic out of a switching fabric is tightly related to how its buffers are managed. Both functions can be implemented using the same hardware as will be discussed throughout this chapter. Scheduling algorithms are used to provide certain quality of service, QoS, guarantees or bounds in terms of bandwidth available to specific connections, packet or cell loss rate, end-to-end delay, delay jitter or combinations of them. For high speed networks, such as ATM networks with small fixed-size cells, the scheduling algorithm must be implemented in hardware in order to be able to make the scheduling decisions within the cell time slot. Considerable work is also being done on variable-size packet networks to allow them to provide QoS using guarantees for persistent connections known as *flows*.

Several scheduling service disciplines were proposed in the literature over the years for switching fabrics that use output buffers [142]. Scheduling algorithms must

possess several desirable features to be useful in practice [129]. These features are generally as follows:

1. Isolation of connections from the undesirable effects of other misbehaving connections. This is necessary even when policing mechanisms are used at the source to shape the traffic since burstiness can accumulate within the network.
2. Low end-to-end delays and to have such bounds dependent on the latency at every scheduler and the source shaping parameters only. The latency is defined as the delay behavior of a connection which should be dependent on the bandwidth reserved for that connection and the connection parameters only.
3. Efficient utilization of the link bandwidth.
4. Fairness in dividing the available link bandwidth among the connections sharing that link. An unfair scheduler may offer widely different service rates to two connections with the same reserved rate over short intervals. Several measures were proposed in the literature to analyze the fairness of schedulers such as the *service fairness index*, SFI [47], and the *worst-case fairness index*, WFI [15, 108].
5. Implementation simplicity in terms of both time and space complexity of the scheduler hardware since the scheduler decisions must be made at a rate comparable to the cell rate.

6. Scalability of the scheduler to cope with a large number of connections as well as a large range of link speeds.

These algorithms can be classified based on the type of guarantees they offer. Some schedulers provide explicit bandwidth guarantees to individual connections which results in implicit end-to-end delay guarantees [125]. However, they do not have independent control of the delay and can not provide explicit delay guarantees. Other schedulers have independent control of both bandwidth and delay bounds but at the expense of increasing complexity [45]. The buffer management unit proposed here targets the first type of schedulers.

Schedulers can also be classified based on their internal structure. They could be either *frame-based* or based on *sorted priority*. Frame-based schedulers divide the time into frames of either fixed or variable length. Reservations for different connections are made in terms of the maximum amount of traffic each connection is allowed to transmit during a frame period. An example of such schedulers is a fixed order round-robin scheduler. Such schedulers may allow the frame length to vary subject to a maximum size and allow a new frame to start whenever a connection has less traffic than its reservation to avoid leaving the server idle. The other type of schedulers, sorted-priority ones, uses a *timestamp* for each cell derived from a global variable known as the *virtual time* associated with the outgoing link. Cells are sorted based on their timestamps and transmitted in that order. The time complexity of sorted-priority schedulers is determined by the complexity of updating the priority list and selecting the highest priority cell and the complexity of computing the timestamp. On the other hand, the space complexity is dominated

by the approach of implementing priority queues.

Many such schedulers are based on the *Generalized Processor Sharing*, GPS, which is an ideal scheduling discipline that uses a fluid traffic model where the cells or packets are infinitely divisible. GPS provides perfect isolation, ideal fairness, low end-to-end connection delays. To make the scheduler more tractable to implementation, a packet-by packet version of GPS is proposed. PGPS or the weighted fair queueing, WFQ, uses a piecewise linear function to approximate the virtual time rather than simulating it in GPS. To simplify further, the *Self-Clocked Fair Queueing* scheduler, SCFQ, avoids the need to maintain the virtual time by approximating it based on the timestamp of the cell currently in service. The price paid is a reduced level of connection isolation causing the end-to-end delay to grow linearly with the number of connections sharing the link. Another scheduler known as the *VirtualClock* uses the real time itself as the virtual time function. This can starve back-logged connections if they received excess bandwidth from the server when other connections were idle.

It is easy to program the proposed architecture to implement many sorted-priority algorithms, however, we are targeting an SCFQ scheduler here as follows:

1. The k^{th} cell of connection i arriving at time a_i^k is tagged with a virtual finishing time F_i^k which is determined from:

$$F_i^k = \frac{1}{\rho_i} + \max \{ F_i^{k-1}, \hat{F}(a_i^k) \} \quad (5.1)$$

and

$$F_i^0 = 0 \quad (5.2)$$

where ρ_i is the offered load of connection i as determined in the connection admission control phase. The cells are then sorted and serviced in increasing order of their virtual finishing times.

2. $\hat{F}(t)$ is the system's virtual time at time t and is equal to the virtual finishing time of the cell receiving service at time t .

This setup provides a per-VC SCFQ scheme although it requires some means to look up the virtual time of the cells at the head of each VC queue. If, on the other hand, the cells are treated per output port, i.e., the subscript i represents an output port rather than a virtual connection and ρ_i is the aggregated offered load for port i , an SCFQ can be achieved for the different outputs of the switch. The disadvantage in this case, however, is the poor isolation between the connections destined to the same output port. Results from this chapter have appeared in [131, 132, 134].

5.2 Buffer Management Using Sorting Queues

Priority control and queueing is needed to implement sorted-priority schedulers and provide the QoS guarantees expected in ATM networks. Two main types of priority exist, delay priority and loss priority. A combination of a head-of-line scheme for delay priority and a push-out scheme for loss priority was proposed for ATM [127]. In the head-of-line priority scheme, a higher priority cell is given service first as

long as there is at least one in the buffer. Push-out priority scheme defines how the cells are lost when the buffer is full. New arrivals can be admitted to a full buffer by pushing out a cell with lower push-out priority, otherwise the new arrival is lost. The queue management unit used in the proposed shared memory switch is based on sorting queues and is capable of implementing a wide range of priority schemes.

5.2.1 Architectures for Sorting Queues

Buffer management using sorting queues was used in [117] to support unicast traffic without any priority schemes. This architecture was modified in [52] to support priority and aging schemes and is modified even further here to provide support for multicasting traffic and to make the sorting time independent of the state of the queue which allows the architecture to support higher ATM traffic rates with the current VLSI technology and with relaxed clocking requirements. A basic sorting queue would be built as a cascade of several *sorting units* such that there are two paths between the units, downstream and upstream paths. A cell arriving from one of the switch input ports is tagged in the tagging unit, stored in a memory location whose address is provided by the memory controller while its tag is fed into the queue management unit where it will be stored and later dispatched. The number of sorting units in the queue is the same as the number of cells that the shared memory can hold. If the shared memory is expanded, the queue can also be expanded by cascading more sorting units. The sorting queue architecture proposed in [25] uses a synchronous common bus for all sorting units which limits the speed and prevents the dynamic sorting of cells while in [52,117] only unicast traffic is supported within the same queue and in [52] the sorting time in the queue was dependent on the

queue state such that a new tag can be fed into the queue only after the previous sorting operation ends. For a sorting operation to end within a cell time in that architecture, the sorting units must have extremely low latency specially as the queue length gets longer. This latency can not be achieved for that architecture to support high ATM traffic rates with the current VLSI technology. That architecture also uses serial streams of data which makes the timing constraints and clocking requirements even tougher. Another approach uses FIFOs as in [138] or CAM memories as in [26, 119] as searchable queues. Several queues are used to queue per-VC traffic or traffic destined to different output ports separately and a separate controller was used to implement the scheduling functionality through a common bus. Sequential search and the common bus limit the speed and the scheduling is limited to fewer priority classes and fewer scheduling algorithms.

5.2.2 The Proposed Sorting Queue

Our objective is to design a sorting queue that makes use of the reduced memory access rate for the shared memory, supports different priority and scheduling schemes, supports multicasting traffic and allows for a small and simple VLSI implementation. To realize such an objective, the algorithm shown in figure 5.1 is used by the sorting units to sort the tags. The tag holds the memory address of the associated ATM cell in the buffer, the destination output port for that cell and some flags to control and indicate the priority and age of the cell.

As each tag travels downstream, its address is compared to the ones in the upstream registers and if the upstream address is larger than its own address or if they are equal but the downstream tag has a higher priority and/or a lower age,

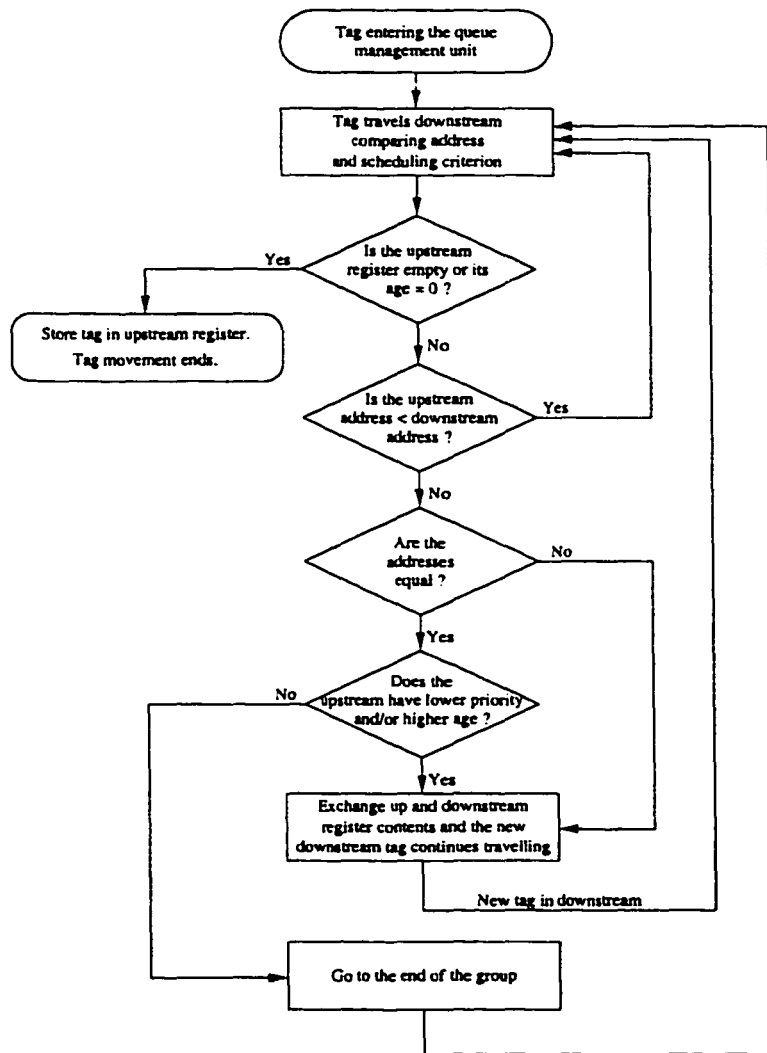


Figure 5.1: The control algorithm of the sorting units.

according to the scheduling algorithm used, the contents of the sorting unit registers are exchanged and the new cell in the downstream register continues traveling downstream. In this way, tags will be sorted into groups where each group has a single tag per output port. Across consecutive groups, the tags from different streams but destined to the same output port are sorted according to their priority and/or age using the scheduling algorithm implemented. The address comparisons in the algorithm are based on the lowest port address in the tag if that tag corresponds to a multicast cell. Multicast traffic support will be discussed in detail in section 5.2.4. Most of the conditions used in the sorting process can be reprogrammed online to change the scheduling algorithm used. This also provides a *push out* mechanism that helps implementing a *complete sharing with virtual partitioning* allocation scheme [136]. In this scheme, all traffic streams are allowed to share the whole buffer of the switch within an output port logical queue. However, tags belonging to streams that are over-utilizing the buffer space at the expense of other streams will be pushed out of the buffer when those other streams need that buffer space.

Each sorting unit in the sorting queue has the basic structure shown in figure 5.2. The comparison and decoding logic operates according to a sorting algorithm that sorts incoming tags based on their priority, time stamp or age, and other scheduling criteria and based on the need to handle multicasting traffic. This is very similar to the sorting queue in [52] but in order to support the reduced memory access rate of the shared buffer, and in order to have reasonable timing constraints and clock periods for the sorting queue, the concept of a circular queue is used, as

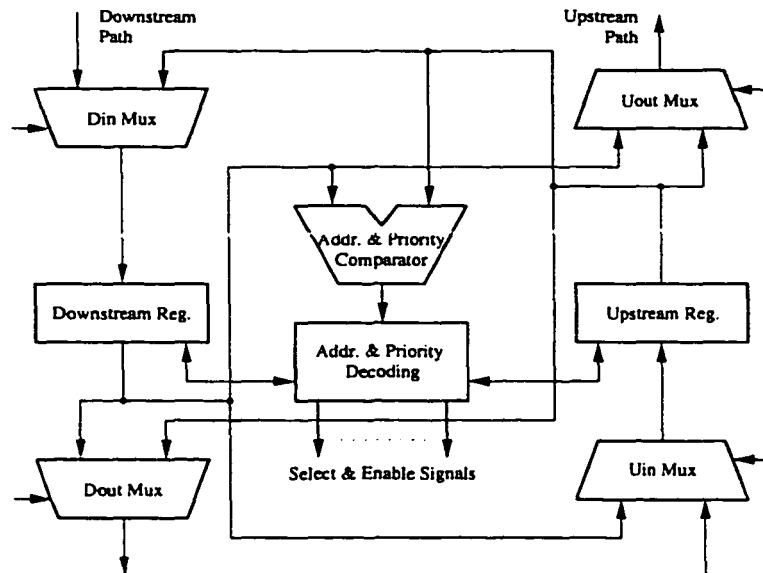


Figure 5.2: The basic structure of a sorting unit.

shown in figure 5.3. In this case, the head of the queue that is used to feed the downstream moves to the next group every cell time slot. Similarly, the head of the queue feeding the upstream moves but this time it moves, on the worst case, N times in a cell slot from one sorting unit to the next so that the largest possible group of N tags can be dispatched within a cell time slot. The upstream head is always one group ahead of the downstream head, this ensures that the tags of the output group has already settled in the upstream registers. The head pointers can be controlled using simple shift registers and the sorting queue can still be cascaded to support larger shared memory buffers.

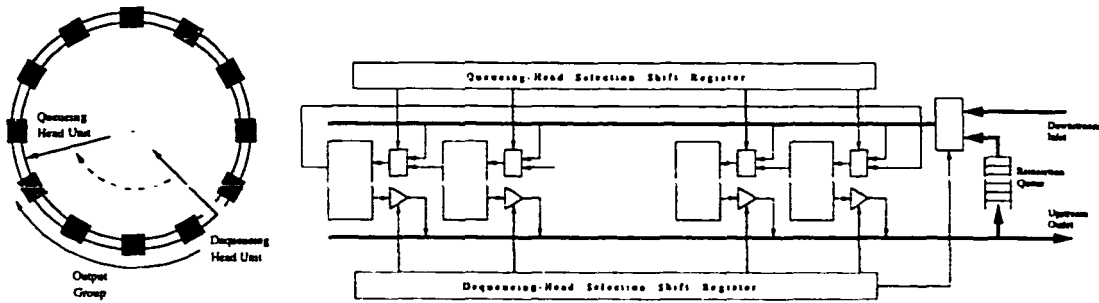


Figure 5.3: Architecture of the circular-queue management unit.

5.2.3 Design of the Tag Structure

The structure for the tag cells is shown in figure 5.4. The output address is used to support both unicast and multicast traffic and consists of N bits, where N is the number of output ports. Each bit in the output address in the tag denotes one of the output ports that the cell associated with the tag is destined to. The priority field is used to support different priority classes for different traffic streams. In its simplest form, it would only be used to mark cells that exceed the rate allowed by access control so that they would be discarded first in case of congestion.

Also, the queueing delay should be bounded for real-time traffic. In other words,

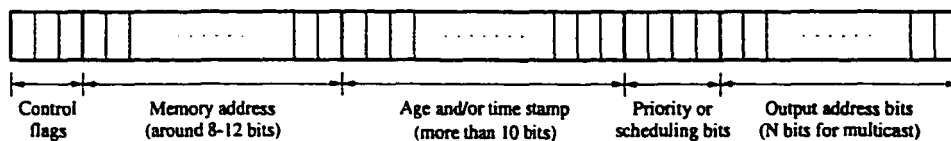


Figure 5.4: The tag cell structure.

if the cells of a real-time traffic exceed a certain value, these cells are not useful anymore and can be discarded. For scheduling schemes that support such discarding techniques, an age or time stamp field is provided. In the tagging unit, this field is assigned the maximum value before the tag is queued in the sorting queue. In the upstream, this field decrements by one in every cell time slot and when it reaches zero, and if the scheduling algorithm uses it, the tag is invalidated or timed-out and can be replaced by any appropriate downstream tag. The number of bits in this field is determined to be 12 bits based on the simulation results of the maximum delay exhibited in the sorting queue as shown in figure 5.5-d. These simulation results will be further detailed in the next section. For the proposed architecture, the tags are 38 bits wide.

5.2.4 Multicast Support

Multicast traffic support is a key feature in broadband networks. A switch can provide this support in different ways depending on the write and read operations involved [73]. Multicast can use a multiple-write, multiple-read (MWMR) approach in which the multicast cell is split into several copies and each of them is treated like a unicast cell. The obvious advantage is the fairness of the scheme while the obvious disadvantage is the large buffer space used by the copies which could be of order $O(N^2)$. Another approach is based on single-write, multiple-read (SWMR) where the buffer space is only of order $O(N)$ due to the single-write operation, however, the multiple-read operation has to be done sequentially which limits the approach to switches of small sizes where the memory bandwidth can support that. Finally, there is the single-write, single-read (SWSR) approach which again

saves on buffer space and allows for replication at output during a single time slot using a common bus or a copy network. The disadvantage is the need of additional hardware to do the replication at the output. Another point to consider in SWSR is the contention between unicast cells and multicast cells. This can be handled using separate output buffers to store the contending cells or using output masking where one of the contending cells is reinserted back into the shared memory according to their priority.

A bursty ON-OFF traffic model is used to evaluate the above multicasting schemes. The source alternates between a busy period and an idle period, where the length of the busy period is geometrically distributed with parameter $\alpha, 0 \leq \alpha \leq 1$ such that,

$$M(i) = \alpha(1 - \alpha)^{i-1}, i \geq 1 \quad (5.3)$$

and the average burst length is given by

$$A = \sum_{i=1}^{\infty} iM(i) = \frac{1}{\alpha} \quad (5.4)$$

Similarly, the length of the idle period is geometrically distributed with parameter $\beta, 0 \leq \beta \leq 1$. Thus, it has the distribution

$$N(i) = \beta(1 - \beta)^i, i \geq 0 \quad (5.5)$$

and the average idle period is

$$B = \sum_{i=1}^{\infty} iN(i) = \frac{1 - \beta}{\beta} \quad (5.6)$$

The offered load to any input port is obtained from

$$p = \frac{A}{A + B} \quad (5.7)$$

Similar to the hot-spot traffic model, the multicast traffic can be represented as a fraction, h , of the total traffic at an input port. Additionally, a multicast cell is assumed to have x destinations, where X is a random variable with a truncated geometric distribution with parameter r such that

$$\Pr[X = k] = \frac{(1 - r)r^{k-1}}{1 - r^N} \quad 1 \leq k \leq N \quad (5.8)$$

Hence, the effective offered load will be

$$p_{\text{eff}} = p(1 - h) + phE[X] \quad (5.9)$$

where

$$E[X] = \frac{1}{1 - r} - \frac{Nr^N}{1 - r^N} \quad (5.10)$$

Figure 5.5 shows the results of simulating an 8×8 shared memory switch, i.e., $N = 8$, with a total buffer size of 256 cells. The traffic model used has a mixture of 10% multicast and 90% unicast traffic, i.e., $h = 0.1$, and an average burst length of 8 cells. The simulation results in figures 5.5-a and 5.5-b show that SWSR with output masking provides the highest throughput and the lowest cell loss probability for a given buffer size. The cost that has to be paid, however, is an increased cell delay. This is due to the fact that with this scheme the shared memory is used more efficiently and the cells are allowed to stay longer in it without being lost.

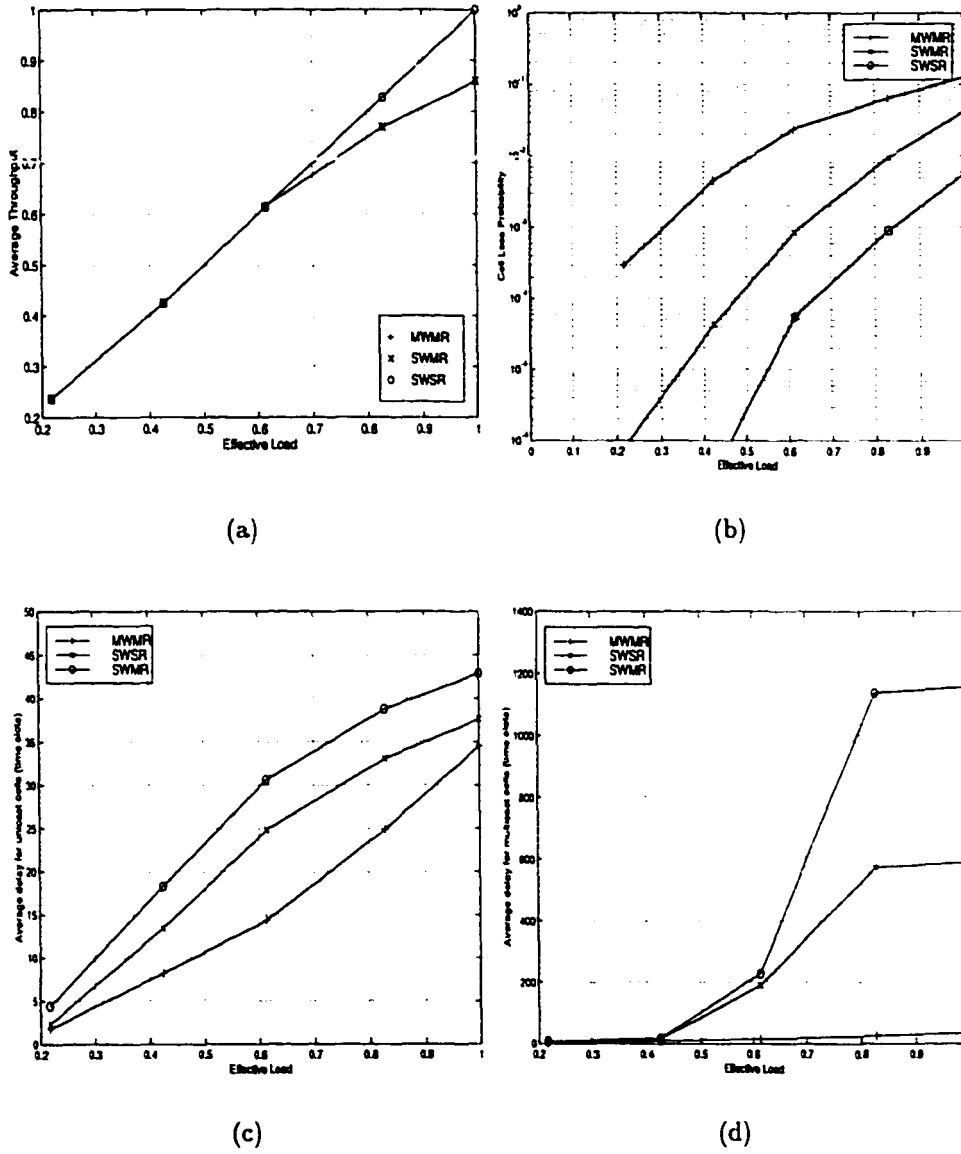


Figure 5.5: Simulation results for different multicasting traffic management schemes. Effective load at the inputs versus: (a) average throughput, (b) cell loss probability, (c) unicast cells average delay, and (d) multicast cells average delay.

The sorting queue proposed here is designed to support the SWSR scheme with output masking. Every ATM cell slot is divided into two phases. During the first phase, N tags are queued in through the queueing head unit while another N tags are dequeued using the dequeuing head unit that is one group ahead. This gives the queued tags traveling downstream a chance to settle in the units of the first group before they are dequeued. When the next ATM slot starts, the queueing head is moved to the end of the group on top of the queue while the dequeuing head would be already pointing to the beginning of that top group. If a multicast tag is dequeued, its corresponding ATM cell is dispatched to the available output ports. If it contends with any other unicast or multicast cells for a certain output port, priority and age are used to resolve these contentions and mask the losers. During the second phase of the time slot, the losers are requeued in the sorting queue using the reinsertion queue shown in figure 5.3. This reinsertion queue will have to hold only N tags on the worst case. Using these two phases, the sorting units require $2N$ accesses during a time slot. If the access time for the queue is equal to or less than half of that of the shared memory blocks then we can still use the reduced number of accesses, N , for the shared memory.

The design can be easily modified to support SWMR if we restrict the sorting algorithm so that it treats multicast tags as if they were unicast tags destined to the lowest port number in their port address field. After dispatching the cell to that port with the lowest number, the bit for that port is reset in the tag and the tag is reinserted in the sorting queue while the cell itself is retained in the shared memory until it is sent to all ports. This will eliminate the need for contention

resolution or masking at the output ports. However, the main disadvantage is that the average delay in the queue will increase dramatically as has been shown in the simulation results in figures 5.5-c and 5.5-d and this might be intolerable for many real-time traffic streams. This is mainly because SWSR with output masking uses all available ports for multicast tags and only reinserts those that lose contention, while SWMR reinserts multicast tags for all ports in a sequence of slots wasting more resources and increasing the delay.

Figure 5.6 illustrates an example of SWSR multicasting using the sorting queue proposed. The multicast cell queued in the first time slot shown is destined to ports 1, 3, 5, 6 and 7. Assuming that the queue is empty, the tags for the cells queued during this slot will constitute the top group on the queue and this group will be dequeued in the second time slot. The multicast cell will contend with three unicast cells for output ports number 5, 6 and 7. Assuming that the unicast cells destined to ports 5 and 7 have higher priority than the multicast cell while the unicast cell destined to port 6 has lower priority, the reinsertion queue will reset bits number 1, 3 and 6 in the address field of the tag for the multicast cell and requeue that tag in the sorting queue with bits 5 and 7 retained. It will also requeue the tag for the unicast cell destined to port 6. The multicast cell and the unicast cell destined to port 6 will both be retained in the shared memory until they have been dispatched to all of their destinations.

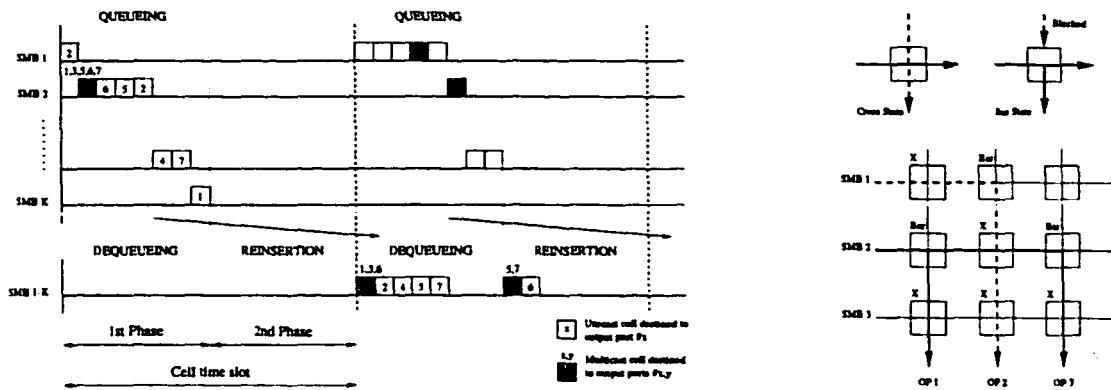


Figure 5.6: An example of multicasting using SWSR with output masking from a shared memory block to several output ports.

5.3 Analysis of Priority Queues

Combined space and delay priorities were analyzed using an $M/M/1/K$ model in [127], using an $M/D/1/K$ model in [49] and using an $M/G/1/K$ model in [126]. However, in all of these cases, only two priority classes were considered to make the analysis tractable. A more realistic analysis, should consider more than two priority classes. For example, CBR and rt-VBR traffic can have certain priority requirements that differ from the nrt-VBR requirements, which in turn differ from those for ABR traffic. In this section, the $M/G/1/K$ model [126] will be extended to accommodate M traffic classes, X_1, X_2, \dots, X_M . Then, the model will be solved using the supplementary variable method [36].

5.3.1 Multi-class Priority Analysis

Assume that class i arrivals are Markovian with a rate λ_i , and all classes receive service with probability density function $g(y)$ and distribution function $G(y)$ for the service time. The buffer maximum capacity is B and is shared by all classes. In [126], a push-out parameter was used but the results showed better performance without that parameter (i.e., if its value was 1), so we will do without it. Also, two combined-priority models can be discussed.

Consider two classes i, j where $i > j$. Model A assumes that class i has both higher loss and delay priorities over class j and model B assumes that class i has higher loss priority and lower delay priority over class j . The proposed sorting queue architecture implements both models. An ATM switch serving VBR, CBR, and ABR traffic classes would normally use model B. However, in the case of satellite switches, our numerical results in section 5.3.2 show that model A can also be used to take advantage of the better cell loss performance of that model and the delay characteristics would not be affected much due to the large propagation delays¹ in that application compared to queuing delay of all classes.

Assume that the probability of having an idle server at time t is denoted by $p(t)$ and the probability of having an occupancy in the buffer of $X_1 = x_1, X_2 = x_2, \dots, X_M = x_M$ cells at time t is denoted by $q(x_1, x_2, \dots, x_M, t, y) dy$ where the

¹For satellite switching applications and depending on the orbit, the switching delay will have a varying effect on the overall delay, propagation and switching. For example, the one-way propagation delay for a geostationary orbit is about 125 msec while the switching delay, including queuing delay, is negligible compared to the propagation delay. For a low earth orbit, typically below 3125 miles, however, the propagation time reduces to a few milliseconds but for high traffic rates this is still very large compared to the switching and queuing delay.

remaining service time for the cell in the server is within the interval $(y, y + dy)$. The inclusion of the single supplementary variable y in the specification of the state of the system makes the whole process Markovian in continuous time [36], and the equations describing the process can be written in the usual way by considering the transitions occurring in Δt .

As $\Delta t \rightarrow 0$, we get the differential equations presented below. The equations would simplify considerably if they have rational Laplace transforms and that justifies the use of the complex transition probabilities.

$$\frac{dp(t)}{dt} = - \left(\sum_{i=1}^M \lambda_i \right) p(t) + q(0, \dots, 0, t, 0). \quad (5.11)$$

$$\begin{aligned} \frac{\partial q(0, \dots, 0, t, y)}{\partial t} - \frac{\partial q(0, \dots, 0, t, y)}{\partial y} = & \\ & - \left(\sum_{i=1}^M \lambda_i \right) q(0, \dots, 0, t, y) \\ & + \sum_{i=1}^M q(0, \dots, x_i = 1, \dots, 0, t, 0) g(y) \quad (5.12) \\ & + \left(\sum_{i=1}^M \lambda_i \right) p(t) g(y). \end{aligned}$$

$$\begin{aligned}
& \frac{\partial q(x_1, x_2, \dots, x_M, t, y)}{\partial t} - \frac{\partial q(x_1, x_2, \dots, x_M, t, y)}{\partial y} = \\
& - \left(\sum_{i=1}^M \lambda_i \right) q(x_1, x_2, \dots, x_M, t, y) \\
& + \sum_{i=1}^M \bar{\delta}_{x_i, 0} \lambda_i q(x_1, x_2, \dots, x_i - 1, \dots, x_M, t, y) \\
& + \sum_{i=1}^M \left(\prod_{j=i+1}^M \bar{\delta}_{j, M+1} \bar{\delta}_{j, i} \delta_{x_j, 0} \right) q(x_1, \dots, x_{i-1}, x_i + 1, 0, \dots, 0, t, 0) g(y), \\
& 1 \leq \sum_{i=1}^M x_i \leq B - 1. \quad (5.13)
\end{aligned}$$

$$\begin{aligned}
& \frac{\partial q(x_1, x_2, \dots, x_M, t, y)}{\partial t} - \frac{\partial q(x_1, x_2, \dots, x_M, t, y)}{\partial y} = \\
& - \sum_{i=2}^M \left(\prod_{j=1}^{i-2} \bar{\delta}_{j, 0} \bar{\delta}_{j, i-1} \delta_{x_j, 0} \right) \left(\sum_{k=i}^M \lambda_k \right) q(0, \dots, 0, x_{i-1}, x_i, \dots, x_M, t, y) \\
& + \sum_{i=1}^M \lambda_i q(x_1, \dots, x_i - 1, \dots, x_M, t, y) \\
& + \sum_{i=2}^M \sum_{j=1}^{i-1} \lambda_i q(x_1, \dots, x_j + 1, \dots, x_i - 1, \dots, x_M, t, y) g(y), \\
& \sum_{i=1}^M x_i = B \text{ and } 1 \leq x_i \leq B - 1 \text{ for } i = 1, \dots, M. \quad (5.14)
\end{aligned}$$

$$\begin{aligned}
& \frac{\partial q(0, \dots, x_i = B, \dots, 0, t, y)}{\partial t} - \frac{\partial q(0, \dots, x_i = B, \dots, 0, t, y)}{\partial y} = \\
& - \left(\sum_{j=i+1}^M \lambda_j \right) \bar{\delta}_{i,M} q(0, \dots, x_i = B, \dots, 0, t, y) \\
& + \lambda_i \sum_{j=1}^{i-1} \bar{\delta}_{j,0} q(0, \dots, x_j = 1, \dots, x_i = B - 1, \dots, 0, t, y) \quad (5.15) \\
& + \lambda_i q(0, \dots, x_i = B - 1, \dots, 0, t, y), \quad i = 1, \dots, M.
\end{aligned}$$

where $\delta_{i,j} = 1$ if $i = j$, otherwise 0 and $\bar{\delta}_{i,j} = 1 - \delta_{i,j}$. Equations (5.11) and (5.12) are for an empty queue and a server at the end of a busy period respectively. The case of a queue that is not full is modeled by equation (5.13) and the case of a full one is modeled by equation (5.14). Finally, equation (5.15) models the boundary conditions.

The differential equations would be exactly the same for model B except for equation (5.13) where the last term would be replaced by the following term:

$$\sum_{i=1}^M \left(\prod_{j=1}^{i-1} \bar{\delta}_{j,0} \bar{\delta}_{j,i} \delta_{x_j,0} \right) q(0, 0, \dots, 0, x_i + 1, x_{i+1}, \dots, x_M, t, 0) g(y)$$

At steady-state and as $t \rightarrow \infty$, P and $q(x_1, x_2, \dots, x_M, y)$ would refer to the equilibrium conditions such that:

$$P = \lim_{t \rightarrow \infty} p(t),$$

$$q(x_1, x_2, \dots, x_M, y) = \lim_{t \rightarrow \infty} q(x_1, x_2, \dots, x_M, t, y).$$

Now, the Laplace transform for the differential equations of the model at equilibrium is found where $q^*(x_1, x_2, \dots, x_m, s)$ is the transform of $q(x_1, x_2, \dots, x_m, y)$ and $G^*(s)$ is the Laplace-Stieltjes transform² of $G(y)$.

$$\left(\sum_{i=1}^M \lambda_i \right) P = q(0, \dots, 0, y = 0). \quad (5.16)$$

$$\begin{aligned} -sq^*(0, \dots, 0, s) + q(0, \dots, 0, y = 0) = \\ - \left(\sum_{i=1}^M \lambda_i \right) q^*(0, \dots, 0, s) \\ + \sum_{i=1}^M q(0, \dots, x_i = 1, \dots, 0, y = 0) G^*(s) \quad (5.17) \\ + \left(\sum_{i=1}^M \lambda_i \right) P G^*(s). \end{aligned}$$

²It will reduce to a regular Laplace transform if $G(y)$ is continuous.

$$\begin{aligned}
& -sq^*(x_1, x_2, \dots, x_M, s) + q(x_1, x_2, \dots, x_M, y = 0) = \\
& - \left(\sum_{i=1}^M \lambda_i \right) q^*(x_1, x_2, \dots, x_M, s) \\
& + \sum_{i=1}^M \bar{\delta}_{x_i, 0} \lambda_i q^*(x_1, x_2, \dots, x_i - 1, \dots, x_M, s) \\
& + \sum_{i=1}^M \left(\prod_{j=i+1}^M \bar{\delta}_{j, M+1} \bar{\delta}_{j, i} \delta_{x_j, 0} \right) q(x_1, \dots, x_{i-1}, x_i + 1, 0, \dots, 0, y = 0) G^*(s), \\
& \qquad \qquad \qquad 1 \leq \sum_{i=1}^M x_i \leq B - 1. \quad (5.18)
\end{aligned}$$

$$\begin{aligned}
& -sq^*(x_1, x_2, \dots, x_M, s) + q(x_1, x_2, \dots, x_M, y = 0) = \\
& - \sum_{i=2}^M \left(\prod_{j=1}^{i-2} \bar{\delta}_{j, 0} \bar{\delta}_{j, i-1} \delta_{x_j, 0} \right) \left(\sum_{k=i}^M \lambda_k \right) q^*(0, \dots, 0, x_{i-1}, x_i, \dots, x_M, s) \\
& + \sum_{i=1}^M \lambda_i q^*(x_1, \dots, x_i - 1, \dots, x_M, s) \\
& + \sum_{i=2}^M \sum_{j=1}^{i-1} \lambda_i q^*(x_1, \dots, x_j + 1, \dots, x_i - 1, \dots, x_M, s) G^*(s), \\
& \qquad \qquad \qquad \sum_{i=1}^M x_i = B \text{ and } 1 \leq x_i \leq B - 1 \text{ for } i = 1, \dots, M. \quad (5.19)
\end{aligned}$$

$$\begin{aligned}
& -sq^*(0, \dots, x_i = B, \dots, 0, s) + q(0, \dots, x_i = B, \dots, 0, y = 0) = \\
& - \left(\sum_{j=i+1}^M \lambda_j \right) \bar{\delta}_{i,M} q^*(0, \dots, x_i = B, \dots, 0, s) \\
& + \lambda_i \sum_{j=1}^{i-1} \bar{\delta}_{j,0} q^*(0, \dots, x_j = 1, \dots, x_i = B-1, \dots, 0, s) \quad (5.20) \\
& + \lambda_i q^*(0, \dots, x_i = B-1, \dots, 0, s), \quad i = 1, \dots, M.
\end{aligned}$$

A recursive solution is described in [126] by which $q(x_1, x_2, \dots, x_M, y = 0)$, the probability distribution for the buffer occupancy, and $q^*(x_1, x_2, \dots, x_M, s = 0)$, the stationary probability that the server is busy given a certain buffer occupancy, can be derived. Then, the probability distribution for the buffer occupancy immediately after departure instants denoted by $q(x_1, x_2, \dots, x_M)$, can be defined as

$$q(x_1, x_2, \dots, x_M) = \frac{q(x_1, x_2, \dots, x_M, y = 0)}{\sum_{x_1=0}^B \sum_{x_2=0}^{B-x_1} \sum_{x_3=0}^{B-x_1-x_2} \dots \sum_{x_M=0}^{B-\sum_{i=1}^{M-1} x_i} q(x_1, x_2, \dots, x_M, y = 0)} \quad (5.21)$$

Hence, the loss probability for class i is given by equation (5.22) where the first term represents losses of class i when there are no lower loss priority cells that can be pushed out and the second term represents losses when there is at least one cell of class i in the buffer and higher loss priority arrivals push it out. The equation reduces to one term for class M , the highest loss priority class, and that is the probability of having a buffer full of class M cells when no cells can be pushed out

as shown in equation (5.23).

$$P_{L_i} = \frac{1}{\lambda_i} \left[\lambda_i \sum_{x_i=0}^B \sum_{x_{i+1}=0}^{B-x_i} \cdots \sum_{x_{M-1}=0}^{B-\sum_{j=i}^{M-2} x_j} q^*(0, 0, \dots, 0, x_i, \dots, x_{M-1}, B - \sum_{k=i}^{M-1} x_k, s = 0) \right. \\ \left. + \left(\sum_{l=i+1}^M \lambda_l \right) \sum_{x_i=1}^B \sum_{x_{i+1}=0}^{B-x_i} \cdots \sum_{x_{M-1}=0}^{B-\sum_{j=i}^{M-2} x_j} q^*(0, 0, \dots, 0, x_i, \dots, x_{M-1}, B - \sum_{k=i}^{M-1} x_k, s = 0) \right],$$

for $i = 1, \dots, M - 1$ (5.22)

$$P_{L_M} = q^*(0, 0, \dots, 0, x_M = B, s = 0) \quad (5.23)$$

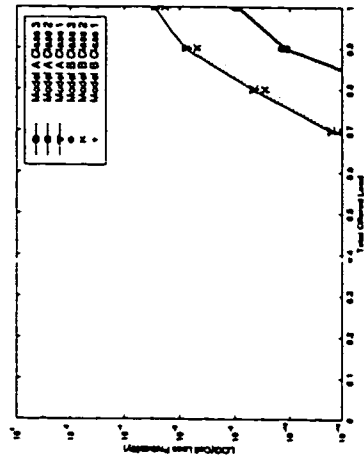
The mean waiting time for class M , the highest delay priority class in model A, is found by Little's law to be as shown in equation (5.24). For each of the remaining classes, the mean waiting time can be computed for all cells including those lost by push-out. This mean waiting time will always be larger than that for those cells that completed service. Hence, the expression in (5.25) is an upper bound for the mean waiting time for class i .

$$W_M = \frac{1}{\lambda_M(1 - P_{L_M})} \sum_{x_1=0}^B \sum_{x_2=0}^{B-x_1} \sum_{x_3=0}^{B-x_1-x_2} \cdots \\ \sum_{x_{M-1}=0}^{B-\sum_{j=1}^{M-2} x_j} x_M q^*(x_1, x_2, \dots, x_{M-1}, B - \sum_{k=1}^{M-1} x_k, s = 0), \quad (5.24)$$

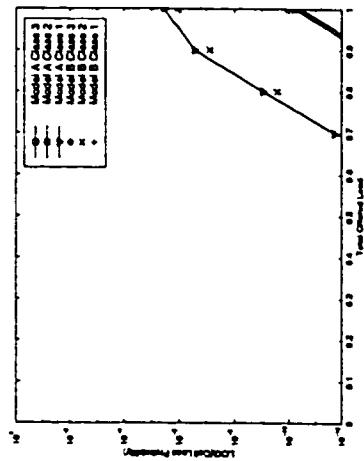
$$\begin{aligned}
W_i \leq & \frac{1}{\lambda_i(1 - P_{L_i})} \sum_{x_1=0}^B \sum_{x_2=0}^{B-x_1} \cdots \sum_{x_i=1}^{B-\sum_{j=1}^{i-1} x_j} \cdots \\
& \sum_{x_{M-1}=0}^{B-\sum_{j=1}^{M-2} x_j} x_i q^*(x_1, x_2, \dots, x_{M-1}, B - \sum_{k=1}^{M-1} x_k, s = 0), \\
& \text{for } i = 1, \dots, M - 1 \quad (5.25)
\end{aligned}$$

5.3.2 Numerical Results

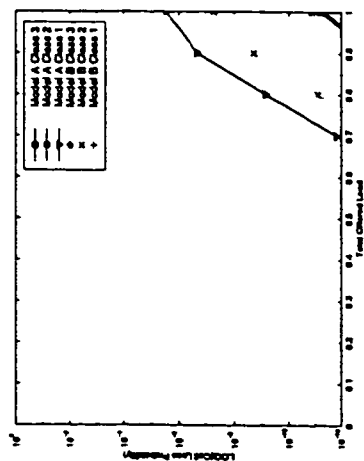
The case of three priority classes is studied numerically to evaluate the priority sorting queue performance and estimate the buffer size needed for the switch. Figure 5.7 shows model A for three traffic classes where X_1 , X_2 and X_3 represent ABR, nrt-VBR and CBR/rt-VBR respectively. Model A is used to analyze satellite switches as explained in the previous section. The origin and transitions out of it are modeled by equations (5.11) and (5.12) while the space confined by the triangle ABC and the $X_1X_2X_3$ planes is modeled by equation (5.13) which represents a buffer that is not full. Equation (5.14) models the surface of the triangle ABC which represents a full buffer and where the push-out occurs. Finally, equation (5.15) models the vertices of the triangle where the buffer can be full with single class cells. To further illustrate the push-out mechanism, assume that at one instant the buffer gets full and its state is represented by point F on the surface of the triangle ABC . If the following arrivals are all during a service time and are all of class X_2 , F will move in direction 1 towards F' where all X_1 cells would have been pushed-out by X_2 cells. If X_3 cells constitute all of the following arrivals, they will push-out X_2 cells



(c) $9\lambda_1 = 3\lambda_2 = \lambda_3$

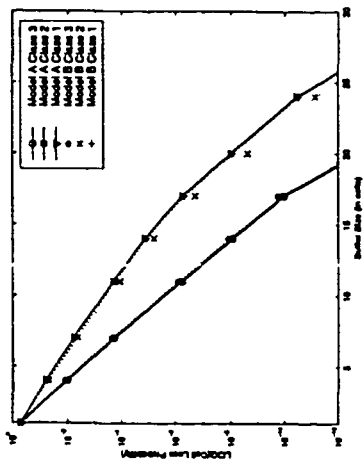


(b) $\lambda_1 = \lambda_2 = \lambda_3$

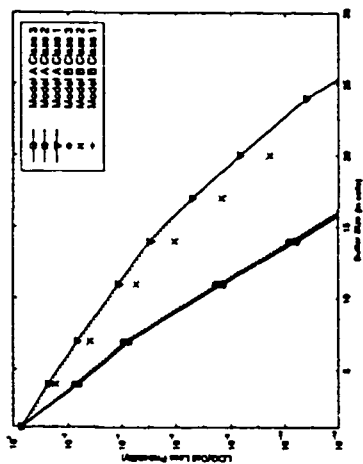


(a) $\lambda_1 = 3\lambda_2 = 9\lambda_3$

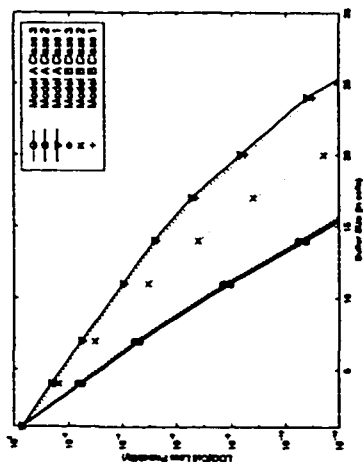
Figure 5.8: Cell loss probability vs. total offered load.



(a) $\lambda_1 = 3\lambda_2 = 9\lambda_3$

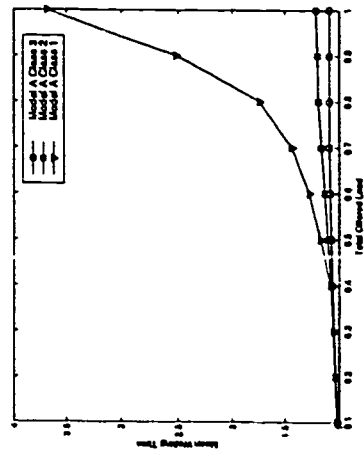


(b) $\lambda_1 = \lambda_2 = \lambda_3$

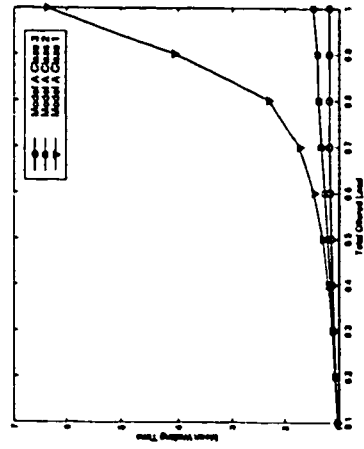


(c) $9\lambda_1 = 3\lambda_2 = \lambda_3$

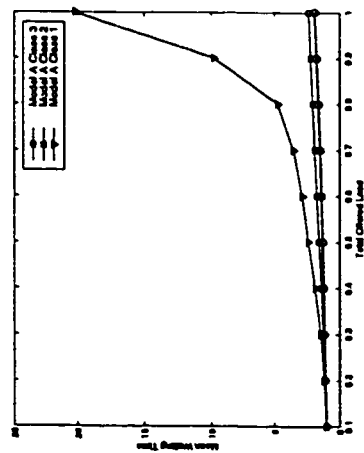
Figure 5.9: Cell loss probability vs. buffer size.



(a) $\lambda_1 = 3\lambda_2 = 9\lambda_3$

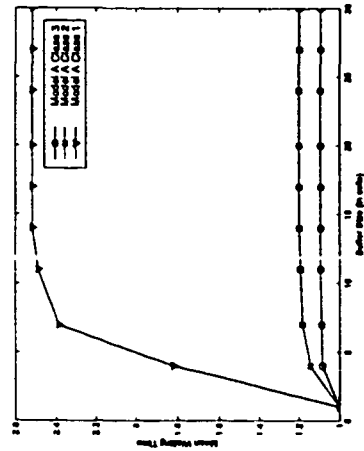


(b) $\lambda_1 = \lambda_2 = \lambda_3$

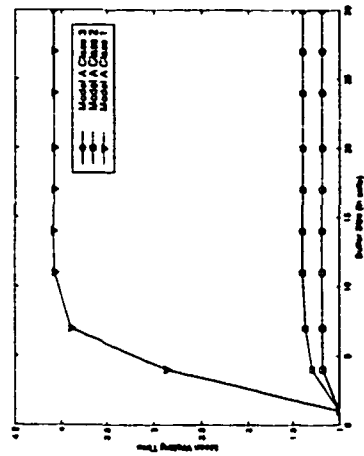


(c) $9\lambda_1 = 3\lambda_2 = \lambda_3$

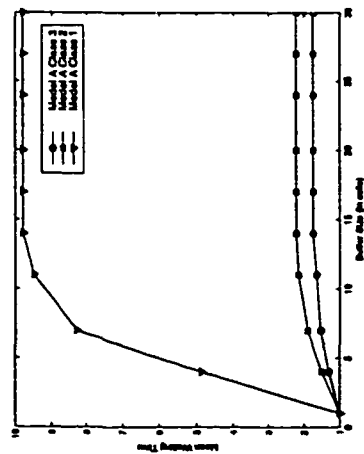
Figure 5.10: Mean waiting time vs. total offered load.



(a) $\lambda_1 = 3\lambda_2 = 9\lambda_3$



(b) $\lambda_1 = \lambda_2 = \lambda_3$



(c) $9\lambda_1 = 3\lambda_2 = \lambda_3$

Figure 5.11: Mean waiting time vs. buffer size.

It is obvious that the traffic classes with high loss priority exhibit the least cell loss probability, specially in model A, even when their arrival rates are higher than the arrival rates of low loss priority classes. On the other hand, the delay experienced by the classes with the low loss priority, class 1, of model A is the largest. This shows that model A is suitable for satellite applications where the propagation delay is so large such that the queueing delay is negligible compared to it even for class 1 of model A. A low cell loss probability is increasingly important for satellite switches, 10^{-10} or less is a typical value, since any retransmissions will incur large delay because of the inherent latency of automatic repeat request systems. Hence, the loss characteristics in figure 5.9 are the most important ones. They are used to choose a suitable buffer size for the switch that achieves the cell loss probabilities required for global broadband networks.

5.4 VLSI Implementation

The guidelines that should be followed to overcome the limitations imposed by the satellite payload and space environment on the physical implementation of the switching system in general and every one of its components including the buffer management unit were summarized in section 3.2. As a step towards achieving these requirements, an ASIC prototype for the buffer management controller with 24 sorting units was implemented in a $0.35\ \mu\text{m}$ CMOS technology and a deep sub-micron design flow. A microphotograph of the chip is shown in figure 5.12. This prototype shows that it is feasible to implement the proposed controller on a single chip. The area needed, excluding the pads ring, for 24 sorting units is $\approx 3.5\ \text{mm}^2$

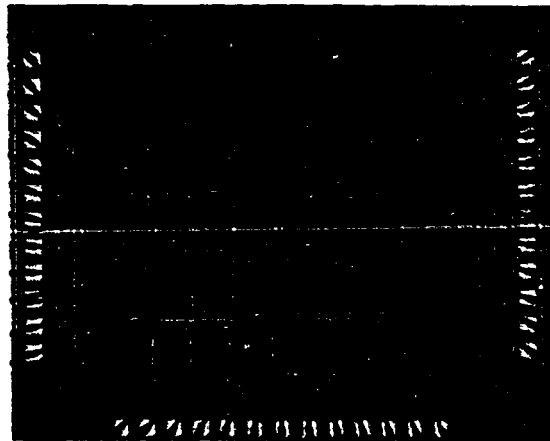


Figure 5.12: Microphotograph of the prototype of the queue management unit.

and for the control logic and reinsertion queue is $\approx 1.5 \text{ mm}^2$. For 160 units³, the core area would be around 27 mm^2 . The design used static standard cells characterized to reduce the power. It also shows a dynamic power consumption of 3.4 mW for each sorting unit at a clock frequency of 100 MHz. This power is 21% less than its value before manipulating the design to reduce the switching activity. Also, a power management technique based on clock gating is used to shut off any unused sorting units. The statistically estimated reduction in the total chip power is 38%. At a clock frequency of 100 MHz, multicast traffic could be supported successfully for an aggregate traffic rate of 21.2 Gbps. This traffic rate is sufficient to support about 40 ports, each operating at 500 Mbps and using MF-TDMA for satellite applications.

Due to the increased speed and complexity of the design, the chip must be

³Corresponding to 8 ports and loss rates of 10^{-10} for typical traffic in a satellite environment.

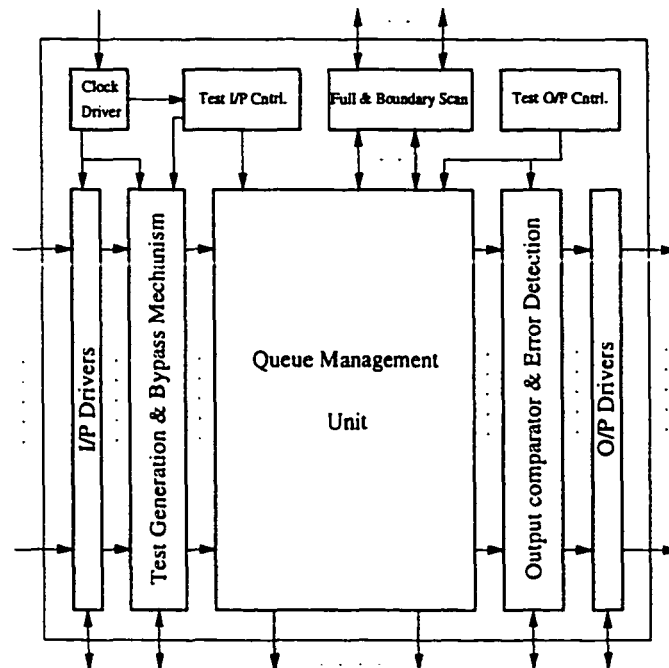


Figure 5.13: Block diagram of the internal testing structures in the chip.

equipped with additional test logic and separate test access ports in order to keep testing manageable. This is a requirement for design validation and testing, therefore, boundary scan elements at the I/O pins and a full-scan design architecture were also used wherever possible as shown in figure 5.13. The chip I/O, however, was the bottleneck in testing this chip because the technologies available for this research limit the I/O data rate to 50 MHz due to physical constraints imposed by the I/O pads in the ring as well as the pins and packaging technology. To be able to test the design running at an internal clock of 100 MHz, a test structure were built in the chip to generate a pseudorandom input traffic to stimulate the sorting queue. Another test structure was used to compare the actual outputs from the sorting queue with those expected due to the pseudorandom traffic and detect any

errors or discrepancies. Monitoring and test points at reduced data rates were also provided through I/O pins whenever possible.

5.5 Summary

This chapter has discussed an architecture for buffer management controllers based on a circular sorting queue that provides combined space and delay priority control. It can also support multicasting traffic. The priority sorting queue was analyzed using a simplified bursty traffic model to study different multicasting schemes. It was also analyzed using an M/G/1/K model and a supplementary variable method for several priority classes. The analysis was used to numerically estimate the buffer size required for the switch and to evaluate its performance. A prototype for the buffer management controller was also implemented in a state-of-the-art CMOS technology. This prototype exhibits a small die area, low-power dissipation and supports high traffic rates at a reasonable clock frequency.

Chapter 6

Conclusion and Future Work

Global satellite networks promise to realize the dream of having a ubiquitous communication connection available for everyone. However, for the global satellite networks themselves to become a reality, so many technical problems and issues have to be resolved. A set of strict physical constraints is imposed by the operating conditions in the satellite environment impose on the electronic processing systems on-board. For our purposes, these constraints affect the choice of both the switching fabric architecture as well as the different design parameters. These physical constraints include limited power dissipation, limited payload size and weight, efficient thermal dissipation and radiation and mechanical vibration resistance measures including but not limited to cooling techniques, packaging technologies, fault-tolerance and error control constraints. Simultaneously, satellite networks need to have very stringent performance constraints. In this work, several issues related to migrating the terrestrial packet switching fabrics on-board satellites have been discussed with emphasis on low power design.

The physical and performance constraints imposed by this new environment

and new application were used identify the most appropriate architecture. A shared memory switching fabric architecture was selected because it lends itself to a smaller highly centralized implementation than other architectures and because it allows for easier implementation of multicasting and prioritization functionalities. Shared memory architectures can be analyzed as output buffers that guarantee full throughput. The memory sharing effect also reduces the buffer size needed to achieve a certain cell loss probability considerably. This was demonstrated using both simulations and analysis for the proposed architecture. One major issue with shared memory architectures is the high memory bandwidth required for such architectures, this problem was alleviated using a simple yet an efficient solution that constructs the logical shared memory pool from several physical memory blocks accessible from all input and output ports using crossbars which reduces the bandwidth requirements on each block by itself. The memory bandwidth requirements are further reduced using a bit-sliced implementation and by exploiting the advances in memory technology. On the other hand, scalability is not much of an issue since satellites have a limited number of beams, hence, a limited number of ports and it is not feasible to retrieve them and expand their payload. In spite of that, a Clos network have been proposed to expand this architecture if necessary. Clos networks along with bit-slicing and other commutation techniques provide a high degree of fault tolerance which was also discussed and used in the proposed architecture.

Next, a novel quantitative framework to the design of the fabric at the system-level is introduced. System-level design of packet switching fabrics has always

focused on performance metrics and rarely considered the physical requirements which are usually addressed later in the design flow at the circuit-level. Moreover, recent advances in VLSI technologies in terms of scaling clock frequencies and higher memory bandwidth have allowed for the design of fabrics with very high aggregate traffic rates. Very dense memories and logic circuits are also being packed to provide for as many connections as possible and to achieve very low cell loss probabilities and better QoS. However, such designs often produce very hot chips and low-power techniques at the circuit-level are not sufficient anymore. This is a very crucial issue in satellite and mobile applications in general. The constraints imposed by the satellite environment add another dimension to the problem.

The proposed framework addresses the problem early in the design process by integrating performance specifications along with physical requirements and constraints to provide for quick and efficient exploration of the design space without compromising the design quality. The system-level design problem is treated as an integer non-linear optimization problem with the dynamic power expression, the dominant component, used as the objective function to be minimized subject to the constraints defined by the traffic-performance metrics, silicon area model and other physical constraints. This framework uses a teletraffic model to integrate the statistics of the network traffic into the power dissipation model as well as in the buffer sizing problem. Different designs for different applications can use the same framework using a different principal criterion as the objective function. The optimization problem was linearized using Fortet's method which linearizes the problem by introducing new variables while maintaining its polynomial size. Then,

an optimal solution was found using an implicit enumeration algorithm based on Davis-Putnam enumeration algorithm. Although, the accuracy of the solution is affected by the linearization process, it is still accurate enough for system-level design. This algorithm is fast enough to explore many regions of the design space and to make informed system-level decisions. Advances in the area of integer non-linear optimization may also increase the efficiency of this framework by providing more accurate solutions for the optimization problem.

The design framework is very flexible and various constraints can be added. However, the real power of this framework lies in the incorporation of the D-BMAP traffic model with the power dissipation model and in the tailoring of the model to real traffic statistics. Another significant advantage of this framework is its adaptability to many types of applications through the selection of the objective function and the bounding constraints. The power dissipation obtained for several experiments were considerably low compared to the estimates for similar fabrics reported for other methods that do not include traffic models and statistics. It should also be noted that optimizing the components of the fabric at lower levels of abstraction may further reduce the power dissipation.

Finally, we have presented a buffer management unit architecture that is based on circular priority queues. This is one of the major components of the switch which implements the scheduling algorithm and provides the QoS guarantees. The architecture was analyzed using an M/G/1/K model for multi-class loss and delay priority traffic. It was also analyzed for multicast traffic support. The results of the analysis can be used to make different choices depending on the QoS required and

the traffic load and environment expected. A prototype for this architecture was also built in a state-of-the-art $0.35\mu\text{m}$ CMOS technology and a deep sub-micron design flow. Several gate-level low-power techniques were used, mainly clock gating and switching activity reduction, to reduce the power by 38%. It can support an aggregate traffic rate of 21.1 Gbps which is sufficient to support about 40 ports, each operating at 500 Mbps for satellite applications.

There are several areas of extension to the work presented here. A major area is the optimization framework. The main areas to consider are mostly the numerical integration of the traffic model into the optimization problem and the numerical solution of the optimization problem. Although improvements in those two areas will increase the accuracy of the results, the main concern is decreasing the run-time needed to explore a large design space. The optimization framework also warrants further work, primarily on extending its application to design architectures for other payload systems such as coding and encryption architectures on-board a satellite. Further, another goal would be to explore the applicability of the shared memory architecture and the priority queues in an environment with a large number of connection or sessions.

In conclusion, this research has advanced the state of the art in the area of switching fabric design for low-power satellite applications through the following contributions:

- The adoption of shared memory fabric architectures for low-power satellite applications [131,132].
- Alleviating the bandwidth limitations using architectural modifications such

as the use of physical blocks with crossbars and the bit-slicing technique [131,132].

- Proposing a novel system-level optimization framework that incorporates both performance and physical constraints and achieves lower power dissipation levels compared to other approaches [133,135].
- Introducing a circular sorting queue architecture for buffer management that can easily implement a wide range of scheduling algorithms and supports both loss and delay priorities as well as multicasting [131,132,134].

Bibliography

- [1] N. Akar, N.C. Oguz, and K. Shoraby. Matrix-geometric solutions of M/G/1-type Markov chains: A unifying generalized state-space approach. *IEEE Journal on Selected Areas in Communications*, 16(5):626–639, June 1998.
- [2] I.F. Akyildiz and S.-H. Jeong. Satellite ATM networks: A survey. *IEEE Communications Magazine*, 35(7):30–43, July 1997.
- [3] D. Anick, D. Mitra, and M. Sondhi. Stochastic theory of a data-handling system with multiple sources. *Bell Systems Technical Journal*, 61(8):1871–1894, 1982.
- [4] R.Y. Awdeh and H.T. Mouftah. MS4—A high performance output buffering ATM switch. *Computer Communication*, 18(9):631–644, Sept. 1995.
- [5] R.Y. Awdeh and H.T. Mouftah. Survey of ATM switch architectures. *Computer Networks and ISDN Systems*, 27(12):1567–1613, Nov. 1995.
- [6] A. Baiocchi, M. Listanti, N. Bléfari-Melazzi, and C. Soprano. An ATM-like system architecture for satellite communications including on-board switching. *International Journal of Satellite Communications*, 14:389–412, 1996.
- [7] H.B. Bakoglu. *Circuits, Interconnects, and Packaging for VLSI*. Addison Wesley, Reading, MA, 1990.
- [8] S. Barbagallo, D. Medina, F. Corno, P. Prineto, and M.S. Reorda. Integrating online and offline testing of a switching memory. *IEEE Design and Test of Computers Magazine*, 15(1):63–70, Jan.-March 1998.
- [9] P. Barth. A Davis-Putnam based enumeration algorithm for linear pseudo-boolean optimization. Research report MPI-I-95-2-003, Max-Planck-Institut für Informatik, Germany, 1995.
- [10] P. Barth and A. Bockmayr. Modelling discrete optimisation problems in constraint logic programming. *Annals of Operations Research*, 81:467–496, 1998.

- [11] A. Bellaouar and M.I. Elmasry. *Low-Power Digital VLSI Design, Circuits and Systems*. Kluwer Academic Publishers, 1995.
- [12] J.M. Benedetto. Economy-class ion-defying ICs in orbit. *IEEE Spectrum*, 35(3):36–41, March 1998.
- [13] L. Benini, A. Bogliolo, M. Favalli, and G. De Micheli. Regression models for behavioral power estimation. In *Proceedings of the International Workshop on Power and timing Modeling, Optimization and Simulation*, pages 179–186, Bologna, Italy, Sept. 1996.
- [14] L. Benini and G. De Micheli. *Dynamic Power Management: Design Techniques and CAD Tools*. Kluwer Academic Publishers, 1998.
- [15] J.C.R. Bennett and H. Zhang. WF²Q: Worst-case fair weighted queuing. *Proc. IEEE INFOCOM*, pages 120–128, 1996.
- [16] E.G. Bernard. Efficient fault location for globally controlled and comparison-based multistage interconnection networks. *IEEE Transactions on Computers*, 45(12):1420–1425, Dec. 1996.
- [17] C. Blondia. A discrete-time batch Markovian arrival process as B-ISDN traffic model. *Belgian Journal of Operations Research, Statistics and Computer Science*, 32(3,4):3–23, 1993.
- [18] D.M. Blough and A. Pelc. A clustered failure model for the memory array reconfiguration problem. *IEEE Transactions on Computers*, 42(5):518–528, May 1993.
- [19] A. Bogliolo and L. Benini. Robust RTL power macromodels. *IEEE Transactions on Very Large Scale Integration Systems*, pages 578–581, Dec. 1998.
- [20] A. Bogliolo, L. Benini, and G. De Micheli. Adaptive least mean square behavioral macro modeling. In *Proc. IEEE European Design and Test Conference*, pages 404–410, March 1997.
- [21] S.E. Butner and R. Chivukula. On the limits of electronic ATM switching. *IEEE Network*, 10(6):26–31, Nov.-Dec. 1996.
- [22] K.S. Chan, S. Chan, and K.L. Yeung. Design of wide-sense nonblocking multicast ATM switches. *IEEE Communications Letters*, 2(5):146–148, May 1998.
- [23] A. Chandrakasan and R. Brodersen. *Low Power Digital CMOS Design*. Kluwer Academic Publishers, 1995.

- [24] A.P. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey, and R.W. Brodersen. Optimizing power using transformations. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 14(1):12–31, Jan. 1995.
- [25] H.J. Chao. A novel architecture for queue management in the ATM network. *IEEE Journal on Selected Areas in Communications*, 9(7):1110–1118, Sept. 1991.
- [26] H.J. Chao, B.-S. Choe, J.-S. Park, and N. Uzun. Design and implementation of Abacus switch: A scalable multicast ATM switch. *IEEE Journal on Selected Areas in Communications*, 15(5):830–843, June 1997.
- [27] C.P. Charalambous. Performance evaluation of TCP extensions on ATM over high bandwidth delay product networks. *IEEE Communications Magazine*, 37(7):57–63, July 1999.
- [28] T.M. Chen and S.S. Liu. *ATM switching systems*. Artech House Publishers, Norwood, MA, 1995.
- [29] T.-H. Cheng. A multichannel ATM switch with output buffering. *Computer Communication*, 29(1):195–208, Jan. 1997.
- [30] P. Chitre and F. Yegenoglu. Next-generation satellite networks: Architectures and implementations. *IEEE Communications Magazine*, 37(3):30–36, March 1999.
- [31] F.M. Chiussi, J.G. Kneuer, and V.P. Kumar. Low-cost scalable switching solutions for broadband networking: The ATLANTA architecture and chipset. *IEEE Communications Magazine*, 35(12):44–53, Dec. 1997.
- [32] S. Civanlar and I. Widjaja, editors. *Broadband Networking Technologies*, volume 3233 of *Proceedings of SPIE*, Dallas, TX, Nov. 1997.
- [33] J. Cong, L. He, and C.-K. Koh. Layout optimization. In W. Nebel and J. Mermet, editors, *Low Power Design in Deep Submicron Electronics*. Kluwer Academic Publishers, 1997.
- [34] P. Coppo, M. D'Ambrosio, and R. Melen. Optimal cost/performance design of ATM switches. *IEEE/ACM Transactions on Networking*, pages 566–575, October 1993.
- [35] Actel Corporation. Design techniques for radiation-hardened field programmable gate arrays. Application Notes, Sept. 1999. <http://www.actel.com/hirel/>.

- [36] D.R. Cox. The analysis of non-markovian stochastic processes by the inclusion of supplementary variables. *Proceedings of the Cambridge Philosophical Society*, 51(3):433–441, 1955.
- [37] R. Crisp. Direct RAMBUS technology: The new main memory standard. *IEEE Micro*, 17(6):18–28, Nov.-Dec. 1997.
- [38] E.G. Cuevas. The development of performance and availability standards for satellite ATM networks. *IEEE Communications Magazine*, 37(7):74–79, July 1999.
- [39] V. Cuppu, B. Jacob, B. Davis, and T. Mudge. A performance comparison of contemporary DRAM architectures. In *Proceedings of the 26th International Symposium on Computer Architectures*, Atlanta, GA, May 1999.
- [40] A.E. Eckberg and T.-C. Hou. Effects of output buffer sharing on buffer requirements in an ATDM packet switch. In *Proc. IEEE INFOCOM*, pages 459–466, New Orleans, LA, March 1988.
- [41] M.S. Elrabaa, I.S. Abu-Khater, and M.I. Elmasry. *Advanced Low-Power Digital Circuits Techniques*. Kluwer Academic Publishers, 1997.
- [42] K.Y. Eng, M.J. Karol, G.J. Cyr, and M.A. Pashan. A 160-Gb/s ATM switch prototype using the concentrator-based growable switch architecture. In *Proc. IEEE International Conference on Communications*, volume 1, pages 550–554, Seattle, WA, June 1995.
- [43] V.J. Friesen and J.W. Wong. The effect of multiplexing, switching and other factors on the performance of broadband networks. *IEEE Network*, 7(1):12–26, Jan. 1993.
- [44] K. Genda and N. Yamanaka. TORUS: Terabit-per-second ATM switching system architecture based on distributed internal speed-up ATM switch. *IEEE Journal on Selected Areas in Communications*, 15(5):817–829, June 1997.
- [45] L. Georgiadis, R. Guerin, V. Peris, and R. Rajan. Efficient support of delay and rate guarantees in an internet. *Proceedings of ACM SIGCOMM '96*, pages 106–116, Oct. 1996.
- [46] J. Gilderson and J. Cherkaoui. Onboard switching for ATM via satellite. *IEEE Communications Magazine*, 35(7):30–43, July 1997.
- [47] S.J. Golestani. A self-clocked fair queueing scheme for broadband applications. In *Proc. IEEE INFOCOM*, pages 636–646, 1993.

- [48] R. Goyal, R. Jain, M. Goyal, S. Fahmy, B. Vandalore, and S. Kota. Traffic management for TCP/IP over satellite ATM networks. *IEEE Communications Magazine*, 37(3):56–61, March 1999.
- [49] A. Gravey and G. Hebuterne. Mixing time and loss priorities in a single server queue. In *Proceeding of ITC 13*, pages 147–152, 1991.
- [50] S.J. Gu. Nonblocking conditions for self-routing Beneš network. In *Proceedings of the IEEE Region 10 Conference on Computer, Communication, Control and Power Engineering, TENCN'93*, pages 203–206, 1993.
- [51] S. Gupta and F. Najm. Power macromodeling for high-level power estimation. In *Proc. ACM/IEEE Design Automation Conference*, pages 365–370, Anaheim, CA, June 1997.
- [52] M.R. Hashemi and A. Leon-Garcia. The single-queue switch: A building block for switches with programmable scheduling. *IEEE Journal on Selected Areas in Communications*, 15(5):785–794, June 1997.
- [53] M.G. Hluchyj and M.J. Karol. Queueing in high-performance packet switching. *IEEE Journal on Selected Areas in Communications*, 6(9):1587–1597, Dec. 1988.
- [54] J.D. Ho and N.K. Sharma. A growable shared-memory based multicast ATM switch. In *Proc. IEEE GLOBECOM*, pages 2363–2368, 1998.
- [55] C.-T. Hsieh, C.-S. Deng, Q. Wu, and M. Pedram. Statistical sampling and regression estimation in power macro-modeling. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 583–588, San Jose, CA, Nov. 1996.
- [56] C.X. Huang, B. Zhang, C.D. An, and B. Swirski. The design and implementation of POWERMILL. In *Proceedings of the ACM/IEEE International Symposium on Low-Power Design*, pages 105–110, April 1995.
- [57] J.Y. Hui. *Switching and Traffic Theory for Integrated Broadband Networks*. Kluwer Academic Publishers, Boston, MA, 1990.
- [58] A. Hung, M.-J. Montpetit, and G. Kesidis. ATM via satellite: A framework and implementation. *Wireless Networks*, 4(2):141–154, Feb. 1998.
- [59] K. Itoh, K. Sasaki, and Y. Nakagome. Trends in low-power RAM circuit technologies. *Proceedings of the IEEE*, 83(4):524–543, April 1995.
- [60] W.D. Ivancic, D. Brooks, B. Frantz, D. Houder, D. Shell, and D. Beering. NASA's broadband satellite networking research. *IEEE Communications Magazine*, 37(7):40–47, July 1999.

- [61] J.-M. Li, I. Widjaja, and M.F. Neuts. Congestion detection in ATM networks. *Performance Evaluation*, 34(3):147–168, 1998.
- [62] B.W. Johnson. *Design and Analysis of Fault Tolerant Digital Systems*. Addison Wesley, Reading, MA, 1989.
- [63] M.B. Josephs, S.M. Nowick, and C.H. van Berkel. Modeling and design of asynchronous circuits. *Proceedings of the IEEE*, 87:234–242, Feb. 1999.
- [64] Y.C. Jung and C.K. Un. Analysis of backpressuring-type packet switches with input and output buffering. *IEE Proceedings-I*, 140(4):277–284, Aug. 1993.
- [65] M.J. Karol, M.G. Hluchyj, and S.P. Morgan. Input versus output queueing on a space-division packet switch. *IEEE Transactions on Communications*, 35(12):1347–1356, Dec. 1987.
- [66] Y. Katayama. Trends in semiconductor memories. *IEEE Micro*, 17(6):10–17, Nov.-Dec. 1997.
- [67] M.G.H. Katevenis, P. Vatsolaki, D. Serpanos, and E. Markatos. ATLAS I: A single-chip ATM switch NOWs. In *Proceedings CANPS'97*, pages 88–101, San Antonio, TX, Feb. 1997.
- [68] M. Kazemi-Nia and H.M. Alnuweiri. VLSI implementation of highly-optimized scalable ATM switches. In *Proc. 18th Biennial Symposium on Communications*, pages 101–104, Kingston, ON, June 1996.
- [69] J. Kessels and P. Marston. Designing asynchronous standby circuits for a low-power pager. *Proceedings of the IEEE*, 87:257–267, Feb. 1999.
- [70] L. Kleinrock. *Queueing Systems: Theory*, volume 1. John Wiley & Sons, Inc., 1975.
- [71] N. Kumar, S. Katkoori, L. Rader, and R. Vemuri. Profile-driven behavioral synthesis for low power VLSI systems. *IEEE Design and Test of Computers Magazine*, 12(3):70–84, March 1995.
- [72] S. Kumar and D.P. Agrawal. A shared-buffer direct-access (SBDA) switch architecture for ATM-based networks. In *Proc. IEEE International Conference on Communications*, pages 101–105, New Orleans, LA, May 1994.
- [73] S. Kumar and D.P. Agrawal. On multicast support for shared-memory-based ATM switch architecture. *IEEE Network*, 10(1):34–39, Jan.-Feb. 1996.

- [74] H. Kuwahara, N. Endo, M. Ogino, and T. Kozaki. A shared buffer memory switch for an ATM exchange. In *Proc. IEEE International Conference on Communications*, pages 118–122, Boston, MA, June 1989.
- [75] L.E. LaForge. Configuration of locally spared arrays in the presence of multiple fault types. *IEEE Transactions on Computers*, 48(4):398–416, april 1999.
- [76] P.E. Landman and J. Rabaey. Activity-sensitive architectural power analysis for the control path. In *Proceedings of the ACM/IEEE International Symposium on Low-Power Design*, pages 93–98, Dana Point, CA, April 1995.
- [77] P.E. Landman and J.M. Rabey. Architectural power analysis: The dual bit type method. *IEEE Transactions on Very Large Scale Integration Systems*, 3(2):173–187, June 1995.
- [78] P.E. Landman and J.M. Rabey. Activity-sensitive architectural power analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(6):571–578, June 1996.
- [79] K.L.E. Law and A. Leon-Garcia. A large scalable ATM multicast switch. *IEEE Journal on Selected Areas in Communications*, 15(5):844–854, June 1997.
- [80] T. Lazaraq, P.-O. Bergstedt, M. Mokhtari, and H. Tenhunen. ATM switching element design for 10 Gbit/sec/port data rate. In *Proc. IEEE International Conference on Communications*, pages 669–674, Dallas, TX, June 1996.
- [81] H. Lee, K.H. Kook, C.S. Rim, K.P. Jun, and S.K. Lim. A limited shared output buffer switch for ATM. In *4th Int. Conf. on Data Communications Systems and Their Performance*, pages 163–179, Barcelona, Spain, June 1990.
- [82] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. On the self-similar nature of Ethernet-traffic. *IEEE/ACM Transactions on Networking*, 2:1–15, 1994.
- [83] S.-Q. Li, S. Park, and D. Arifler. SMAQ: A measurement-based tool for traffic modeling and queuing analysis: II. Network applications. *IEEE Communications Magazine*, 36(8):66–70, Aug. 1998.
- [84] S.-Q. Li and H.-D. sheng. Generalized folding algorithm for transient analysis of finite QBD processes and its queueing applications. In W.J. Stewart, editor, *Computations with Markov Chains*, pages 463–481. Kluwer Academic Publishers, Dordrecht, 1995.

- [85] D. Lidsky and J. Rabaey. Early power exploration – A World Wide Web application. In *Proc. ACM/IEEE Design Automation Conference*, pages 27–32, 1996.
- [86] D. Liu and C. Svensson. Power consumption estimation in CMOS VLSI chips. *IEEE Journal of Solid State Circuits*, 29(6):663–670, June 1994.
- [87] J.W. Lockwood, H. Duan, J.J. Morikuni, S.-M. Kang, S. Akkineni, and R.H. Campbell. Scalable optoelectronic ATM networks: The iPOINT fully functional testbed. *IEEE Journal of Lightwave Technology*, 13(6):1093–1103, June 1995.
- [88] S.-K. Lu, S.-Y. Kuo, and C.-W. Wu. Fault-tolerant interleaved memory systems with two-level redundancy. *IEEE Transactions on Computers*, 46(9):1028–1034, Sept. 1997.
- [89] D.M. Lucantoni. The BMAP/G/1 queue: A tutorial. In L. Donatiello and R. Nelson, editors, *Performance '93 and Sigmetrics '93, Lecture Notes in Computer Science - 729*, pages 330–358. Springer Verlag, 1993.
- [90] E. Macii, M. Pedram, and F. Somenzi. High-level power modeling, estimation, and optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(11):1061–1079, Nov. 1998.
- [91] D.J. Marchok and C.E. Rohrs. First stage multicasting in a growable packet (ATM) switch. In *Proceedings of the IEEE International Conference on Communications*, pages 1007–1013, June 1991.
- [92] D. Marculescu, R. Marculescu, and M. Pedram. Information theoretic measures for power analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(6):599–610, June 1996.
- [93] T.L. Martin and D.P. Siewiorek. A power metric for mobile systems. In *Proceedings of the ACM/IEEE International Symposium on Low-Power Electronics and Design*, pages 37–42, Aug. 1996.
- [94] H. Mehta, R. Owens, and M.J. Irwin. Energy characterization based on clustering. In *Proc. ACM/IEEE Design Automation Conference*, pages 702–707, Las Vegas, NV, June 1996.
- [95] I. Mertzanis, G. Sfikas, R. Tafazolli, and B.G. Evans. Protocol architectures for satellite ATM broadband networks. *IEEE Communications Magazine*, 37(3):46–54, March 1999.

- [96] A. Metzger, C.E. Chang, P.M. Asbeck, K.C. Wang, K. Pedrotti, A. Price, A. Campana, D. Wu, J. Liu, and S. Beccue. A 10 Gb/s 12×12 cross-point switch implemented with AlGaAs/GaAs heterojunction bipolar transistors. In *Proc. IEEE GaAs IC Symposium*, pages 109–112, Anaheim, CA, Oct. 1997.
- [97] H. Michiel and K. Laevens. Teletraffic engineering in a broad-band era. *Proceedings of the IEEE*, 85:2007–2033, Dec. 1997.
- [98] K. Muller-Glaser, K. Kirsch, and K. Neusinger. Estimating essential design characteristics to support project planning for ASIC design management. In *Proceedings of the IEEE International Conference on Computer-Aided Design*, pages 148–151, Los Alamitos, CA, Nov. 1991.
- [99] M. Nemani and F. Najm. Toward a high-level power estimation capability. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(6):588–598, June 1996.
- [100] M. Nemani and F. Najm. High-level area prediction for power estimation. In *Proceedings of the Custom Integrated Circuits Conference*, pages 483–486, Santa Clara, CA, May 1997.
- [101] M.F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. The John Hopkins University Press, Baltimore, MD, 1981.
- [102] M.F. Neuts. *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Marcel Dekker, Inc., New York, NY, 1989.
- [103] L.S. Nielsen and J. Sparsø. Designing asynchronous circuits for low power: An IFIR filter bank for a digital hearing aid. *Proceedings of the IEEE*, 87:268–281, Feb. 1999.
- [104] A. Nunez, R. Sarmiento, R. Esper-Chain, J. Jakobsen, J.A. Montiel-Nelson, J. Lopez, V. Armas, and F. Tobajas. GaAs ICs for 10 Gb/s ATM switching. In *Proc. IEEE GaAs IC Symposium*, pages 101–104, Anaheim, CA, Oct. 1997.
- [105] H. Ohsaki, N. Wakamiya, M. Murata, and H. Miyahara. Performance of an input/output buffered-type ATM LAN switch with back-pressure function. *IEEE/ACM Transactions on Networking*, 5(2):278–290, April 1997.
- [106] S.F. Oktuğ and M.U. Çağlayan. Design and performance evaluation of a banyan network based interconnection structure for ATM switches. *IEEE Journal on Selected Areas in Communications*, 15(5):807–816, June 1997.
- [107] K. Padmanabhan. An efficient architecture for fault-tolerant ATM switches. *IEEE/ACM Transactions on Networking*, 3(5):527–537, October 1995.

- [108] A.K. Parekh and R.G. Gallager. A generalized processor sharing approach to flow control -- The single node case. *Proc. IEEE INFOCOM*, 2:915–924, May 1992.
- [109] B. Patel, F. Schaffa, and M. Willebeek-LeMair. The Helix switch: A single chip cell switch design. *Computer Communication*, 28:1791–1807, 1996.
- [110] A. Pattavina. Nonblocking architectures for ATM switching. *IEEE Communications Magazine*, 31(2):38–48, Feb. 1993.
- [111] A. Pattavina and G. Bruzzi. Analysis of input and output queuing for non-blocking ATM switches. *IEEE/ACM Transactions on Networking*, 1(3):314–328, June 1993.
- [112] S.R. Powell and P.M. Chau. Estimating power dissipation of VLSI signal processing chips: The PFA technique. In *VLSI Signal Processing IV*, pages 250–259, 1990.
- [113] Q. Qiu, Q. Wu, M. Pedram, and C.-S. Deng. Cycle-accurate macro-models for RT-level power analysis. In *Proceedings of the ACM/IEEE International Symposium on Low-Power Electronics and Design*, pages 125–130, Monterey, CA, Aug. 1997.
- [114] V. Ramaswami. A stable recursion for the steady state vector in Markov chains of M/G/1 type. *Communications Statistics - Stochastic Models*, 4(1):183–188, 1988.
- [115] E.P. Rathgeb, W. Fischer, C. Hinterberger, E. Wallmeier, and R. Wille-Fier. The MainStreetXpress core services node—A versatile ATM switch architecture for the full service network. *IEEE Journal on Selected Areas in Communications*, 15(5):795–806, June 1997.
- [116] J. Roberts, U. Mocci, and J. Virtamop, editors. *Broadband Network Teletraffic*, volume 1155 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996. Final Report of Action COST 242.
- [117] J.W. Roberts, P.E. Boyer, and M.J. Serval. A real time sorter with application to ATM traffic control. In *Proc. International Switching Symposium*, pages 258–262, Berlin, Germany, April 1995.
- [118] R. Savara and A. Turudic. A 2.5 Gb/s 16×16 bit crosspoint switch with fast programming. In *Proc. IEEE GaAs IC Symposium*, pages 47–48, San Diego, CA, Oct. 1995.

- [119] K.J. Schultz and P.G. Gulak. CAM-based single-chip shared buffer ATM switch. In *Proc. IEEE International Conference on Communications*, pages 1190–1195, New Orleans, LA, May 1994.
- [120] K.J. Schultz and P.G. Gulak. Physical performance limits for shared buffer ATM switches. *IEEE Transactions on Communications*, 45(8):997–1007, August 1997.
- [121] S. Shaikh, M. Schwartz, and T. Szymanski. A comparison of the Shuffelnet and the Banyan topologies for broadband packet switches. In *Proc. IEEE INFOCOM*, pages 1260–1267, 1990.
- [122] H. Shi, C. Zukowski, and O. Wing. VLSI design optimization of input/output-buffered broadband ATM switches. In *Proc. IEEE INFOCOM*, pages 810–817, 1996.
- [123] J.A. Silvester, N.L.S. Fonseca, and S.S. Wang. D-BMAP models for performance evaluation of ATM networks. In *IFIP Workshop on Performance Modelling and Evaluation of ATM Networks*, pages 325–346, 1994.
- [124] J. Solé-Pareta and J. Domingo-Pascual. Burstiness characterization of ATM cell streams. *Computer Networks and ISDN Systems*, 26:1351–1363, 1994.
- [125] D. Stiliadis and A. Varma. Latency-rate servers: A general model for analysis of traffic scheduling algorithms. *Proc. IEEE INFOCOM*, pages 111–119, 1996.
- [126] S. Sumita. Analysis of finite buffer head-of-the-line priority queues with push-out scheme as space priority. *IEICE Transactions on Communications*, E81-B(1):23–31, Jan. 1998.
- [127] S. Sumita and T. Ozawa. Achievability of performance objectives in ATM switching nodes. In *Proceedings of the International Seminar on Performance of Distributed and Parallel Systems*, pages 45–56, Amsterdam, 1988. North-Holland.
- [128] F.A. Tobagi. Fast packet switch architectures for broadband integrated services digital networks. *Proceedings of the IEEE*, 78(1):133–166, Jan. 1990.
- [129] A. Varma and D. Stiliadis. Hardware implementation of fair queuing algorithms for asynchronous transfer mode networks. *IEEE Communications Magazine*, pages 54–68, Dec. 1997.
- [130] K. Wang and C.-K. Wu. Design and implementation of fault-tolerant and cost effective crossbar switches for multiprocessor systems. *IEE Proc.—Computer and Digital Techniques*, 146(1):50–56, Jan. 1999.

- [131] A.G. Wassal and M.A. Hasan. A VLSI switch architecture for broadband satellite networks. In *Proceedings of the IEEE Midwest Symposium on Circuits and Systems*, pages 42–45, Notre Dame, IN, Aug. 1998.
- [132] A.G. Wassal and M.A. Hasan. Prioritized ATM switches on-board satellites: Architectural analysis and design. *Submitted to the IEE Proceedings-Communications*, March 1999.
- [133] A.G. Wassal and M.A. Hasan. Traffic-driven low-power design of VLSI packet switching fabrics. In *Proceedings of the Third Canadian Conference on Broadband*, pages 78–89, Ottawa, ON, Nov. 1999.
- [134] A.G. Wassal and M.A. Hasan. A VLSI architecture for ATM algorithm-agile encryption. In *Proceedings of the IEEE 9th Great Lakes Symposium on VLSI*, pages 325–328, Ann Arbor, MI, March 1999.
- [135] A.G. Wassal and M.A. Hasan. Low-power system-level design of packet switching fabrics. *Submitted to the IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Jan. 2000.
- [136] G.-L. Wu and J.W. Mark. A buffer allocation scheme for ATM networks: Complete sharing based on virtual partition. *IEEE/ACM Transactions on Networking*, 3(6):660–670, Dec. 1995.
- [137] H. Yamada, M. Tsunotani, F. Kaneyama, and S. Seki. 20.8 Gb/s GaAs LSI's self-routing switch for ATM switching systems. *IEEE Journal of Solid State Circuits*, 32(1):31–37, Jan. 1997.
- [138] H. Yamanaka, H. Saito, H. Kondoh, Y. Sasaki, H. Yamada, M. Tsuzuki, S. Nishio, H. Notani, A. Iwabu, M. Ishiwaki, S. Kohama, Y. Matsuda, and K. Oshima. Scalable shared-buffering ATM switch with a versatile searchable queue. *IEEE Journal on Selected Areas in Communications*, 15(5):773–784, June 1997.
- [139] Y. Yang and J. Wang. Wide-sense nonblocking Clos networks under packing strategy. *IEEE Transactions on Computers*, 48(3):265–284, March 1999.
- [140] Y.-S. Yeh, M. Hluchyj, and A. Acampora. The knockout switch: A simple, modular architecture for high performance packet switching. *IEEE Journal on Selected Areas in Communications*, 5(8):1274–1283, Oct. 1987.
- [141] E.W. Zegura. Architectures for ATM switching systems. *IEEE Communications Magazine*, 31(2):28–37, February 1993.
- [142] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83:1374–1396, Oct. 1995.