Isochronets: a High-speed Network Switching Architecture

Danilo Florissi

Technical Report CUCS-021-95

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY
1995

ABSTRACT

Isochronets: a High-speed Network Switching Architecture

Danilo Florissi

Traditional network architectures present two main limitations when applied to High-Speed Networks (HSNs): they do not scale with link speeds and they do not adequately support the Quality of Service (QoS) needs of high-performance applications. This thesis introduces the Isochronets architecture that overcomes both limitations.

Isochronets view frame motions over links in analogy to motions on roads. In the latter, traffic lights can synchronize to create green waves of uninterrupted motion. Isochronets accomplish similar uninterrupted motion by periodically configuring network switches to create end-to-end routes in the network. Frames flow along these routes with no required header processing at intermediate switches.

Isochronets offer several advantages. First, they are scaleable with respect to transmission speeds. Switches merely configure routes on a time scale that is significantly longer than and independent of the average frame transmission time. Isochronets do not require frame processing and thus avoid conversions from optical to electronic representations. They admit efficient optical transmissions under electronically controlled switches.

Second, Isochronets ensure QoS for high-performance applications in terms of latency, jitter, loss, and other service qualities. Isochronet switches can give priority to frames ar-

riving from selected links. At one extreme, they may give a source the right-of-way to the destination by assigning priority to all links in its path. Additionally, other sources may still transmit at lower priority. At the other extreme, they may give no priority to sources and frames en route to the same destination contend for intermediate links. In between, Isochronets can accomplish a myriad of priority allocations with diverse QoS.

Third, Isochronets can support multiple protocols without adaptation between different frame structures. End nodes view the network as a media access layer that accepts frames of arbitrary structure.

The main contributions of this thesis are:

- Design of the Isochronets architecture.
- Design and implementation of a gigabit per second Isochronet switch (Isoswitch).
- Definition of the Loosely-synchronous Transfer Mode (LTM) and the Synchronous Protocol Stack (SPS) that add synchronous and isochronous services to any existing protocol stack.
- Performance evaluation of Isochronets.

Table of Contents

List of Figures

List of Acronyms

AAL	ATM Adaptation Layer
AL	Application Layer
AL	Arbitration Logic
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband Integrated Service Digital Networks
BS	Burst Switching
CM	Configuration Memory
CS	Circuit Switching
CT	Configuration Table
FPS	Fast Packet Switching
HoL	Head of Line
HSN	High-speed Network
HSNSA	HSN Switching Architecture
IP	Internet Protocol
Isoswitch	Isochronet switch
LLN	Linear Lightwave Networks
LCA	Logic Cell Array
LRU	Least Recently Used
LTM	Loosely-synchronous Transfer Mode

LTM-MAC	LTM MAC
MAC	Medium Access Control
NTP	Network Time Protocol
PCR	Program Counter Read
PCW	Program Counter Write
PL	Physical Layer
PS	Packet Switching
QoS	Quality of Service
RAM	Random Access Memory
RDMA	Route Division Multiple Access
RTP	Real Time Protocol
SAP	Service Access Point
SPS	Synchronous Protocol Stack
ST-II	Internet Stream Protocol version II
STM	Synchronous Transfer Mode
TDM	Time Division Multiplexing
TDMA	Time Division Multiple Access
TL	Transport Layer
VC	Virtual Circuit
VCI	VC Identifier
VP	Virtual Path
VPI	VP Identifier

WDM	Wavelength Division Multiplexing

Para minha esposa, com amor.

To my wife, with love.

Acknowledgments

I consider myself very fortunate to having been given the opportunity of working with my advisor, Yechiam Yemini (YY). This small paragraph cannot make justice to his incredible guidance and friendship. As an advisor, he inspired and challenged me with his brilliant knowledge, wisdom, and intuition. This work would not have happened without his creative ideas and extraordinary assistance. As a friend, he has helped me through countless difficulties. I am forever in debt to him. I will strive to account for the time he has spent educating me.

I am in debt to my former advisor and friend, Calton Pu. I will never forget his advice and supervision.

I am most thankful to my thesis committee, Daniel J. Duchamp, Steven Nowick, Thomas Stern, and Charles A. Zukowski, for their outstanding comments and suggestions on this work. I feel very fortunate for having had this work evaluated by such group of outstandingly bright and dedicated researchers.

I thank Ron Erlich for his help with the implementation of the switch. I thank Rahul Sood and Gong Su for their help in building a demo video application using the switch.

I thank the Brazilian Research Council (CNPq) grant 204544/89.0 for their generous support during the first four years in the program.

I thank many current members of, former members of, and visitors to the Distributed Computing and Communications (DCC) Lab for uncountable valuable discussions, helpful insights, and great friendly times. These include Timothy Balraj, Apostolos Dailianas, Alexander Dupuy, Ashutosh Dutta, Hong Liu, and Sushil da Silva. I would also like to acknowledge the other DCC members for their great friendship.

I thank our chairman Zvi Galil. The department has been very fortunate for having had his guidance and perceptiveness.

I thank Susan Tritto for her incredible friendship and great help in making me understand complicated bureaucratic procedures at Columbia. I thank Germaine L'Eveque and Mel Francis for their friendship, their worm reception when we arrived in the department, and their support throughout these many years at Columbia.

Many Faculty members helped me with friendship and intelligent discussions: Shree Nayar, Sal Stolfo, and Thanasis Tsantilas.

I was fortunate to find many friends in the department and I thank them too.

I am forever in debt to my friend, office mate, program mate, discussion mate, and, most importantly, wife, Patrícia Gomes Soares Florissi. This thesis and many other steps I had to go through during my Columbia years would simply not have happened without her incredible support, help, and patience through most difficult and worrisome times. I cannot thank her enough, but I will start by dedicating this work to her.

I am most fortunate to have had the love, support, and endurance of my family during these many years apart. I am most thankful to my mother Adriana, my father Sergio, my brother Aldo, and my sister Susanna. They must take the credit for making me who I am today. I am also most thankful for my mother-in-law Hercília and my father-in-law Waldecy for their outstanding help and understanding.

I thank God for His help through this challenging task. I could not have climbed this difficult step without His upholding thrust.

Chapter 1

Introduction

High-Speed Networks (HSNs) present two fundamental challenges for candidate network architectures. First, the network architecture should be scaleable with transmission speeds. Transmissions in the optical domain can use efficient coding techniques and very high bandwidth rates not available in current electronic technology. If the architecture relies on electronic frame processing, it will introduce two main limitations on the maximum end-to-end throughput. The first limitation is intrinsic in the fact that bit processing by electronic gates is currently less efficient than transmissions of optically coded information. If optical signals are converted to electronics for processing, the maximal end-to-end network efficiency will hit the natural electronic bound instead of the desirable optical one. The second limitation is due to the fact that networks must service multiplexed streams generated and consumed by electronic devices at the network periphery. If the multiplexed streams are switched using the very same electronic technology in internal network devices, a small number sources operating at peak rate may congest the network. The ideal architecture should be prepared to operate at traffic volumes much larger than what an individual source may generate.

Second, motions must happen with strict Quality of Service (QoS) guarantees. Applications need guarantees in the end-to-end delay, jitter, bandwidth, etc. to provide their services. The ideal network architecture should strictly assure the negotiated QoS parameters. Many times such assurance imply trading optimal resource use for optimal service. For example, in some circumstances it is preferable to waste bandwidth on communication channels that require strict QoS in order to guarantee minimal delay and jitter.

The goal of this dissertation is to develop, build, and analyze the novel Isochronets switching architecture that solves both challenges. The reminder of this chapter overviews the architecture and summarizes main contributions of this work. It can be skipped by those interested in reading the work in more detail, starting from Chapter 2.

1.1 Isochronets and Route Division Multiple Access (RDMA)

Isochronets are a novel switching architecture that *coordinates* traffic motions in the network over time, similar to how green waves coordinate traffic motions in roads. In the latter scenario, coordination is accomplished through synchronized traffic lights to create routes for uninterrupted and collision-free motions from source to destinations. In Isochronets, coordination is accomplished by enabling a set of routes by means of periodic configuration of network switches. Switches set up routing trees, that is, routes from all nodes to a given destination. A frame is switched by following its path through an enabled tree. The coordination mechanism is called *Route Division Multiple Access (RDMA)* because it divides network bandwidth among routing trees.

In RDMA, the basic construct used to schedule traffic motion is a time-band (*green-band*) assigned to a routing tree, as depicted in Figure 1.1. The green-band is a time interval during which network switches are configured to route according to a particular routing tree. During the green-band (shaded), a frame transmitted by a source will propagate down the routing tree to the destination root. If no other traffic contends for the tree, it will move uninterrupted, as depicted by the straight line. Additionally, intermediate nodes perform no header processing. Multiple simultaneous routing trees can schedule transmissions in parallel (have simultaneous green bands), as long as they do not share any link in the same direction.

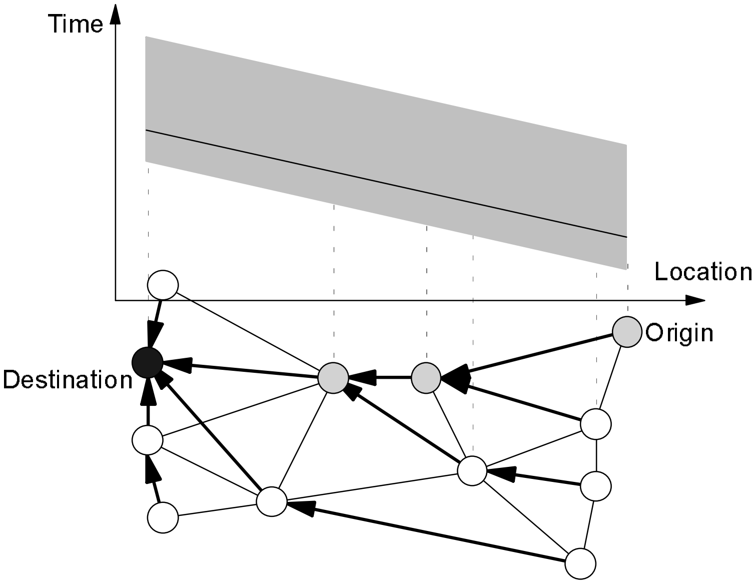


Figure 1.1: Green-band

Similar to Circuit Switching (CS), green-bands allocate reserved network resources. However, the units to which resources are allocated are neither point-to-point connections, nor traffic bursts, but routes. Routes represent long-lived entities and, thus, scheduling complexities can be resolved over time scales much longer than circuit slots. Additionally, routes are pre-allocated and not allocated on-demand like circuits.

Similar to Packet Switching (PS), traffic within a routing tree contends for outgoing links. However, contention is only among frames to the same final destination. Additionally, there is no contention for processing resources.

The allocation of synchronized time bands to routing trees and resolution of frame collisions are the primitive constructs used by RDMA to control traffic motions and QoS. *Contention bands* permit frame contention to happen for outgoing tree links. *Priority bands* are allocated to sources requiring absolute QoS guarantees, similar to a circuit service. Traffic from a priority source is given the right of way by switches on its path during its priority band. Unlike CS networks, however, priority sources do not own their bands. Contention traffic may access a priority band and utilize it whenever the priority source does not. During a *multicast band*, the routing tree is reversed and the root can broadcast to any subset of nodes.

Bands are allocated as portions of a fixed *clock cycle*. Every time the clock turns, the bands become available in sequence according to their position in the cycle.

The collision resolution during a contention band is designed in terms of signs “-”, “+”, and “++”. In RDMA- only one frame proceeds, while the other colliding frames are discarded. In RDMA+, only one frame proceeds, but the others are buffered. Nevertheless, when a band finishes, all buffered frames are discarded. RDMA++ uses the same collision resolution technique as RDMA+, but stores frames beyond band termination, rescheduling them during the next band.

Isochronets present a few distinguishing characteristics:

- The band periodicity may vary with the type of traffic served. For example, traffic such as file transfers may use periods of long duration, whereas interactive voice or video traffic may use much shorter periods.
- All stack layers above the Medium Access Control (MAC) layer are delegated to

interfaces at the network periphery. Multiple protocol frames may coexist in the same network without adaptation.

- Interconnection of Isochronets can be accomplished via MAC layer bridges using extensions of current well-understood technologies.

It is interesting to compare RDMA with respect to PS and CS. If all the bands associated with a routing tree are priority bands, RDMA is operating on an optimized CS mode when the tree is enabled. That is, each source is allocated a circuit (priority band) to the tree root. The form of CS supported by Isochronets is superior to traditional CS in a few ways. First, circuits only get priority over band usage but do not own it. In a situation where the band has been allocated to a source serving real-time isochronous traffic, non-real-time data traffic may take advantage of underutilized parts of the band. Second, CS requires strict synchronization, usually per byte when implemented using Time Division Multiple Access (TDMA) [Halsall 92]. RDMA can offer the same services requiring synchronization per band that is usually much larger than a byte. Third, CS requires fast switching of bytes that can be difficult to accomplish when link speeds are gigabits or even terabits per second. RDMA needs only to switch routes independently of transmission speeds. Finally, CS requires a connection setup phase prior to communication that potentially can take longer than the transmissions in HSNs. RDMA can forward frames when the proper band starts, which depends on the clock cycle and not on the propagation delay in the network.

Consider now an Isochronet operating in RDMA++ contention resolution mode. If the entire band is allocated to contention traffic, frames moving down the tree will be stored and forwarded as in an ordinary PS network. The form of PS supported by Isochronets is superior to traditional packet switching in a few ways. First, Isochronets support virtual cut-through mechanisms [Kermani 78] as frames arriving to a free switch will continue without store-and-forward delay. Second, there is no header processing in Isochronet switches. Third, contention happens only among frames to the same destination (and not among non-correlated traffic). Finally, RDMA can offer QoS to demanding sources.

In summary, at the two extremes of band allocations, Isochronets compare favorably with CS and PS networks. In-between, Isochronets can allocate combinations of priority and contention bands to produce a spectrum of switching techniques.

1.2 Protocols in Isochronets

1.2.1 The Problem

Isochronets need to solve three essential operational problems:

- How to allocate trees in the networks?
- How to allocate bands to trees?
- How to synchronize the bands at different nodes?

This section overviews each of these problems.

Tree allocation. In this problem, the objective is to allocate routing trees in the network so as to satisfy some optimization criteria. The exact criterion is dependent on the particular goals that the network is intended to satisfy. For example, trees can be allocated to minimize the distance between sources and destinations. This is particularly useful when links are operating at high speeds because the propagation delay is a major component in the total end-to-end delay. Another example particular to Isochronets is to allocate trees such that the number of bands necessary to cover all possible destinations is minimized. This goal can be accomplished by allocating as many non-interfering trees as possible in the same band. In this case, the tree allocation mechanism should be geared toward choosing as many trees as possible that do not share links in the same direction.

Band synchronization. Given the band allocations in each node, the objective is to synchronize the bands, that is, assure that frames transmitted from a given node during band B_i reach the next node in the tree within the same band B_i. Synchronization mismatches in Isochronets may result in frames being misrouted. For example, if a frame transmitted during band B_i reaches an intermediate node during another band B_j, frames are forwarded from that point on toward B_j's destination. Another concern is to avoid bands intersecting each other. For example, if the intersection between bands B_i and B_j at some node is not empty, routing during the overlap period is ambiguous because incoming frames could potentially follow two separate paths.

Band allocation. Given a set of band transmission requirements, the goal is to allocate bands that will fit the requirements while being synchronized and not interfering with each other. The band allocation is thus concerned in finding two parameters for each band: initiation time within the clock cycle and size. The allocation changes over time given new demands computed from past history.

1.2.2 Main Results

Tree allocation. The proposed initial solution is an exhaustive search algorithm. It begins by generating all spanning trees in the network. Then it builds all possible combinations of generated trees into maximal sets of non-interfering trees. That is, sets that contain only non-interfering trees such that no tree can be added without violating the non-interference constraint. Trees in the same maximal set can be assigned to the same band. Finally, the protocol finds the minimal number of maximal sets that covers all possible destinations. This number is also the number of bands in the cycle.

This exhaustive search solution is acceptable because the tree allocation protocol is executed only when the network configuration changes. The tree allocation protocol can be implemented off-line and execute at slow speed relative to transmissions.

Band synchronization. This problem is solved by manipulating the link propagation delays in the network. There is a device in each link that can be set to increase the link delay. The protocol computes the real link delay and then increases it using the device to make the final delay a multiple of the clock cycle period. Because of this manipulation, if something is sent at time *t* within the cycle from a given node, it reaches the next node still at time *t* but many cycles later. This fact trivializes synchronization because all that is necessary after link delays are set is to start and end each band exactly at the same moment within each cycle.

This protocol can operate off-line with transmissions. It is invoked periodically, depending on the accuracy desired.

Band allocation. For the purpose of this protocol, bands are always synchronized using the band synchronization protocol. The main concern of the band allocation protocol is then to size the bands. The task is pursued using a dynamic algorithm based on the Least

Recently Used (LRU) memory allocation policy. The protocol operates by shrinking band sizes for destinations that are not accessed frequently and then increasing band sizes of destinations that need higher demands.

This protocol may also operate off-line with transmissions. It may operate in a central location responsible for computing the allocation and then distributing it.

1.3 Protocols for Loosely-Synchronous Stacks

1.3.1 The Problem

Existing protocol stacks must be extended to support real-time traffic required by current applications. New real-time applications, such as multimedia exchanges, demand synchronous or isochronous services from the protocol stack. One important question is how to extend existing stacks in this direction while still providing current functionality. Ideally, the new services must be provided as extensions of current protocols to maintain compatibility with existing systems.

1.3.2 Main Results

This thesis presents two main contributions in the direction of providing real-time services. First, a novel *Loosely-synchronous Transfer Mode (LTM)* provides the functionality needed to support synchronous services to upper layers. Second, any protocol stack can be extended with synchronization signals from the LTM and become a *Synchronous Protocol Stack (SPS)*.

LTM. LTM is a novel transfer mode that issues *loosely-synchronous signals* to periphery nodes about current network status, that is, destinations that are reachable and associated QoS. An LTM network enables transmissions to some destinations during certain periodic time intervals or *bands*. Each band has associated QoS in terms of the bandwidth, end-to-end delay, jitter, and loss experienced in the communication.

LTM has many advantages:

- Sources and destinations are synchronized with the network through loosely-synchronous signals. That is, sources are synchronized with network status and not with frames or slots within frames as in Synchronous Transfer Mode (STM). Consequently, the synchronization does not need to be as accurate as in STM but still can attain the same sort of guaranteed service.
- QoS can be controlled and guaranteed because the loosely-synchronous signals coordinate how sources and the network must interact. Guaranteed performance is essential for time-sensitive applications such as multimedia conference.
- Multiple protocol frames can be transferred without fragmentation and reassembly because there is no pre-defined frame structure in the communication. This makes the extension of existing protocol stacks with the loosely-synchronous signals simple. Additionally, interconnection of multiple stacks is simplified.

RDMA in Isochronets is an example of an LTM.

SPS. The SPS is any stack extended with the loosely synchronized signals provided by the LTM. An SPS forwards LTM synchronization signals through its layers up to the applications. In the SPS:

- Existing protocol stacks are supported unchanged. For example, the Internet Protocol (IP) stack can be accommodated as follows. For sending, sources collect frames and forward them when bands to the desired destinations are signaled. For receiving, the protocol works as in IP. The Service Access Points (SAPs) available at each IP layer can be supported unchanged.

- Existing stacks are extended with novel source and destination synchronization. For example, the IP stack can be enhanced with real-time service support. Real-time demanding frames are forwarded only when the network signals the required QoS to their destinations. The SAPs are extended with these novel services, but otherwise kept unchanged.

1.4 An Isochronet Switch Design and Implementation

1.4.1 The Problem

One of the problems addressed in this thesis is how to build an Isochronet switch (Isoswitch). The switch must operate at giga bit per second rates and ideally use only simple (that is, off-the-shelf) components.

1.4.2 Main Results

Electronic switch design and implementation. An electronic Isoswitch has been designed and implemented. It has four input and four output ports each operating at 1 Gb/s. The overall organization is depicted in Figure 1.2. The Host machine computes band allocation information off-line with respect to data transmission and reception. Sporadically, it stores the new configurations in a table inside the Control Unit. The Control Unit contains the arbitration logic that, based on the current configuration, sets up the switching fabric to provide the desired input and output connectivity. The configurations are changed over time according to the band allocations. The Input Line Card receives bit-serial data from the network trunks and converts them into bit-parallel words that are routed to the proper output ports by the switching fabric. Similarly, the Output Line Cards converts bit-parallel words into bit-serial streams to be transmitted through the outgoing network trunks.

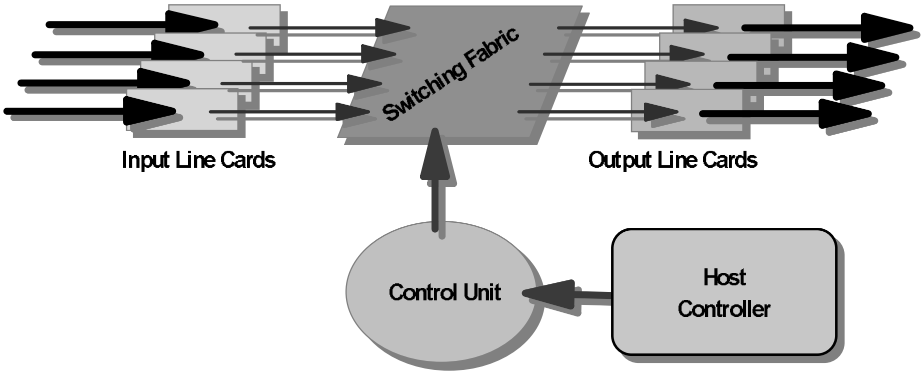


Figure 1.2: Isochronet switch organization

The design addressed the following challenges (where *n* is the number of input lines and *m* is the number of output lines):

- The design is scaleable with respect to number of channels and link speeds. The overall arbitration complexity is *O*(*nm*) and its latency is *O*(*n*).
- The implementation is easy to integrate with existing hardware. There is no adaptation among different protocol structures which makes the interface to other network devices simple.
- The Isoswitch can provide novel services. The first new service is the provision of synchronization signals that can be used at periphery nodes to schedule transmissions with required QoS. The second new service is direct frame forwarding without adaptation among different frame structures. The third new service is guaranteed (as opposed to statistical) QoS provision.

Interface card to a SPARC 1 machine. An interface card between the Isoswitch and a Sun SPARC 1 machine has been built. The card has a nominal throughput of 22 Mb/s. The card signals the beginning of band to the SPARC processor and can thus be directly used to implement LTM and SPS.

Optical switch design. A preliminary design of an all-optical switch implementation is proposed. An all-optical realization of Isochronets must avoid buffering at intermediate switches. The design proposed uses Wavelength Division Multiplexing (WDM) [Acampora and Karol 89, Brackett 91, Dono et al. 90] to allocate one wavelength for each band.

The implementation has a few advantages:

- No conversion of en-route signals to the electronic domain, which makes possible implementations potentially at hundreds of terabits per second.
- All routing trees are open at all times, thus contention band synchronization is not necessary.
- Fewer wavelengths are necessary than in pure WDM, namely one per band that, in the worst case, is the number of destinations in the network.

1.5 Performance Characterization

1.5.1 The Problem

The goal of the performance study is to evaluate the performance of RDMA when compared with more traditional switching techniques: PS and CS. The study is carried using both analytical and simulation models.

1.5.2 Main Results

Analytical approximations. The time-dependent behavior of RDMA complicates the performance study. Usual analytical performance analysis techniques ignore time-dependent (or transient) behaviors when simplifying models in search for tractable solutions. This study derived approximate closed-form formulas for end-to-end delay and loss in RDMA-, RDMA+, and RDMA++ for a single node and Poisson arrivals with mean rate λ. The analytical approximations are very accurate when checked with the simulation results, even for a network of nodes.

Under RDMA-, the whole routing tree can be seen as a server that transmits frames with sizes distributed according to some generic distribution $B(x)$ with mean rate μ. Whenever the server is transmitting, arriving frames are lost. If λ is the mean rate at which frames are arriving to the tree and $\tilde{z}$ is the inter-departure time for successfully transmitted frames. Then, the distribution of $\tilde{z}$ is:

$$\Pr\{\tilde{z} > z\} = e^{-\lambda z} - \int_0^z \overline{B}(z-x)\lambda e^{-\lambda x}\,dx \tag{1.1}$$

From this expression the mean loss rate can be computed as (where $\rho = \lambda/\mu$ is the tree utilization):

$$L = \frac{\rho}{\rho+1} \tag{1.2}$$

One can see that as the load reaches 1, the maximum loss in the system is 50%.

Additionally, there is no queueing delay in RDMA- and the mean delay is equal to the mean transmission time $(1/\mu)$:

$$D = \frac{1}{\mu} \tag{1.3}$$

Under RDMA++, the whole tree can again be seen as a server. It is active during the band (of size *U*) to the tree root and then goes on vacation (of size *V*) during other bands. From the point of view of an arriving frame when the system load is not very high, it needs to wait for the following events before being serviced. First, it needs to wait for the service of all frames in the queue upon its arrival plus the residual transmission time (or the time left to complete transmission) of the frame in service. Second, it needs to wait for the residual vacation time upon arrival. Finally, it needs to wait for the frame transmission. The formula that approximates the mean delay of a frame in RDMA++ is the sum of three expressions, where $\overline{x^2}$ is the second moment of the frame size distribution:

$$D = \frac{1}{2}\frac{\lambda\overline{x^2}}{(1-\rho)} + \frac{1}{2}\frac{V^2}{(U+V)}\frac{1}{(1-\rho)} + \frac{1}{\mu} \tag{1.4}$$

One can identify the first expression in the sum in the right hand size as the mean queueing delay for an M/G/1 system. The third expression is the mean transmission time by definition. The second may be interpreted as the probability of arriving during the vacation period ($V/(U+V)$) times the mean delay due to vacation ($V/2$) and the penalty due to the system load ($1/(1-\rho)$).

To be exact, the formula should additionally take into account vacations that occurred while servicing frames in the queue. The computation of such additional delay is not trivial because the expressions are not linear. Nonetheless, when the load in the system is low or medium, such added delay does not happen most of the time because all frames in the queue are serviced before the tree goes on vacation. The formula in (1.4) is thus a good approximation for low to medium loads.

Finally, there is no loss in RDMA++ and:

$$L = 0 \tag{1.5}$$

RDMA+ mean delay can be approximated as the mean delay in M/G/1 if the band size is large compared to the mean frame size:

$$D = \frac{\lambda \overline{x^2}}{2(1-\rho)} + \frac{1}{\mu} \tag{1.6}$$

Finally, the loss in RDMA+ can be approximated using the mean number of frames in an M/G/1 system. This is also the mean number of losses per band that can then be approximated as:

$$L = \frac{\lambda \overline{x^2}}{2U(1-\rho)} + \frac{1}{\mu U} \tag{1.7}$$

When the band size is large, the loss tends to 0.

Simulation studies. The main goal of the simulation study is to compare RDMA with respect to PS and CS for Poisson arrivals and bursty on-off arrivals. Figure 1.3 depicts a typical behavior when the frame size is fixed at 53 bytes, the cycle size is 125 µs, the band size is 25 µs, and transmissions happen at 2.4 Gb/s. The load axis is the mean input frame rate and the delay axis indicates the mean frame delay in the system.

For low input loads, PS outperforms RDMA and CS because there is no admission delay waiting for a band or circuit. When PS needs to guarantee QoS, the performance advantage may vanish due to admission control mechanisms. When the load increases, the frame header processor becomes saturated and then RDMA outperforms PS. CS always performs worst than RDMA because the penalty incurred waiting for a circuit is larger than the one waiting for a band.

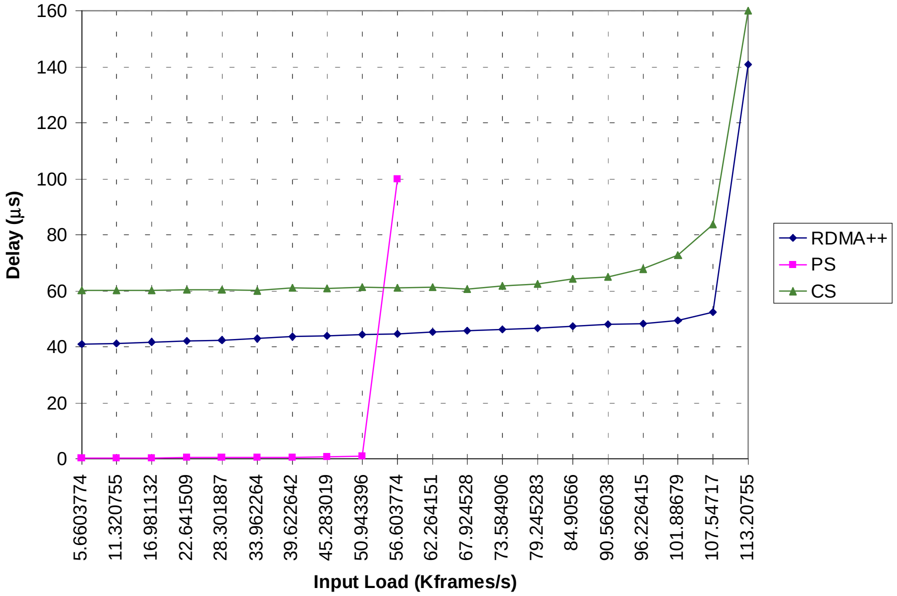


Figure 1.3: Mean frame delay comparison (in µs) for Poisson arrivals and deterministic service (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 25 µs, the cycle period 125 µs)

1.6 Organization of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 analyzes other HSN architecture proposals with respect to a set of desirable characteristics for HSNs. Chapter 3 is a detailed account on the concepts in Isochronets and RDMA. Chapter 4 elaborates further on Isochronet protocols for tree allocation, band allocation, and band synchronization. Chapter 5 details LTM and SPS and overviews how existing stacks can be enhanced to provide real-time services. Chapter 6 explains the Isoswitch electronic design and implementation as well as the optical design. Chapter 7 uses analytical and simulation tools to study the performance of Isochronets and compare it with PS and CS. Chapter 8 suggests future work in Isochronets. Finally, Chapter 9 concludes.

Chapter 2 ________________________

Related Work

2.1 Introduction

2.1.1 The Goal

This chapter has two goals. First, it identifies the main desirable features that HSN Switching Architectures (HSNSAs) should posses in order to efficiently support application requirements. Second, it uses the defined features to characterize and compare proposed HSNSAs.

An HSNSA must be designed to take maximum advantage of high-speed transmissions while maintaining backward compatibility with existing technology. To best profit from high-speed transmissions, an HSNSA should:

- Minimize processing and queueing latency in the end-to-end transport path;
- Support speed-scalability for link bandwidth ranging from hundreds of megabits to terabits per second;
- Support scalability for large size networks (ranging from ten to hundred million end nodes);
- Support efficient interfacing with end nodes.

Toward backward compatibility, an HSNSA should support:

- End-to-end transport of multiple protocol frames;
- Efficient internetworking with existing stacks and media.

Additionally, high-speed transmissions enable a whole new set of applications with novel requirements. For these, it is important to support:

- Tunable guaranteed QoS of end-to-end transport;
- Asynchronous, synchronous and isochronous traffic streams;
- Efficient bandwidth sharing and flexible adaptation to service demands.

These features will be used to evaluate architectures for HSNs. The following sections start by analyzing the more traditional architectures: Circuit Switching (CS) [Halsall 92, Tanenbaum 88] and Packet Switching (PS) [Halsall 92, Tanenbaum 88]. These analyses are used as cornerstones to study the following HSN extensions of the basic architectures: Burst Switching (BS) [Amstutz 83, Haselton 83], Asynchronous Transfer Mode (ATM) [De Prycker 93, Le Boudec 92], Metanet [Ofek and Yung 90, Ofek 94], Highball [Mills et al. 90], Wavelength Division Multiplexing (WDM) [Acampora and Karol 89, Brackett 91, Dono et al. 90], and Linear Lightwave Networks (LLNs) [Stern 90]. It will be concluded that these architectures may not fully support some of the features outlined.

HSN architectures that are not switched in nature will not be considered in this study. These architectures are usually restricted to local or metropolitan size networks. Examples include FDDI and DQDB [Bertsekas and Gallager 92].

2.1.2 Chapter Organization

This chapter is organized as follows. Section 2.2 further discusses the features to be supported by an HSNSA in descending order of importance. Sections 2.3, 2.3, 2.5, 2.6, 2.7, 2.8, 2.9, and 2.10 analyze the proposed architectures for HSNs. Finally, Section 2.11 summarizes.

2.2 Qualitative Characterization of HSNSAs

This section defines qualitative measures that will be used in evaluating HSNSAs. The measures capture how well an ideal HSNSA will support diverse application needs at the user interface to the network. The measures are the following:

1. *Support efficient end-to-end transport of multiple protocol frames.* On the one hand, QoS can be severely degraded if internal HSNSA operations are anchored on a single protocol suite. In such scenario, the adaptation between external protocols and the internal one may severely delay and complicate end-to-end communications. On the other hand, multiple protocols already inhabit today's interconnected networks providing services that cannot be eliminated after transition to high-speed transmissions. The ideal HSNSA must efficiently support the transmission of multiple protocols without adaptation between them.
2. *Support tunable guaranteed QoS.* Application needs range from best-effort service to guaranteed QoS. For example, electronic mail delivery applications can use best effort services at low cost while an embedded multiprocessor system must have strict QoS guarantees at higher cost. An HSNSA must provide flexibility in resource allocation to enable fine tuning of QoS needs given budgetary constraints.
3. *Support asynchronous, synchronous and isochronous traffic streams.* Future integrated network services include an array of synchronization needs, that is, asynchronous (e.g., electronic mail delivery), synchronous (e.g., embedded parallel processors), and isochronous (e.g., voice communications) streams. The provision of integrated services to higher layers may present difficulties if the underlying HSNSA is synchronous or asynchronous in nature. For example, if the HSNSA is

asynchronous, synchronization at higher layers will incur errors bound only by the maximum end-to-end jitter. It may be unacceptable for some applications. On the other hand, if the HSNSA is synchronous, the provision of asynchronous services may be inefficient or eliminate resource multiplexing advantages. The ideal situation is thus to directly support all synchronization needs in the HSNSA.

4. *Minimize processing and queueing latency.* Network processing is expected to be a major bottleneck due to the imbalance between processing and transmission speeds and thus must be minimized. For example, at 2.4 Gb/s transmissions, an ATM cell has to be processed in 177 ns. Additionally, network queueing delays are unpredictable and must happen under controlled supervision, with the option of being completely eliminated for very demanding applications. An ideal HSNSA should minimize processing and control queueing to cope with high-speed transmissions.
5. *Support speed-scalability for link bandwidth ranging from hundreds of megabits to terabits per second.* HSNSA operations cannot rely on any assumptions about link speed. For example, when it is necessary to process frame headers, the available processing time per frame will decrease with link speed. As a consequence, the architecture will not adapt once faster links are installed. An HSNSA must avoid any operation at link speed pace.
6. *Support efficient internetworking with existing stacks and media.* Current protocol stacks will continue to exist in future networks because a large portion of legacy software uses them. Additionally, an HSNSA should not rely on any particular medium. For example, an HSNSA must be capable to operate using electronic or optical transmissions to maintain compatibility with existing transmission technologies on which present protocol stacks rely. An HSNSA must provide compatibility with existing stacks and media either by directly supporting them or by efficient interfaces.

7. *Support scalability for large size networks (ranging from ten to hundred million end nodes).* HSNs are expected to grow rapidly because they will bring integrated communication services all the way to the subscriber premises (which in many cases are homes). An ideal HSNSA must thus support large networks.
8. *Support efficient interfacing with end nodes.* The interface to the network can be a major bottleneck in end-to-end QoS performance. For example, if the interface needs to allocate resources prior to any transmissions, the latency may become unbearable and the complexity in designing the interface may render costs prohibitive to some markets. An HSNSA must enable the implementation of efficient and relatively simple interfaces.
9. *Support efficient bandwidth sharing and flexible adaptation to service demands.* Bandwidth sharing is the driving force to lower cost services. Nevertheless, QoS can be severely compromised if sharing is not strictly controlled and adapted to changes. Resources can be multiplexed to provide lower cost services when the network load is low. Additionally, an ideal HSNSA must be able to adapt the degree of multiplexing to protect QoS demanding applications from sudden load changes.

In the following sections, proposed HSNSA will be evaluated with respect to these measures. The number associated with each measure in this section will be used as a reference to it in the following sections.

2.3 Packet Switching

PS or store-and-forward networks operate by fragmenting information steams into variable size frames (or packets). Frames are equipped with special control headers that embody routing information such as destination addresses. PS is commonly used in data networks, such as the Internet [Comer 91].

Switching in PS works as follows. Nodes in the path parse each incoming frame to extract the destination addresses. These addresses are then used to index a routing table that maps destination addresses into respective node output port identifiers. These parsing and mapping functions are performed by the processor at each node. It is in effect a shared resource among all frames crossing the node. The union of all the routing tables in network nodes form a set of routing trees, that is, a spanning tree that defines paths from all the sources in the network to a particular destination (the root of the tree) for each destination.

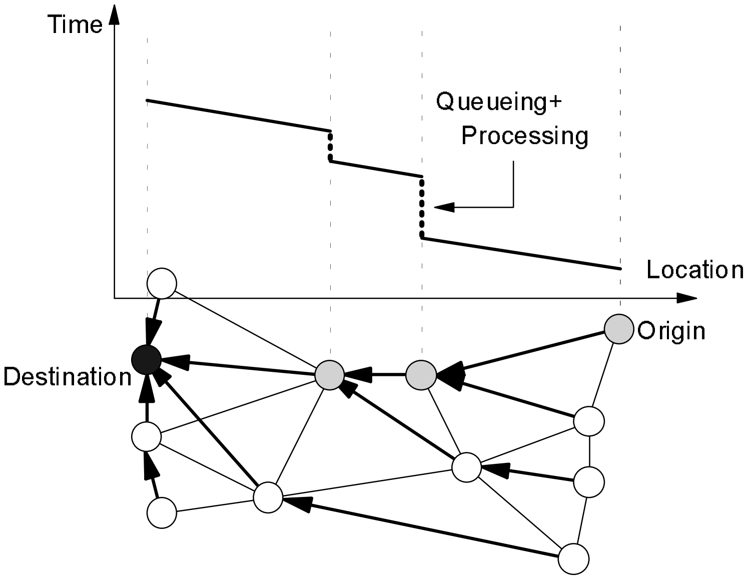


Figure 2.1: Store-and-forward routing on trees

Figure 2.1 highlights one such routing tree. The location and time diagram for a typical frame moving along the routing tree is depicted on top of it. Each point in the graph represents the location and time of the frame motion. Initially, the frame incurs some transmission and propagation delay while moving to the next node in the path to the destination. When it reaches the next node, the frame contends for resources such as processor and outgoing link. It may thus incur some random motionless delay while waiting for these resources and while being processed. This period is depicted as a vertical line on top of the node. Eventually the frame moves to the next node in the path where similar disrupted motions take place.

It is important to notice that contention happens among all frames crossing the node (to different destinations) because they must share the same processor. Thus, the random queueing delay is dependent not only on contention due to the frames sent to one particular destination, but also on other frames crossing the node.

In PS:

1. Muti-protocol support is not efficient because it involves cost intensive operations to adapt between multiple external frame structures and the internal one. These operations usually incur fragmentation and reassembly of frames and require considerable processing. As the link speeds increase, the time to perform such operations becomes more and more limited.
2. QoS cannot be guaranteed or tuned. Upon contention for network resources such as processor or outgoing links, frames incur unpredictable queueing delays or eventual loss when queues overflow. As a consequence, end-to-end delays, jitters, or effective bandwidth use are difficult to predict. One partial solution to support QoS in PS is to suppress the very sources of traffic randomness via global admission control and policing mechanisms [Sidi et al. 89]. During the admission phase, resources are requested with associated QoS parameters. For example, bandwidth may be requested with some maximum allowed end-to-end delay and loss rate. These parameters are used by the underlying transport layer to allocate network resources (bandwidth, buffers, processing, etc.). During the policing phase,

sources are inspected to see if they are generating traffic that complies with the requested QoS parameters. For example, if the source requested a specific bandwidth, it cannot transmit over the agreed parameter. Admission delays and reduced network utilization are traded-off against reduced contention. Nevertheless, QoS can only be statistically guaranteed. The interruptions seen by a source depend on aggregated contention traffic of other random sources. For example, all sources contend for the processor and thus the queue at the processor contains an interleaved mixture of frames from all sources. Statistical QoS results in increased admission delays, reduced network utilization, and lower effective bandwidth seen by sources. If contention is to be eliminated with high probability, all resources need to be pre-allocated and consequently the very value of PS for on-demand resource allocation becomes questionable.

3. Asynchronous services can be provided directly, but limited support exists for synchronous or isochronous services. Synchronization can only be provided by upper protocol layers and is thus limited by the maximum end-to-end jitter in the network. Since in PS no such bound exists, synchronization is difficult to attain.
4. Processing takes place per frame. To switch, frame headers need to be parsed in order to extract destination addresses. Processors use these addresses to index routing tables and find the proper outgoing node port in the route to the destination. Additionally, contention for network resources such as processors and links is resolved by queueing frames. Such queueing delays are unbound and can contribute significantly for end-to-end delays.
5. Switching speed may represent a problem for speed scalability. When link speeds are increased, processing speeds must be increased accordingly. Nevertheless, processing technology may not be capable to cope with recent link speed. For example, at 2.4 Gb/s, a 53 byte packet has to be processed in 177 ns or about 17 instructions on a 100 MIPS machine.

6. Internetworking is complex due to the difficulty of supporting multiple protocols. Additionally, PS is difficult to adapt to non electronic media. Implementation of PS in all-optical networks would involve all-optical frame header processors. The technology to implement such processors is current beyond the state of the art.
7. Scalability in size is better supported than in other architectures. Frames allocate resources on-demand strictly to use them. That is, at any given time, the network links and processors are being used by the frames at the head of the queues. When the queue is empty, links and processors are ready for on-demand use. Since resources are never allocated to idle sources, this architecture is likely to support more nodes than others given a constant set of resources.
8. The interface is simple to implement. End nodes can directly relay frames at any time without resource pre-allocation. Additionally, since there is no need to pre-allocate resources, frame delay to access to resources is minimal when network load is low (because there is minimal queueing overhead due to contention in this case).
9. Bandwidth sharing is one of the most appealing features. Resources are shared and only allocated on-demand when necessary. Potentially, resource wastage does not exist in PS.

2.4 Circuit Switching

In CS, a physical connection is established between the source and destination nodes. Each link in the network is multiplexed as follows. The physical layer generates a periodic frame (not to be confused with PS or higher layer data frames) that is divided into slots that are then assigned to connections. Slots are usually 8 bits long and the frame is repeated at 8 KHz, providing a throughput of 64 Kb/s per connection. CS is normally used for voice communication. A connection is established by mapping idle incoming slots into idle outgoing slots in each node in the source to destination path. This switching techniques also called Time Division Multiplexing (TDM).

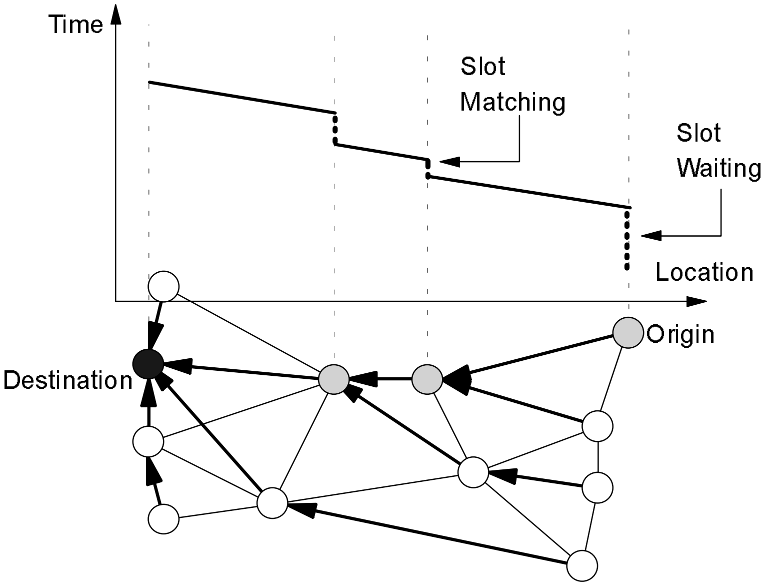


Figure 2.2: Circuit motion on trees

Switching at intermediate nodes is accomplished by mapping incoming ports and slots onto outgoing ports and slots, regardless of information content, as depicted in Figure 2.2. The delay at each node depicted in the figure is to match incoming and outgoing slots and is bound by the maximum frame size. Extensions of the CS concept [De Prycker 93] have been proposed to overcome some of the drawbacks to be described in this section, but they greatly increase the implementation complexity.

In CS:

1. Frames are transported transparently, that is, without header processing at internal switches. As a consequence, multiple protocol frames can be simultaneously switched without adaptation to a common internal frame structure.

2. QoS is strictly guaranteed because frame slots are dedicated to sources and destinations for the duration of the connection. Nevertheless, it cannot be finely tuned for applications that do not have strict QoS demands. Each communication uses dedicated resources and consequently has full QoS guarantees.
3. On the one hand, synchronous and isochronous communication streams are supported directly by the architecture. Signaling from frames and slots within frames can be used to synchronize sources and destinations. On the other hand, asynchronous communications have to be emulated within a synchronous infrastructure, with unnecessary added delays for media access and potential loss of resource multiplexing.
4. There is no frame header processing. Switching occurs solely by matching incoming and outgoing link frame slots. Additionally there is no queueing inside the network because slots are dedicated to the connection.
5. Link speed increase may generate problems in synchronization and switching speed. For example, if the slot size is 8 bits and transmissions happen at 1 Gb/s, each slot lasts 8 ns. In such scenario, peer nodes in a link must synchronize and switch slots within nanoseconds. A potential solution is to increase the slot size but this will incur bandwidth waste since many applications such as voice conversations will occupy only a few bits per frame. Additionally, the circuit reservation delays constitute a significant factor in the end-to-end communication efficiency when links are fast. For example, the protocol exchanges to allocate the circuits may incur end-to-end propagation delays in the order of hundreds of milliseconds when sources and destinations are far apart. Nevertheless, an exchange of 1 Kbyte of information at 1 Gb/s takes only 8 μs.
6. Internetworking with existing stacks is simple due to the support for multiple protocol transport. That is, multiple protocols can coexist on the same medium without adaptation. Additionally, TDM may function in multiple media, ranging from

all-optical to all-electronic networks.

7. The increase in the number of nodes is a potential problem. The more end-nodes exist in the network, the richer the network connectivity and bandwidth must be to support end-to-end dedicated channels.
8. The interface is relatively complex to implement because it needs to allocate and maintain circuits prior to communication. It also must be fast to synchronize communications to frame slots at high speeds.
9. Bandwidth sharing or adaptation does not exist. Slots are dedicated to end-to-end connections and may not be used when the corresponding source is idle.

2.5 Fast Packet Switching and Asynchronous Transfer Mode

Many research efforts have been undertaken in recent years to develop Fast Packet Switching (FPS) architectures [Ahmadi and Denzel 89, Chao 91, Eng et al. 89, Giacopelli 91, Huang and Knauer 84, Lee 90, Murakami 91, Oie et al. 91, Pattavina 90, Tobagi 90, Tobagi et al. 91, Turner 86, Turner 88, Venkatesan 92, Widjaja and Leon-Garcia 92, Yeh et al. 87, Yum and Leung 92] of which Asynchronous Transfer Mode (ATM) is a representative example. FPS uses a combination of PS with virtual connections to transfer information. The idea is to attach stochastic QoS guarantees to connections while switching according to PS. Connections are monitored and controlled to deliver the negotiated QoS. ATM operations will be described as a representative architecture of FPS.

ATM networks transmit 53 byte fixed size cells structured as 5 byte header and 48 byte payload. ATM switches cells using Virtual Paths (VPs) and Virtual Circuits (VCs). A VP is a channel that may contain one or more VCs. Each ATM cell contains two identifiers: a VP Identifier (VPI) and a VC Identifier (VCI). Within a VP, switches use only VPIs in each cell to take switching decisions. When a switch connects different VPs, both VPIs and VCIs are used. Prior to transmission, a VC must be established.

ATM can be regarded as a combination of PS with added CS characteristics. Switching per se operates like in PS. Nevertheless, a connection must be established prior to transmission like in CS and cells in the same connection follow the same path to the destination. ATM inherits many of its operational characteristics from the combination of PS and CS:

1. Multiple protocol transport may be inefficient and difficult to implement. It involves fragmentation of frames into cells and their transmission over end-to-end VCs when the latter is available for the requested end-to-end connection. If VCs are not available, frames may need to cross multiple VCs with corresponding fragmentation and reassembly at each router in the path further compromising efficiency.
2. ATM provides only a coarse level of QoS guarantees as far as cell structure goes: cells are either high or low priority. Guarantees can be provided only in a probabilistic sense because the underlying technology is PS with characteristic random behavior due to cell queueing and loss. For bursty traffic scenarios guarantees require worst-case input admission control to avoid internal cell queueing or loss. Conservative admission control mechanisms may lead to significant waste of bandwidth.
3. Asynchronous services can be provided directly, but limited support can be given for synchronous or isochronous services due to the intrinsic end-to-end jitter in ATM operations.
4. In similarity to PS, processing takes place per cell and queueing is used to overcome contention.
5. In similarity to PS, speed scalability support may generate problems in switching

speed. To attenuate this problem, many FPS switch implementations have been proposed (as the ones referenced in the first paragraph of this section). Many times they use multiple specialized processors or special interconnection fabrics. Nevertheless, they significantly add to the implementation complexity and cost and may still be difficult to scale due to the inherent processing per cell.

6. In similarity to PS, interconnection is complex.
7. Scalability with number of nodes is somewhere between CS and PS. On one hand, if QoS must be guaranteed for all connections, independent VCs must be allocated with no resource sharing. On the other hand, if no guarantee is necessary, VCs may share resources and thus maximize multiplexing as it is the case in PS. Scalability is dictated by the number of VCIs that can be allocated and the QoS that must be provided to end nodes.
8. In similarity to CS, transfers need to be preceded by a VC establishment. Transmissions can then proceed using the VCI. Nonetheless, transmissions need to cope with the agreed QoS parameters. A policing mechanism monitors and controls violations. The implementation of such policing mechanisms may further complicate the interface.
9. Bandwidth sharing happens between QoS demanding and non-QoS demanding traffic by using the priority mechanism in cells. It is thus limited to a two-level mechanism.

2.6 Burst Switching

Burst Switching (BS) is an extension of the PS concept to switch bursts of information, usually integrating digitized voice and data. Bursts are generated from speech spurts or data messages and are embedded into a frame (or burst) with a header that identifies its destination address and a trailer (because bursts have variable size). The novel aspect of

BS networks is that they disperse switching decisions into hundreds and thousands of processors connected through shorter link lengths (thus permitting higher bandwidth links). The links use TDM. When a burst is sent, one of the TDM slots is allocated to the burst.

BS is thus very similar to PS with very fast switches and TDM links. The following highlights only the points where BS and PS differ.

2. BS can guarantee QoS within bursts, but not between bursts. Additionally, to support voice communications, the switches are made very fast by distributing processing and thus increasing processing throughput to a level that minimizes contention.
3. BS supports synchronization within bursts by using TDM signals.
5. Even though fundamentally not very scaleable as it is the case with PS, link speed increase can be better accommodated since header processing happens per burst (which is larger than a frame).

2.7 MetaNet

The MetaNet network operates by embedding ring topologies within arbitrary networks. MetaNet uses a type of deflective routing algorithm through the rings by which packets are forwarded following the ring structures. The same algorithms for media access used for ring topologies are employed in the MetaNet to accomplish fairness, deadlockless, and dynamic self-routing.

The MetaNet operates essentially as a ring network and thus incorporate many of the characteristics of PS networks. Here are a few more specific observations:

2. QoS can be difficult to guarantee. Statistical QoS can be accomplished only through the addition of special features to provide virtual circuit allocation in the network. These mechanisms are orthogonal to MetaNet operations. Tuning of QoS seems not to be possible.
3. Asynchronous services can be provided directly, but synchronous or isochronous services can only be provided in limited fashion as in ATM networks. The underline transport does not contain synchronization signals.
4. Processing takes place per packet at each node to check if the packet is destined to the current node. Queueing is needed to solve contention for network resources such as processors and links.
5. Link speed scalability support may generate problems in switching speed. This may be attenuated by the fact that the kind of processing in a ring topology can be completely realized in hardware.
7. Scalability with number of nodes is supported by the introduction of clusters of rings to improve overall available bandwidth.
8. End nodes have to perform ring media access functions that can be quiet elaborate.
9. Bandwidth sharing only happens for asynchronous traffic, since the synchronous traffic must use separated virtual circuits.

2.8 Highball

In the Highball network proposal, nodes schedule traffic bursts by configuring switches to support uninterrupted motion similar to train motions through intersections. Nodes broadcast requests to all other nodes, specifying their data transmission needs to all possible destinations. This information is then used to compute a train schedule at each node and establish time intervals during which output links are dedicated to specific input links. The scheduling problems are NP-complete and are thus approximated through heuristic solutions. Highball networks are geared to serve traffic that can tolerate the latency delays between requests to transmit and their granting.

Highball networks are similar to CS in that no processing of frames occurs in the network. Nonetheless, they are more flexible because bandwidth allocation is done according to traffic needs and is not bound to a fixed slot. The following highlights the main points where the Highball architecture and CS differ:

2. QoS is always guaranteed, but there is some flexibility in that bandwidth can be allocated according to traffic demands.

5. Speed scalability has less impact than in CS because slots assigned to particular end-to-end connections in Highball have variable size and tend to be much bigger than the CS fixed size slot.

7. Network growth can increase resource allocation complexity. Since the scheduling problems are NP-complete, network size may slow the resource allocation procedures.

9. Bandwidth allocation in Highball is more flexible and can very closely mimic usage patterns, thus decreasing the probability of wasting bandwidth assigned to idle sources.

2.9 Wavelength Division Multiplexing

Wavelength Division Multiplexing (WDM) networks provide dedicated access to destinations via appropriate allocation of wavelengths. Routing is accomplished by configuring nodes to switch wavelengths in order to provide source-destination connectivity. Contention among simultaneous transmissions to the same destination must be solved at switches. Unfortunately, optical tuning of switches at incoming traffic rates is beyond the current state of the art. To cope with this limitation, current implementations of switched WDM use dedicated wavelengths between node pairs. Packets may only be sent to a node’s peer. At the peer, packets need to be processed (and thus be converted to the electronic domain) in order to determine the destination route. WDM may operate either in CS or PS modes. Nevertheless, multiple hop (switched) versions usually employ PS. This will be the assumption in this section.

The big advantage in WDM is that the optical bandwidth available is much larger than the transmission speeds achievable by end nodes. Multiple channels may be created using different colors. In fact, the network topology may be completely changed by simply tuning optical transmitters and receivers to build new links using colors. Each color corresponds to a different link. The following highlights the main differences between WDM and PS:

4. In addition to normal frame header processing, WDM network need to perform optical to electronic frame conversion which may significantly delay communications in the current state of the art.
6. WDM relies on optical technology and thus internetworking must incur conversions between optics and electronics.
7. The optical bandwidth is much larger than the traffic electronic end nodes may

generate and thus can be divided in many wavelengths that can be used as multiple independent virtual networks. As a consequence it can support a larger number of end nodes.

8. The interface must perform media conversion between optics and electronics which may involve complex circuitry.

2.10 Linear Lightwave Networks

Linear Lightwave Networks (LLNs) provide all-optical communication using wavebands and wavelengths within wavebands. Wavebands are used to create various channels in each link while wavelengths are used to provide end-to-end connectivity. Optical switches can switch only wavebands (wavelengths are indistinguishable for them), while end nodes can separate wavelengths within wavebands. All switching operations are optical. They are realized by generating power at an output ports from a linear combination of the powers at the input ports. Wavebands are broadcast through all possible paths that carried wavelengths need to traverse. When accepting a new call, a wavelength and a path to the destination must be assigned it. The procedure toward this goal is to examine each link and verify that the candidate wavelength is not being broadcast through it.

It is important to notice that there are situations in which it may not be possible to assign the same wavelength to two distinct pairs of nodes even if they have disjoint paths. The reason is that LLNs have to broadcast wavebands (and consequently wavelengths within them) through all paths the wavebands connect. As a consequence, the assignment of different wavelengths for connections may become intricate. It is necessary to make sure that all combined wavelengths in a given link do not interfere with the new wavelength that is being assigned. The assignment of wavelengths to incoming calls is NP-complete and needs to be approximated using heuristics.

LLNs also divides bandwidth in the waveband domain, in similarity to WDM. Nevertheless, no processing or buffering is necessary at intermediate switches. It is designed to serve applications that can tolerate the potentially long call set up delay to find a proper wavelength for the call.

LLNs operate in a way very similar to CS. The main departure from CS occurs due to the large bandwidth available in the optical domain, due to the bandwidth division among wavelengths (and not slots), and due the fact that all internal switching is optical. Here are some more specific differences:

3. Synchronous and isochronous communication streams can be supported because the network guarantees the communication QoS. Nevertheless, there is no signaling provided by the network.
5. Link speed scalability is not a problem in LLNs because bandwidth is divided in wavelengths as opposed to CS time slots.
6. LLNs must operate in the optical domain and thus electronic conversion is necessary for interconnection.
7. The network can grow due to the enormous bandwidth available in the optical domain.
8. The interface must perform media conversion between optics and electronics that may involve complex circuitry. Additionally, the allocation of wavelengths may complicate the interface design.

2.11 Summary

Proposed HSNSAs have been evaluated using a series of important qualitative measures. Figure 2.3 summarizes the main conclusions. Each column represents an HSNSA and each row represents a measure. A gray mark is placed only at the intersection of HSNSAs that fully support the corresponding measure.

Many measures cannot easily be satisfied by current architectures. The most interesting findings are that:

- No architecture fully support both guaranteed and tunable QoS;
- No architecture has explicit support for the three types of synchronization needs;
- No architecture supports the whole set of measures defined.

The next chapter develops a new architecture for HSNs that incorporates each of these features.

	Circuit Switching	Packet Switching	Fast Packet Switching	Burst Switching	MetaNet	Highball	Wavelength Division Multiplexing	Linear Lightwave Networks
Efficient multiple protocol frames	▨					▨		▨
Tunable guaranteed QoS								
Asynchronous, synchronous, and isochronous streams								
Minimal processing/queueing	▨					▨		▨
Speed scalability						▨	▨	▨
Internetworking with existing stacks/media	▨					▨		
Size scalability		▨	▨	▨	▨		▨	▨
Efficient interfacing		▨		▨				
Bandwidth sharing		▨	▨	▨	▨		▨	

Figure 2.3: Characteristics of proposed high-speed architectures

Chapter 3

Isochronets and Route Division Multiple Access (RDMA)

3.1 Introduction

3.1.1 The Problem

HSNs set forth novel challenges in designing a network architecture, the most important of which have been detailed in Chapter 2 and summarized in Figure 2.3. From the figure, it can be concluded that many proposed architectures address some of the challenges, but fail to tackle core aspects of HSN operations. The goal of this chapter is to develop a new HSNSA from the requirements set forth in Chapter 2.

3.1.2 Main Results

Isochronets. Isochronets is a novel HSNSA that satisfies the requirements in Chapter 2. The main guideline behind the Isochronets switching architecture design is to eliminate any processing in intermediate network nodes that is dependent on frame headers. This goal is achieved by making network operations independent of frame structures.

Route Division Multiple Access (RDMA). RDMA is the novel switching architecture in Isochronets. Switching in Isochronets happens based on time, similar to how traffic lights control road intersections. Traffic lights enable routes solely based on time, according to pre-set schedules. Accordingly, switching is Isochronets is mainly accomplished by mapping node input port to output port configurations based on time. The net effect is to divide bandwidth among routes. RDMA does not impose any frame structure and, consequently, multiple frames pertaining to diverse protocols can be switched without any adaptation inside the network. Additionally, by controlling the manner in which switching takes place, Isochronets may offer a spectrum of guaranteed QoS.

3.1.3 Chapter Organization

The reminder of this chapter is organized as follows. Section 3.2 presents Isochronets and their switching technique: RDMA. Section 3.3 evaluates the suitability of Isochronets for HSNs using the qualitative measures defined in Chapter 2. Section 3.4 positions RDMA in context with respect to PS and CS. Section 3.5 summarizes the main characteristics of Isochronets.

3.2 Isochronets and Route Division Multiple Access (RDMA)

Isochronets is a novel switching architecture tailored to addresses the issues discussed in the previous section. Isochronets *coordinate* traffic motions in the network over time, in similarity to how green waves coordinate traffic motions in roads. In the latter scenario, coordination is accomplished through synchronized traffic lights to create routes for uninterrupted and collision-free motions from source to destinations. Over time, the traffic lights are switched to cover all necessary routes.

In Isochronets, coordination is similarly accomplished by enabling a set of routes through periodic configuring of network switches. Switches set up routing trees, that is, routes from all nodes to a given destination. A frame is switched by following its path through an enabled tree. Coordination is among routing trees to solve routing ambiguities. That is, two trees that share a link in the same direction cannot be enabled concurrently. The coordination mechanism is called *Route Division Multiple Access (RDMA)* because it divides network bandwidth among routing trees. Routing trees become available periodically for a given interval of time or *band*.

3.2.1 Operation

In RDMA, the basic construct used to schedule traffic motion is a time-band (*green-band*) assigned to a routing tree, as depicted in see Figure 3.1. The green-band is an interval of time during which network switches are configured to route according to a particular routing tree (that is, the routing tree is *enabled*). During the green-band (shaded), a frame transmitted by a source will propagate down the routing tree to the destination root. If no other traffic contends for the tree, it will move uninterrupted, as depicted by the straight line. Additionally, intermediate nodes perform no processing that is dependent on frame header contents.

The green-band is maintained at switching nodes through timers synchronized to reflect latency along tree links. Synchronization is per band size, which is large compared to frame transmission time. It can thus be accomplished through relatively simple mechanisms. Routing along a green-band is accomplished by configuring switches to schedule frames on incoming tree links to the respective outgoing tree link. A source sends frames by scheduling transmissions to the green bands of its destination.

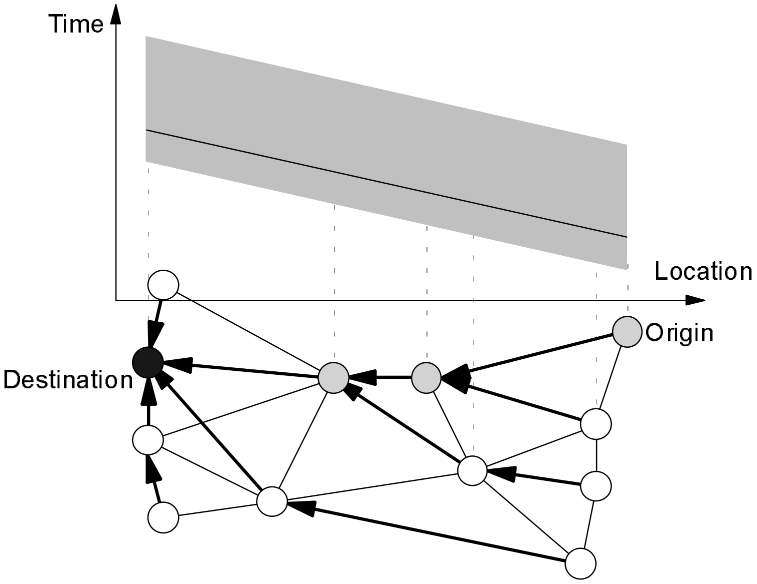


Figure 3.1: Green-band

In similarity to CS slots, green-bands allocate reserved network resources. However, the units to which resources are allocated are neither point-to-point connections, nor traffic bursts, but routes. Routes represent long-lived entities and, thus, processing and scheduling complexities can be resolved over time scales much longer than latency.

3.2.2 An Example

To illustrate the concepts in Isochronet operations, assume that the network topology is as shown in Figure 3.2. A possible routing tree allocation in the network is shown in Figure 3.3. Notice that the trees to A and C do not interfere and can coexist in the same band. The same is true for the trees to B and D. Nonetheless, for example, the trees to A and B interfere because they share links in the same direction.

A possible band allocation is depicted in Figure 3.4. The clock contains 12 units and is divided among the two bands necessary for the allocation in Figure 3.3. Trees A and C are active simultaneously (share the same band) between 12 and 8:15. Trees B and D are active between 8:15 and 12. In addition, the tree to B gives priority to source D between 10 and 11.

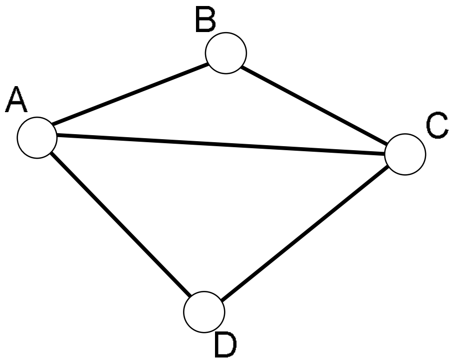


Figure 3.2: Example network topology

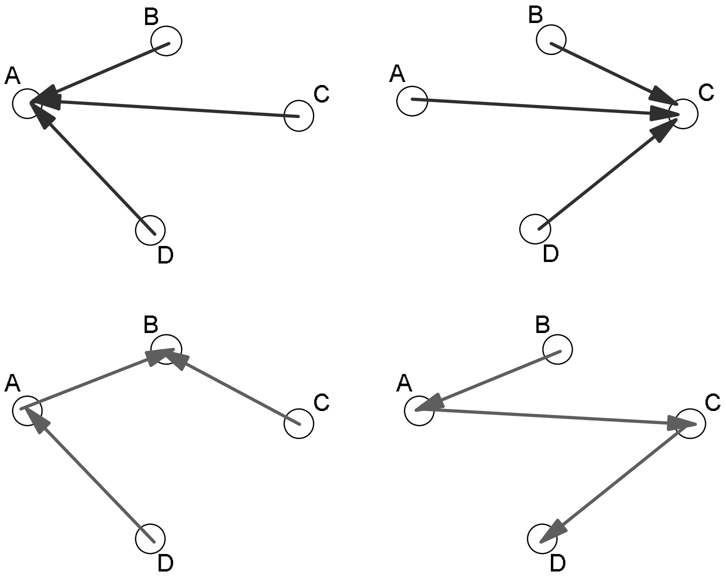


Figure 3.3: Example routing tree allocation

3.2.3 Band Allocation and Contention Resolution in RDMA

The allocation of synchronized time bands to routing trees and resolution of frame collisions are the primitive constructs used by RDMA to control traffic motions and QoS.

Green bands (or *contention bands*) are allocated according to the traffic demands per tree. The same band can be of different sizes at different nodes in the tree. Indeed, one can view a green band as a resource that is distributed by a node to its up-stream descendants, as long as the bands allocated to descendants are scheduled within the band of the parent.

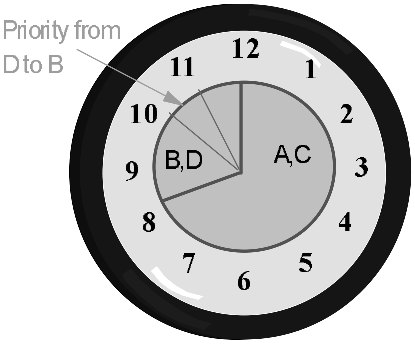


Figure 3.4: Example band allocation

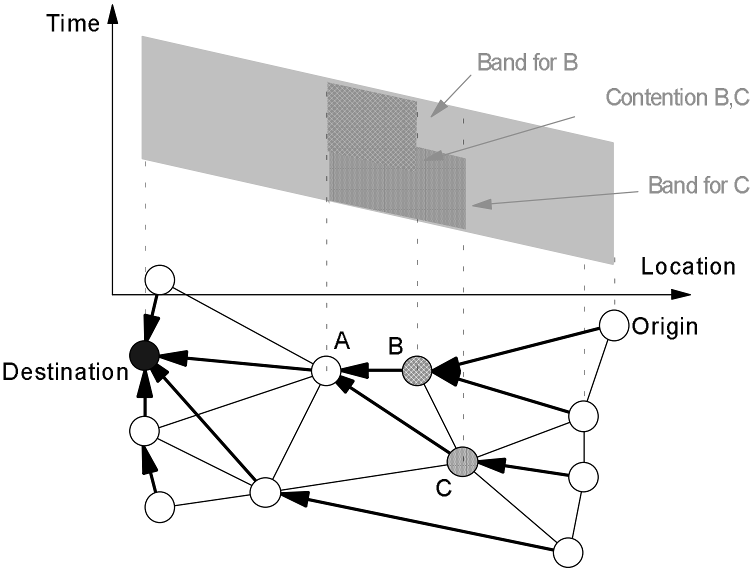


Figure 3.5: Control of contention through degree of band overlap

Figure 3.5 illustrates this concept by showing the band allocation at an intermediate node A to the destination. A partitions its band between its descendants B and C. Notice that the sizes allocated to B and C are different and only overlap partially. In the portion where the bands overlap, traffic from B and C contend in A toward the destination. In particular, if the bands allocated to two descendants do not overlap, their traffic does not contend. By controlling band overlaps, switches can fine-tune the level of contention and statistical QoS seen by traffic.

Similar considerations can take place in the allocation of priority bands. That is, one may view the bands distributed by node A in Figure 3.5 as priority bands. In such scenario, in the portion when both priority bands overlap, both B and C contend at A. In the other portions, either B or C has priority toward the destination. Notice that, for example, B having priority is equivalent to the whole subtree with root in B having priority to the destination. By finely tuning priority bands, a whole set of priority criteria can be built to solve contention for outgoing links in the trees.

It will be assumed in the reminder of this work that two priority bands in the same node do not overlap. This assures uninterrupted motions during priority bands that are allocated all the way from the source to the destination.

RDMA bands can also be *priority* or *multicast*, in addition to contention bands. Priority bands are allocated to sources requiring absolute QoS guarantees, similar to a circuit service. Traffic from a priority source is given the right of way, by switches on its path, during its priority band. Unlike CS networks, however, priority sources do not own their bands. Contention traffic may access a priority band and use it whenever the priority source does not. During a multicast band, the routing tree is reversed and the root can broadcast to any subset of nodes.

One may view these mechanisms to schedule traffic motions via band allocations as a MAC technique. The entire network is viewed as a routing medium consisting of routing trees. Bandwidth is time- and space-divided among these routes. Sources need access respective trees during their band times, seeing the network as a time-divided medium, much like Time Division Multiple Access (TDMA) [Halsall 92]. This similarity is the reason for the name Route Division Multiple Access.

Within a contention band, frames may contend competing for the outgoing link. The collision resolution mode used is designed in terms of signs "-", "+", and "++". In RDMA- only one frame proceeds, while the other colliding frames are discarded. In RDMA+, only one frame proceeds upon collision, but the others are buffered. Nevertheless, when the band finishes, all buffered frames are discarded. RDMA++ uses the same collision resolution technique as RDMA+, but stores frames beyond band termination and reschedules them during the next band.

Multiple simultaneous routing trees can schedule transmissions in parallel, that is, have simultaneous green bands, depending on the network topology. For an extreme example consider a fully connected network: all trees to all nodes can be simultaneously active without interference. In more realistic examples, significant parallelism can be accomplished. Figure 3.6 shows two non-interfering routing trees.

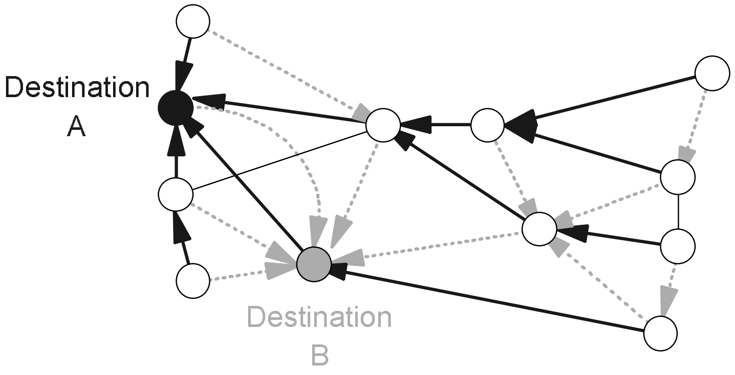


Figure 3.6: Multiple non-interfering trees

An interesting problem to consider is that of selecting clock periods for band repetitions, or the *cycle*. Let *U* indicate the shortest clock unit used in band allocation. Let *P* denote the periodicity of the clock measured in *U* units. For example, let *U* = 1 µs and *P* = 125 *U*; that is, after 125 µs the clock returns to 0. Time may then be indicated in terms of counters similar to seconds, minutes, hours, etc. For example, the time <12, 3>, with the above *U* and *P*, means 3 periods 125 µs long plus 12 µs.

Traffic may also vary in terms of typical frame sizes. Consider the choices of *U* and *P* above over a 2.4 Gb/s link. During a period of *P* = 125 µs, some 300 Kb can be transmitted. If the link is equally shared among 3 to 6 trees, this means that each tree can be allocated an average of 50 to 100 Kb. Additionally, since link speeds may vary greatly, Isochronets may wish to use different periodicity over links. For example, a link of 155 Mb/s may use a period of 16 *P* = 2 ms. Arrivals over this link will be buffered and delivered to higher speed links. Discussion of this general case, however, is beyond the scope of this work.

The band periodicity may vary with the type of traffic served. Low duty traffic such as file transfers may use periods of long duration, whereas interactive voice or video traffic may use much shorter periods.

For the reminder of this work, the physical clock period is fixed at 125 µs. This value is also the sampling rate for voice communications. Bands are allocated as portions of the physical clock.

3.2.4 Protocol Support and Interconnection

All stack layers above the MAC layer are delegated to interfaces at the network periphery. A typical stack organization for Isochronets is depicted in Figure 3.7.

Interconnection of Isochronets can be accomplished via MAC layer bridges using extensions of current well-understood technologies. Conversions need only handle physical layer interfaces and MAC. Above the MAC layer, interconnection becomes transparent. Contrast this with the problem of internetworking two distinct HSNSAs via higher-layer gateways.

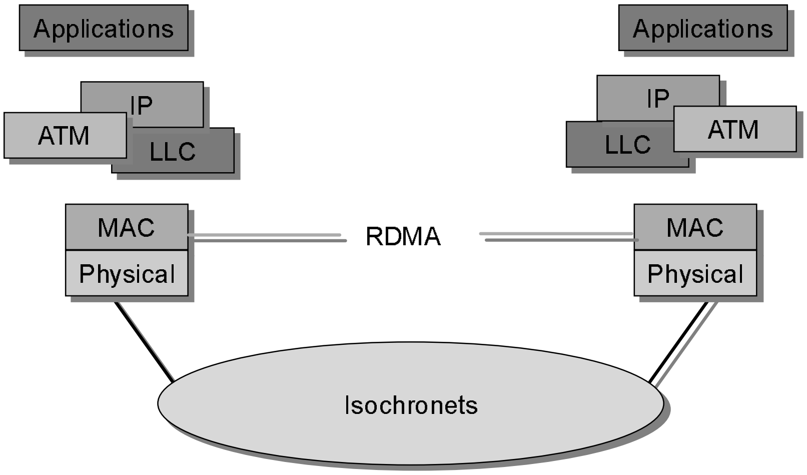


Figure 3.7: Multiple protocol stacks in Isochronets

3.3 Isochronets for HSNs

The main characteristics of Isochronets with respect to the features discussed in Chapter 2 are:

1. *Support end-to-end transport of multi-protocol frames.* Isochronets do not rely on nor need to adapt to any frame structure because frame headers do not need to be parsed inside the network to be switched. Multiple frame structures (e.g., IP, ATM, etc.) can coexist in the network without adaptation.
2. *Support tunable guaranteed QoS of end-to-end transport.* Through band alloca-

tion, sources can be given priority in a tree. When transmitting in this mode, traffic has guaranteed bandwidth and experiences no jitter, loss, nor random delays introduced by contention with other sources. On the other hand, contention bands do not provide any QoS guarantee. By sizing the many kinds of bands at each tree node, an array of services can be tuned. Consequently, QoS is strictly guaranteed through tunable bands.

3. *Support asynchronous, synchronous and isochronous traffic streams.* Synchronization signals are forwarded to network interfaces when bands begin and can be used to synchronize traffic motions. Nevertheless, within a band, motion is asynchronous. Depending on the kind of service a band provides (priority or contention), the end-to-end motion may attain every synchronization need.
4. *Minimize processing and queueing latency in the transport path.* Switching is accomplished by re-configuring nodes over time. Frame headers are not processed to be switched. As a consequence, nodes do not need to process frames at incoming high-speed rates. Rather, nodes need only re-configure themselves periodically. Additionally, queueing only happens to access outgoing links and can be finely controlled, and even avoided completely, by assigning priorities to sources in a band.
5. *Support speed-scalability for link bandwidth ranging from hundreds of megabits to terabits per second.* On the one hand, transmission speeds are independent of control speeds. Control is responsible for node configuration, synchronization, and band allocation. These mechanisms can be implemented completely off-line with respect to transmissions. On the other hand, switching only depends on link speeds for frame detection and contention resolution. These are minimal functions for any switching technique that allows some degree of frame multiplexing. It will be shown in Chapter 6 that these mechanisms can be implemented very efficiently and special techniques can be used to make the dependency on link speeds virtually

non-existent.

6. *Support efficient internetworking with existing stacks and media.* The fact that no frame adaptation is necessary together with the fact that Isochronets may operate on any media (electronic or optical) because it does not process frames make interconnection with existing protocol stacks simple.
7. *Support scalability for large size networks (ranging from ten to hundred million end nodes).* Isochronets offer a degree of scalability between PS and CS. Bandwidth is dedicated (as opposed to available on-demand as in PS), but to routing trees (instead of circuits as in CS), thus enabling a large degree of multiplexing and consequently supporting scalability in number of nodes.
8. *Support efficient interfacing with end nodes.* Band allocation is the single mechanism used to finely tune traffic demands in the network. This simplifies network interface. The interface must receive the synchronization signals from the network indicating the beginning of bands and forward them to upper layer protocol stacks. It also must be able to perform the usual reception and forwarding of frames.
9. *Support efficient bandwidth sharing and flexible adaptation to service demands.* Bandwidth sharing can be controlled through the band allocation mechanism. The band of a tree may be presented in smaller sections to the sources and can thus be used to control the level of contention in the network. On one extreme, all the band is available to all sources, maximizing network resource use. On the other extreme, the band is partitioned in non-overlapping intervals at the sources, thus maximizing QoS. In between, an array of performance behaviors can be carefully tuned.

3.4 RDMA between CS and PS

This section positions RDMA in the context of PS and CS.

If the band associated with a routing tree consists of priority-bands only, that tree is operated in an optimized CS mode. That is, each source is allocated a circuit (priority band) to the tree root. The form of CS supported by Isochronets is superior to traditional CS as circuits only get priority over band usage but do not own it. In a situation where the entire band has been allocated to priority bands serving real-time isochronous traffic, non-real-time data traffic may take advantage of underutilized parts of the band.

Consider now an Isochronet operating in RDMA++ contention resolution mode. If the entire band is allocated to contention traffic, frames moving down the tree will be stored and forwarded as in an ordinary PS network. The form of PS supported by Isochronets is advantageous to traditional one in a few ways. First, Isochronets support virtual cut-through mechanisms as frames arriving to a free switch will continue without store-and-forward delays. Second, no headers are processed in Isochronet switches. Third, buffered frames are aggregated into larger units and transmitted at once, improving the efficiency of buffer retrieval. Fourth, contention happens only among frames to the same destination (and not among non-correlated traffic).

Isochronets, it may be argued, could potentially under-perform PS networks due to the time-division of bandwidth among routes. In situations where significant traffic bursts are randomly generated at multiple routes while other routes are empty, the bandwidth committed to unused routes will be underutilized while the routes serving a burst may have insufficient bandwidth to handle it. A PS network would have permitted the traffic burst to move into the network and utilize its entire band without pre-allocation. Typically, however, admission-control policies will prevent large bursts from entering the network. Such mechanisms as leaky-bucket [Sidi et al. 89] reduce the effective bandwidth available to any given source. A PS network governed by admission policies that limit source bandwidth, presents no advantage over an Isochronet that limits the bandwidth to sources through pre-allocation to routes.

In summary, at the two extremes, Isochronets compare favorably with CS or PS networks. In-between, Isochronets can be configured to span a spectrum of switching techniques of superior performance characteristics to both.

3.5 Summary

Isochronets substantially differ from other HSN architectures. Isochronets virtually eliminate the network layer, reducing it to the MAC layer. As a result: (1) no frame processing (for routing, switching, etc.) is required in the network; (2) there is no need for adaptation layer between different protocol stacks at network interfaces; (3) internetworking is reduced to MAC layer bridging; (4) the network can adapt to the frame sizes and arrival statistics of sources.

In Isochronets, network control functions are entirely separated from transmission activities, which render them capable of: (1) transmission-speed elasticity—transmission speeds can be arbitrarily faster than control speeds; (2) distance elasticity—the network can extend over local, metropolitan, and wide areas; (3) bandwidth-heterogeneity—the network can incorporate links of different transmission speeds; (4) media independence—electronic or all-optical implementations.

Chapter 4

Protocols for Isochronet Operations

4.1 Introduction

4.1.1 The Problem

This chapter defines the main protocols necessary for Isochronet operations. The protocols deal with three main issues: (1) how to allocate trees in the networks, (2) how to assign bands to trees, and (3) how to synchronize bands at different nodes.

Tree allocation. In this problem, the objective is to allocate routing trees in the network so as to satisfy some optimization criteria. The exact criterion is dependent on the particular goals that the network must accomplish. For example, trees can be allocated to minimize the distance between sources and destinations. This strategy is particularly useful when links are operating at high speeds because the propagation delay is a major component in the total end-to-end delay. Another example particular to Isochronets is to allocate trees such that the number of bands necessary to cover all possible destinations is minimized. This goal can be accomplished by allocating as many non-interfering trees as possible in the same band. The tree allocation mechanism should be geared toward choosing as many trees as possible that do not share links in the same direction.

Band synchronization. Given the band allocations in each node, the objective is to synchronize the bands, that is, assure that frames transmitted from a given node during band B_i reach the next node in the tree within the same band B_i. Synchronization mismatches in Isochronets may result in frames being misrouted. For example, if a frame transmitted during band B_i reaches an intermediate node during another band B_j, frames are forwarded from that point on toward B_j's destination. Another concern in band synchronization is to avoid bands intercepting each other. For example, if the intersection between bands B_i and B_j at some node is not empty, routing during the overlap period is ambiguous because incoming frames could potentially follow two separate paths.

Band allocation. Given a set of band transmission requirements, the goal is to allocate bands that will fit the requirements while being synchronized and not interfering with each other. The band allocation is thus concerned in finding two parameters for each band: initiation time within the clock cycle and size. The allocation changes over time given new demands computed from past history.

These problems (and their generalizations) could per se become a very intricate study. They are not the main subject of this thesis and more elaborate solutions are left for future work. This chapter gives a preliminary solution to each problem.

4.1.2 Main Results

Tree allocation. The solution proposed is an exhaustive search. This solution is acceptable in this case because the tree allocation protocol is executed only when the network configuration changes.

Band synchronization. The proposed solution operates by manipulating the link propagation delays in the network. It involves a special device placed at each switch outgoing port, that is, at the start of each network link. The device can be set to increase the link delay. A distributed protocol computes each link delay and then increases it using the special device to make the final delay a multiple of the cycle period. Because of this manipulation, if a frame is sent at time *t* within the cycle from a given node, it reaches the next node in the path still at time *t* but many cycles later. This fact trivializes synchronization because all that is necessary in this scenario is to start and end each band exactly at the same moment within each cycle.

Band allocation. For the purpose of this protocol, bands are always synchronized using the band synchronization protocol. The main concern of the band allocation protocol is then to size the bands. The task is pursued using a dynamic algorithm based on the Least Recently Used (LRU) memory allocation policy. The protocol operates by shrinking band sizes for destinations that are not accessed frequently and then increasing band sizes of destinations that need higher demands.

All of the discussed protocols can be implemented off-line, that is, they may execute at slow speed relative to transmissions.

4.1.3 Chapter Organization

This chapter is organized as follows. Section 4.2 discusses the mathematical model that will be used to formally define each problem. The tree allocation problem, the band synchronization problem, and the band allocation problem are discussed in Sections 4.3, 4.4, and 4.5, respectively. Section 4.6 overviews future work in each of these problems. Finally, Section 4.7 concludes.

4.2 The Model

This section described the mathematical model used to formulate the problems in the next sections. A network is viewed as a directed graph [Behzad et al. 79] $G = \langle V_G, E_G \rangle$, where V_G is the set of nodes and E_G is the set of edges. Each edge e has positive real-valued capacity $c(e)$ and propagation delay $d(e)$. Edges are also denoted by an ordered pair $e = (e_s, e_d)$ where e_s is the source and e_d is the destination of the edge. The following property must hold in G: $\forall u, v \in V_G \cdot (u, v) \in E_G \Rightarrow (v, u) \in E_G$. That is, only edges in both directions connect pairs of nodes.

A spanning tree $T = \langle V_T, E_T \rangle$ is a connected acyclic subgraph of G where $V_T = V_G$, $E_T \subseteq E_G$. Of particular interest are spanning trees that have a distinguished node r that can be reached from all other nodes. Such tree is a *routing tree* and node r is the root or destination of the tree. The tree with root r is labeled T_r.

Another class of trees of interest is spanning trees with a distinguished source node s that has a path in the tree to all other nodes. These are the *multicast trees*.

A clock is associated with each node in the network. The clock cycle period ranges from 0 to C. A band for a set of disjoint trees Γ at node n is an interval $B_n(\Gamma) = [b_n(\Gamma), e_n(\Gamma)]$, where $0 \le b_n(\Gamma) \le e_n(\Gamma) \le C$.

4.3 Tree Allocation

The main concern of the tree allocation problem is to build spanning trees in the network satisfying some optimization criteria. As discussed in Section 4.1, one such criteria would be to minimize the distance between nodes. This is a useful approach especially in HSNs where the propagation delay plays a major role in the end-to-end delay. Minimizing the number of links or, alternatively, the physical distance between nodes by choosing minimal distance paths will significantly decrease the total end-to-end delay. For example, when transmissions occur at 2.4 Gb/s, an ATM cell takes only 177 ns to be transmitted. The direct United States cross-country propagation delay is about 30 ms. It is easy to see that, if tree allocations are such that cross-country communication crosses a few intermediate nodes possibly with some nodes not geographically located in the right direction, the end-to-end total delay may be increased by a few milliseconds. It is thus important to minimize the number of hops or the physical distance in this scenario. Trees can be allocated with respect to this optimization criterion using one of the standard techniques to find minimal distance routes in PS networks [Comer 91, Halsall 92].

This section exploits an interesting novel minimization criterion particular to Isochronets. According to this criterion, the total number of bands to cover all possible destinations is the variable to be minimized. The path to accomplish this minimization criterion is to fit as many trees as possible in the same band. Two trees can inhabit the same band as long as their paths do not interfere, that is, as long as they do not share a link in the same direction. Figure 3.6 (Chapter 3) illustrates an example of two such trees.

The formal definition of the tree allocation problem is given in Problem 4.1.

Problem 4.1: Tree allocation.

Given: A network G.

Find: A set Θ of $|V_G|$ routing trees.

Satisfying: $\forall n \in V_G \cdot \exists T_n \in \Theta$.

Minimizing: $r(\Theta)$, where $r(\Theta)$ is the minimal number of subsets of Θ such that two elements of Θ are in the same subset if and only if they do not share any link in the same direction and the union of all the subsets is equal to Θ.

Problem 4.1 states that, given a generic network, the goal is to find a routing tree for each node (all nodes are possible destinations). The set of routing trees found must satisfy the following minimization criterion. It must be possible to subdivide the set in subsets of disjoint trees, that is, trees that do not share any link in the same direction. Furthermore, the number of such subsets must be minimal and the union of the subsets must be equal to the original set of routing trees. Notice that each subset may be assigned to the same band. Consequently, by minimizing the number of such subsets, the number of bands is minimized.

This section proposes a solution that uses an exhaustive search algorithm. It is an acceptable solution in this case because the problem can be solved off-line, before the system starts operating. It is assumed that links or nodes fail rarely. In such cases, the tree allocations need to be re-computed. Another alternative is to pre-compute tree allocations for all possible configurations of node or link failures and store them in a table for future look-ups.

Another compelling argument for exhaustive search algorithms is that Isochronets many times operate on networks with few nodes. If the network size is large, a cluster technique outlined in Chapter 8 is used to build Isochronets within and among clusters. The goal of the clustering technique is to create a hierarchical solution such that the number of clusters and the nodes within clusters is kept small.

Protocol 4.1: *Generates an optimal solution for Problem 4.1 using an exhaustive search approach. Given a network G, the protocol finds the minimal number of bands that will cover all destinations.*

Generate all routing trees in *G*, building the set *R* of all possible routing trees.

1. Generate all maximal subsets of *R* such that two trees in *R* are in the same subset if and only if they do not share any link in the same direction. The resulting set of maximal subsets of *R* is *S*.
2. Select a minimum number of elements in *S* such that the union of all selected elements has one routing tree per network node.
3. If the elements of *S* selected in Step 3 have more than one tree for any destination, eliminate redundant trees so that the final solution contains only one tree per destination.
4. Distributed the allocation among the nodes in the network.

The exhaustive search algorithm works as sketched in Protocol 4.1. The algorithm is executed at a central site where the allocation is computed based on the network topology. Later, the allocation is distributed among the nodes in the network. Another option for reliability is to replicate the algorithm in a few sites that can run it in tandem.

Step 1 generates all possible routing tree in *G*. Step 2 seeks to generate all possible sets of routing trees that do not interfere, that is, that do not share a link in the same direction. The generated sets are maximal, that is, no routing tree can be added to the set without violating the non interference criteria. Notice that each maximal set contains trees that can be allocated in the same band. Step 3 finds the minimal number of maximal sets whose union contains a routing tree for each node in the network. In other words, Step 3 effectively finds the minimal number of bands that can cover all possible destinations. Step 4 removes redundant trees. The solution directly satisfies the minimization criteria in Problem 4.1.

Algorithm 4.1: *Generates all possible routing trees of a network G. The number of nodes in G is n.*

1. Generate all possible sets of *n*-1 links in *G*. The resulting set is *L*.
2. Generate a new set *L'* from *L* such that *L'* is a subset of *L* and all elements of *L'* constitute a valid routing tree. *L'* contains all possible routing trees of *G*.

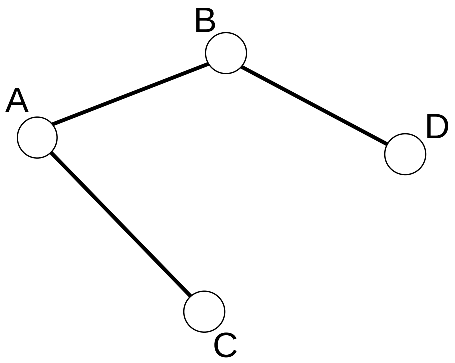


Figure 4.1: Example network topology

The implementation of Protocol 4.1 should not present difficulties, except for Step 1 that is solved in Algorithm 4.1. It operates by generating all possible sets of *n*-1 links of *G* and then filtering the ones that do not constitute a valid routing tree.

Example 4.1: *Allocates routing trees in the example four node network depicted in Figure 4.1. The routing trees allocated minimize the number of necessary bands to cover all possible bands in the network.*

Consider the network *G* in Figure 4.1. Let us begin by generating all possible routing trees according to Algorithm 4.1. Notice that the links are (A,B), (B,A), (A,C), (C,A), (B,D), (D,B). Step 1 generates $\binom{6}{3} = 20$ sets of 3 links in *G*. For example, {(C,A), (A,B), (B,D)} is a valid tree while {(A,B), (A,C), (B,D)} is not. The following are the valid routing trees after the filtering phase in Step 2:

- {(A,B), (C,A), (B,D)};
- {(A,B), (C,A), (D,B)};
- {(B,A), (A,C), (D,B)}; and
- {(B,A), (C,A), (D,B)}.

Protocol 4.1 generates in Step 2 the following maximal subsets:

- {{(A,B), (C,A), (B,D)}, {(B,A), (A,C), (D,B)}};
- {{(A,B), (C,A), (D,B)}}; and
- {{(B,A), (C,A), (D,B)}}

Usually many possibilities will exist in Step 3, unlike the situation in this example in which the network contains only one path from each source to each destination. The solution for the example is to assign one band per maximal set, resulting in tree bands.

Example 4.1 illustrates how these protocols can be used to allocate routing trees in a four node network.

There are a few directions that can be used to generalize Protocol 4.1. The first one is to allocate not only routing trees, but also multicast trees according to demands. The most complete scenario would be to allocate both a routing and a multicast tree per node. Toward this goal, Step 1 in Protocol 4.1 should generate all possible routing and multicast trees. Step 2 in Algorithm 4.1 can be tuned to generate all multicast trees. Step 2 in Protocol 4.1 needs no modification. Steps 3 and 4 need to be extended to cover all necessary routing and multicast trees.

Another interesting generalization of Protocol 4.1 is to fit trees with similar loads in the same band. That is, two trees are placed in the same band if their loads are similar. Otherwise, some of the trees would not fully use the band. Allocating bands taking into account the load of the trees would minimize bandwidth wasted due to load imbalances among trees sharing the same band. This problem is left for future investigation.

The complexity of the protocols discussed increases rapidly with the number of bidirectional links in the network. Algorithm 4.1 in Step 1 generates $\binom{2l}{n-1}$ possibilities, where $n-1 \le l \le n^2$ is the number of links in the network. Step 2 in Protocol 4.1 is in the worst case of size $\binom{2l}{n}^{\binom{2l}{n}}$. The other steps can be executed within this complexity.

As pointed out previously, Isochronets are designed to operate on networks with few nodes that are then interconnected to other networks. This fact and the fact that the tree allocation algorithm is run once before the network begins operating make the performance of this solution acceptable.

4.4 Synchronization

There are two kinds of synchronization necessary for Isochronets: clock synchronization and band synchronization. Any of the traditional protocols such as the Network Time Protocol (NTP) [Mills 91] may be used to solve the clock synchronization problem. This section defines and provides a solution for the band synchronization problem.

Band synchronization must ascertain the following constraints:

- ***Band Constraint.*** For each path from a source to a given destination, the corresponding band at each node must be strictly contained, when propagation delay is added, within the same band at the next node.
- ***Overlap Constraint.*** The bands of different trees on the same node do not intersect.

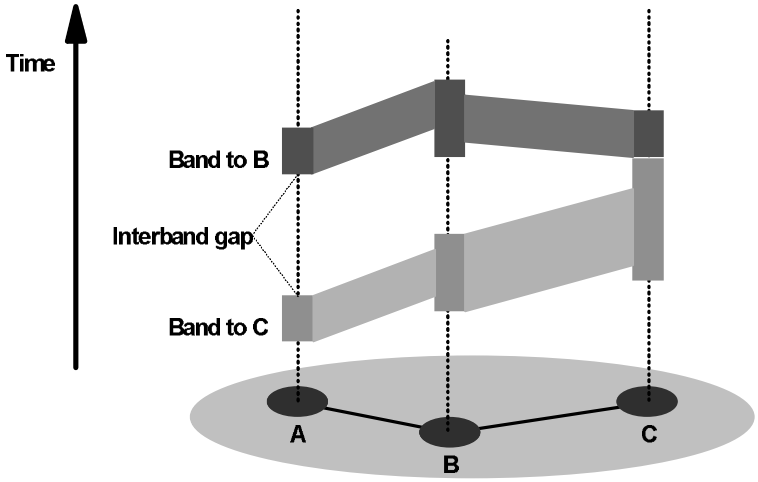


Figure 4.2: Band and overlap constraints

The goal of band synchronization is to establish band initiation values within the clock cycle that satisfy both the band constraint and the overlap constraint at each node. Figure 4.2 illustrates both constraints. The graph above illustrates the band allocation (boxes) at each node in the network underneath. The Band Constraint is illustrated by the fact that each band is contained, when propagation delay is added, within the same band in the next node toward the destination. The Overlap Constraint means that boxes cannot intersect. Notice that this constraint forced the inter-band gap at nodes A and B.

The network links are assumed to contain special hardware devices or *delay element* that can be tuned to increase the link delay by a certain positive value between zero and the cycle period. The delay element can be tuned by the source switching node incident to the link. It can be implemented as a buffer at the node output, as will be explained in Chapter 6.

Formally, the propagation delay in each link is an element of a group [Suzuki 77]. The domain of the group is the set of all real number s (or *shifts*) in the interval $0 \le s \le C$, where C is the clock cycle size at each link. The group contains the operation sum modulo C that is denoted by "$\cdot$". The reason why the propagation delay can be abstracted using this group is that the value of interest for the synchronization problem is at what point *within the cycle* a transmitted frame reaches the destination rather than the total propagation delay. Given an edge e, the shift on e is denoted by s_e. The "$\cdot$" operation on bands is defined by $[b_n(\Gamma), e_n(\Gamma)] \cdot [s_{(n,m)}, s_{(n,m)}] = [b_n(\Gamma) \cdot s_{(n,m)}, e_n(\Gamma) \cdot s_{(n,m)}]$. The formal definition of the synchronization problem is the following.

Problem 4.2 states that, given a network, a collection Φ that contains sets of trees that participate in the same band (that is, trees that do not interfere), the goal is to find:

1. For each node in the network, and for each set of non-interfering trees, the initiation and termination times of the band.
2. Delays in each link in the network that participates in some tree.

Problem 4.2: *Band synchronization.*

Given: A network G and a collection Φ of sets Γ of non-interfering spanning trees of G.

Find: For each node $n \in V_G$ and each $\Gamma \in \Phi$, a band $B_n(\Gamma)$.

For each edge e that occurs in elements of Φ, a shift s_e.

Satisfying: For each $\Gamma \in \Phi$ and each directed edge (n,m) that occurs in elements of Γ, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cdot \left[s_{(n,m)}, s_{(n,m)}\right] \subseteq \left[b_m(\Gamma), e_m(\Gamma)\right]$.

For each node $n \in V_G$, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cap \left[b_n(\Gamma'), e_n(\Gamma')\right] = \varnothing$, if $\Gamma \neq \Gamma'$.

Minimizing:
$$L = \sum_{\Gamma \in \Phi, (n,m) \in E_T, T \in \Gamma} \sum \left[\left(e_m(\Gamma) - e_n(\Gamma) \cdot s_{(n,m)}\right) + \left(b_n(\Gamma) \cdot s_{(n,m)} - b_m(\Gamma)\right)\right] + \sum_{n \in V_T} [C - \sum_{\Gamma \in \Phi} (e_n(\Gamma) - b_n(\Gamma))] .$$

Solutions are restricted so as to satisfy the band and overlap constraints. The minimization criterion is to avoid wasting bandwidth. It has two parts. The first computes how much bandwidth is wasted due to band shifting by link propagation delays. It is computed as the summation of the differences between the ending times and the difference between the starting times of the same band at the two nodes of each link. Ideally, there should be no such waste if the bands are synchronized. That is, each band should be allocated such that, when a given band at the start of a link is shifted by the delay in the link, it exactly matches the same band at the end of the link.

The second component adds the difference at each node between the cycle period and the sum of all bands. It computes how much of the cycle is not allocated to a band. The ideal allocation would make such waste 0.

This formula is a measure of bandwidth waste in terms of band offsets at the source and destination of each link and gaps between bands. The offset is illustrated in Figure 4.2. For example, the band to C at A and B are of different sizes and thus introduce some wasted bandwidth. The inter-gap between bands is also illustrated in the figure. The gap is sometimes necessary in order to satisfy the overlap constraint.

The following optimal solution is proposed: add delays to the links so as to make the resulting link delays equal to 0 modulo C and make each band initiation and termination times the same in all the nodes. It is easy to verify that, in this case, $L = 0$. Whenever a new band is allocated (see next section), the beginning and ending times for each band are set to the same value for all the nodes in the network.

The only open question is how to make the final delay 0 modulo C for all links in the network. Protocol 4.2 ensures this property. The idea is to use the group property of existence of an inverse element for each link shift. The inverse element is added to the link delay, making the total link delay become 0 modulo C.

In Step 1, node A sends a request to B to find out the delay in the link between them. In Step 2, B time-stamps the reply and then transmits it. In Step 3, A receives the reply and marks the arrival time. A is now in position to compute the offset between A and B. The offset is the propagation delay between the nodes modulo the cycle period. Using the computed offset, A then updates its delay element in the link between A and B in Step 4.

If the delay element was set with a value larger than the offset computed, the latter is simply subtracted from the contents of the delay element making the total end-to-end delay 0 modulo the cycle period. Otherwise, the contents of the delay elements are incremented with the value necessary to make the total delay between A and B a multiple of the cycle period.

Protocol 4.2: *Sets the delay at each link to 0. Given two nodes A and B, the protocol sets the delay in the link (A,B) to 0 modulo C. The delay introduced by the delay element at the output of A to B is d*(A,*B*).

1. A->B: Request For Delay (RFD) message for link (A,B).
2. B->A: Delay Response (DR); B marks time *T* at which DR is sent.
3. A marks arrival time *R* of DR. A measures the offset *O*=(*R*-*T*) modulo *C*.
4. If *d*(A,B) > *O*, set *d*(A,B) to *d*(A,B)-*O*. Otherwise, set *d*(A,B) to *d*(A,B)+*C*-*O*.

Example 4.2 illustrates some issues in Protocol 4.2.

Example 4.2: *Computes the incremental delay that needs to be added to the delay element in an example link (A,B). The propagation delay between the links is 10.732 ms and the cycle period is 125* μ*s.*

Any frame sent by A at time *t* will reach B at time *t*+107 μs (modulo the cycle period) 85 cycles later. The idea in Protocol 4.2 is to add 18 μs to the propagation delay between A and B so that now any frame sent by A at time *t* will reach B at time *t*, but 86 cycles later.

4.5 Band Allocation

The goal of band allocation protocols is to establish appropriate band duration given minimal required tree loads. The loads are expressed in terms of band sizes. The allocation must satisfy the band and the overlap constraints. Band allocation is formally defined as follows.

Problem 4: *Band allocation.*

Given: A network G, a collection Φ of sets Γ of non-interfering spanning trees of G, and a tree load δ_Γ for each $\Gamma \in \Phi$.

Find: For each node $n \in V_G$ and each $\Gamma \in \Phi$, a band $B_n(\Gamma)$.

For each edge e that occurs in elements of Φ, a shift s_e.

Satisfying: For each $\Gamma \in \Phi$ and each directed edge (n,m) that occurs in elements of Γ, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cdot \left[s_{(n,m)}, s_{(n,m)}\right] \subseteq \left[b_m(\Gamma), e_m(\Gamma)\right]$.

For each node $n \in V_G$, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cap \left[b_n(\Gamma'), e_n(\Gamma')\right] = \varnothing$, if $\Gamma \neq \Gamma'$.

For each node $n \in V_G$ and each $\Gamma \in \Phi$, $e_n(\Gamma) - b_n(\Gamma) \geq \delta_\Gamma$.

Minimizing:

$$L = \sum_{\Gamma \in \Phi, (n,m) \in E_T, T \in \Gamma} \sum \left[\left(e_m(\Gamma) - e_n(\Gamma) \cdot s_{(n,m)}\right) + \left(b_n(\Gamma) \cdot s_{(n,m)} - b_m(\Gamma)\right)\right] + \sum_{n \in V_T} [C - \sum_{\Gamma \in \Phi} (e_n(\Gamma) - b_n(\Gamma))] .$$

All the nodes must know what is the position of each band within the cycle because all the trees are spanning. Thus, in order to allocate bands, it is necessary to communicate the allocation to all the nodes in the network. Notice that the problem definition is similar to the synchronization problem, except that now band sizes must be at least as big as the required tree load.

The band allocation problem can be solved in a manner similar to band synchronization. By setting the link delays to 0 modulo the cycle period and the band initiation and termination values to be the same at each node, $L = 0$. To complete band allocation, it is necessary to set what the band initiation and termination times should be. The value can be computed by allocating sizes that can fit minimal requested loads.

Problem 4.3 can be generalized to dynamic load demands as follows. The minimal band sizes now changed periodically based on source load demands. The synchronization solution can still be adopted in this case. Nevertheless, the process of sizing the bands becomes dynamic rather than static. A possible solution to the dynamic band sizing problem is to allocate bands according to a memory management algorithm such as Least Recently Used (LRU). Protocol 4.3 is an example of such an approach.

Protocol 4.3: *Adapts the band sizes according to the LRU memory management algorithm. Given b bands, each of size s*[*i*]*, 1 ≤ i ≤ b, and a cycle of size C, the protocol adapts the band sizes according to the traffic demands d*[*i*]*, 1 ≤ i ≤ b, where d*[*i*] *is the highest demand of all trees sharing the band. The resulting new band sizes are s'*[*i*].

1. Find all bands *i* such that *d*[*i*] < *s*[*i*]. Make *s'*[*i*] = *d*[*i*] for those bands and compute *LOB,* the LeftOver Bandwidth (that is, summation of *s*[*i*] - *d*[*i*] for all bands such that *d*[*i*] < *s*[*i*]).
2. Find all bands *i* such that *d*[*i*] > *s*[*i*]. Compute *NB*, the Needed Bandwidth (that is, summation of *d*[*i*] - *s*[*i*] for all bands such that *d*[*i*] > *s*[*i*]).
3. Distribute *LOB* among the bands *i* such that *d*[*i*] > *s*[*i*]. Make *s'*[*i*] = *s*[*i*] + *LOB**(*d*[*i*] - *s*[*i*])/*NB*.
4. Distribute the band allocation among all nodes in the network.

Step 1 searches all bands such that the traffic demand is smaller than the allocated band size. These are shrunk into the size requested and the left over band is computed to be later distributed among the more demanding destinations. Step 2 searches for the bands that have more demand than the allocated bandwidth. The total needed band is computed. Step 3 distributes the left over band computed in Step 1 among the needing destinations according to their demands.

Protocol 4.4 can be extended to include filters, as in traditional LRU algorithms, to avoid oscillations in band allocations.

4.6 Future Work

As stated in the introduction, a complete study of these problems can be quite complex and is left for future endeavors. The goal of this section is to state directions that can be pursued.

4.6.1 Tree Allocation

The algorithm for tree allocation is an exhaustive search solution. Potential work in the direction of improving this initial solution would greatly improve the performance of the protocol presented. Simple optimization could be pursued depending on how the network will be used. For example, if only routing trees are needed, Algorithm 4.1 can generate trees more efficiently by not trying to include links originating at nodes already in the tree.

Additionally, novel approximate solutions using heuristics could be of great potential value. For example, trying to generate only trees with minimal degree at each node will tend to use fewer links per node and thus potentially leave links for other non-interfering trees.

4.6.2 Synchronization

One interesting extension is to study synchronization in the absence of delay elements. For example, in an all-optical implementation it may be difficult to implement delay elements that can be finely tuned. Another example is a multiprocessor system connected by Isochronets. In this case, the end-to-end delay in the network is a critical parameter and may not be increased artificially. It would be interesting to find synchronization solutions for such systems.

4.6.3 Band Allocation

Probably the richest opportunity for future work in Isochronet mechanisms is to find new techniques to solve the band allocation problem. The solution presented in this chapter is but one possible dynamic band sizing protocol. Many others need to be exploited and analyzed to compare their performance values.

Another rich domain is the study of priority band distribution among nodes in the network. This distribution has the potential of trading resource use for QoS demands.

4.7 Summary

The mechanisms necessary for Isochronet operations are the solutions to three problems: (1) tree allocation, (2) band synchronization, and (3) band allocation. Each of these can lead to intricate studies. This chapter defines formally each of the problems and gives initial solutions to them.

The tree allocation problem is concerned with allocation routing and multicast trees in the network. It is solved using an exhaustive search approach. This initial solution is acceptable because the allocation only happens upon network reconfiguration.

The band synchronization problem is to allocate bands so that frames transmitted from a node within a particular band reaches the next node in the path within the same band. The synchronization must also avoid overlapping different bands at a node. The proposed solution to this problem is to increase or decrease the link delays so as to make the final delay a multiple of the cycle period. By using this technique, frames transmitted at time *t* within a cycle reach each node in its path at the same time *t* some cycles later. Finally, one can simply allocate exactly the same bands in all nodes and they will be automatically synchronized.

The band allocation problem is concerned with sizing bands so as to accommodate the loads in each tree. A possible technique to solve this problem is to use an LRU approach similar to the one used in memory allocation policies. In such solution, bands from idle trees are allocated to busy trees in order to balance traffic demands.

All of these solutions are but a first cut through the problems. Future research has ample opportunities in each of these domains, especially in the band allocation problem.

Chapter 5

Protocols for Loosely-synchronous Stacks

5.1 Introduction

5.1.1 The Problem

The problem addressed in this chapter is how to extend existing protocol stacks to support real-time traffic, that is, synchronous or isochronous traffic. Traditional stacks are built to handle only asynchronous applications. New real-time applications, such as multimedia exchanges, demand synchronous or isochronous services from the protocol stack. Ideally, the new services must be provided as extensions of current protocols to maintain compatibility with existing systems.

5.1.2 Main Results

This chapter conveys two main contributions. Firstly, it describes a novel *Loosely-synchronous Transfer Mode (LTM)* that provides synchronization signals in order to support real-time services. Secondly, it explains how any protocol stack can be extended with synchronization signals from the LTM and become a *Synchronous Protocol Stack (SPS)*.

LTM. LTM is a novel transfer mode that informs periphery nodes about current network status, that is, destinations that are reachable and associated QoS. The information is supplied as *loosely-synchronous* signals. Upon reception of such signals, periphery nodes can best schedule their traffic streams according to current reachable destinations and necessary QoS.

The period between loosely-synchronous signals is a *band*. That is, each band represents reachable network destinations with associated QoS in terms of the end-to-end delay, jitter, and loss experienced in the communication.

LTM has many advantages when compared with other transmission modes:

- Sources and destinations are synchronized with the network through loosely-synchronous signals. That is, sources are synchronized with network bands (status) and not with frames or slots within frames as in Synchronous Transfer Mode (STM) [Halsall 92]. Bands last much longer than slots or frames. Consequently, the synchronization does not need to be as accurate as in STM but still can attain the same sort of synchronous service.
- QoS can be controlled and guaranteed because the loosely-synchronous signals coordinate how sources and network must interact. The signals convey information about current provided QoS that can be used by sources to best schedule their traffic needs. Guaranteed performance is essential for time-sensitive applications such as voice communications.
- Multiple protocol frames can be transferred without fragmentation and reassembly because there is no pre-defined frame structure in the communication. This simpli-

fies extensions of existing protocol stacks with the loosely-synchronous signals and interconnection of multiple stacks.

SPS. The SPS is any stack extended with the loosely synchronous signals provided by the LTM. An SPS forwards LTM synchronization signals through its layers up to the applications. In the SPS:

- Existing protocol stacks are supported unchanged. The new services can be implemented as orthogonal additions to the existing ones.
- Existing stacks are extended with novel source and destination synchronization. The synchronization can be used to implement novel real-time services at multiple stack layers, up to the applications.

5.1.3 Chapter Organization

The reminder of this chapter is organized as follows. Sections 5.2 and 5.3 introduce LTM and SPS respectively. Section 5.4 defines the LTM MAC services. Section 5.5 discusses how current transport protocols can be implemented and extended using LTM services. Section 0 shows an example of transport protocol services. Sections 5.7 and 5.8 discuss related and future work respectively. Finally, Section 5.9 summarizes.

5.2 Loosely-synchronous Transfer Mode (LTM)

Emerging Broadband Integrated Service Digital Networks (B-ISDNs) will have to integrate traffic requiring a broad range of guaranteed Quality of Services (QoS). The network transfer mode must be able to provide guarantees on delay, jitter, and loss to address the needs of data, voice, or video applications. Additionally, certain applications may require synchronization of remote activities and transfers. For example, synchronization is required among remote real-time computations or applications that use the network as a massively parallel computing resource. Current transfer technologies, the Synchronous Transfer Mode (STM) [Halsall 92] and the Asynchronous Transfer Mode (ATM) [De Prycker 93, Le Boudec 92], are limited in providing full coverage of these requirements. For example, ATM networks do not support guaranteed synchronization and offer a limited form of QoS guarantees. STM requires very tight synchronization that may be difficult to accomplish when transmissions occur at high speeds.

The primary function of a transfer mode is to move frames in a network. Toward this end, it needs to: (1) send and receive frames and (2) resolve contention for network resources such as intermediate links and buffers. In the context of HSNs, the transfer needs also to (3) provide QoS in the network.

In order to guarantee QoS, nodes must know how and when resources they use will be allocated to them. If this is not the case, the way in which the system resolves contention cannot be predicted and thus allocations cannot be guaranteed. For example, to deliver a frame every 33 ms in a video exchange, intermediate links in the path must be dedicated to transfer data frames from that connection every 33 ms.

There are two basic manners in which sources may be informed about resource allocation in the network. The first is the periphery-centered information. A periphery node requests resources and remembers how the allocation is structured. For example, in ATM resources are allocated beforehand. Based on the requests, the system decides how resource allocation is mediated among competing sources. Periphery nodes can derive how their resources will be allocated based on their requests and on partial information about the resource allocation policy. If a node requests 1 Mb/s end-to-end connection, all it needs to do is remember to send information at this rate. Periphery-centered information works best when the network guarantees the allocations. Otherwise, requests do not match allocations and thus the information at the nodes is an assumption as opposed to a precise schedule on how resources are allocated.

The second method is network-centered information. Periphery nodes may still request resources, but the network notifies them when they become available, that is, nodes acquire status information from the network. This is the method used in the LTM. The network status embody information on the destinations that are reachable and on the associated QoS. It is periodically distributed to the nodes when the network it changes. Furthermore, sometimes resources do not need to be requested. For example, if only a route without QoS is necessary, sources can wait until the network status indicates that the route is available.

The network conveys its status through loosely-synchronous signals. The name reflects the fact that the signals are not necessarily synchronized with clocks, but with network status. Nevertheless, they still provide means to synchronize traffic motions. For example, a source may synchronize its generation of QoS-demanding traffic to a particular destination to the corresponding network signal.

Loosely-synchronous signals are issued by internal network switches and then forwarded to upper layers of the protocol stacks. Switches decide to issue these signals based on network status changes. At periphery nodes, signals are captured by the network interface and forwarded through the periphery protocol stack, up to the application layer.

The period of time between loosely-synchronous signals is a band. Bands embody two global network status: (1) a connection to destinations in the network, and (2) a certain QoS associated to the band. Bands are repeated periodically in a cycle.

RDMA in Isochronets is an example of an LTM. The loosely-synchronous signals indicate the beginning of bands. The priority, contention, and multicast bands in Isochronets can exist in any LTM and this will be the assumption for the reminder of this chapter.

In reality, STM may be seen as an extreme example of LTM. STM is network-centered because it signals peripheries when frames and slots within frames begin. Nevertheless, STM does not exploit the full potential in LTM because it supports only dedicated resource allocation.

LTM can adapt to traffic characteristics and mimic advantageous characteristics of both STM and ATM. Similarly to STM, QoS can be guaranteed. That is, through the allocation of priority to destinations in the network, LTM can deliver the requested QoS. For example, end-to-end delay can be bound by the time waiting for the priority band (which in turn is bound by the cycle duration) plus the transmission and propagation delay in the network. Additionally, since sources get priority and not exclusive use of resources, bandwidth utilization is improved in LTM when compared to STM.

Furthermore, nodes at network periphery do not need to synchronize their clocks globally, as in STM. Necessary synchronization information is given by the network and nodes need to synchronize only locally with the network. The necessary accuracy is much lower when compared with STM because nodes need to know only what is the current band (which usually lasts for a long time). For example, an 8 bit slot in a STM frame at 2.4 Gb/s transmission rate lasts 3.3 ns. Typical bands last between a few hundreds of nanoseconds up to a few scores of microseconds because the network status (that is, routes and associated QoS provided by the network) is designed to change at this rate.

Similarly to ATM, diverse traffic classes may be serviced by LTM. Transfers of frames from a given class occur during periods in which the network is offering the most appropriate characteristics for the service requirements. For example, video traffic must be sent during periods when the network provides priority in the correct source and destination connection. Data traffic may be sent during any period in which the network enables the correct destination.

LTM may achieve accurate traffic synchronization and potentially avoid buffering in the network. In ATM, such buffering is necessary to compensate synchronization mismatches due to network resource multiplexing. Since in LTM global network information is known, sources may tune traffic generation in order to minimize contention buffering in the network.

In the context of this work, LTM is to be used as the transfer technique in the backbone network. Signaling information is supplied by the network switches to the periphery nodes. The latter can use the signals to implement their protocol stacks.

5.3 The Synchronous Protocol Stack

Traditional protocol stacks were designed to support asynchronous (or non real-time) services. Two main events may trigger execution in traditional stacks: message arrival from upper or lower layers and time-out signals. Time-out signals are usually generated by local clocks that may be out of synchronization with other clocks in the network while message arrivals are asynchronous in nature. It can thus be safely concluded that only asynchronous events fuel protocol motions in traditional stacks.

Synchronous and isochronous applications are difficult to build on traditional stacks. These applications have strict timing requirements that cannot be captured by asynchronous events. For example, a frame must be delivered every 33.3 ms in a video exchange. Such application requires dedicated network and stack resource for end-to-end transport without delays or jitter every 33 ms.

The challenge is to devise elementary functionality that could be added to a protocol stack to support real-time services. One possibility is to propagate loosely-synchronous signals through the stack. Such signals embody status information that can be used to synchronize transmissions at each layer in the stack. Such enhanced stack is an SPS.

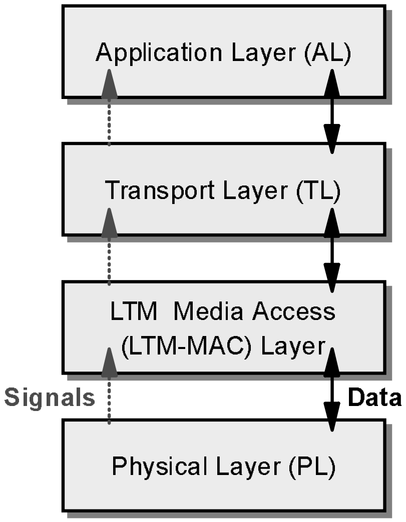


Figure 5.1. Synchronous Protocol Stack (SPS) structure

The SPS is the stack at peripheral nodes attached to a backbone network that uses LTM. In addition to the normal data flow between stack layers, SPS implements a bottom-up flow of synchronization signals from the underlying LTM network. These signals can be used at any layer, including the application, to implement synchronization functions. The general structure of an SPS is depicted in Figure 5.1.

The Physical Layer (PL) does not need to be bound to any special technology (i.e., electronic or photonic implementations may be employed) because LTM does not rely on any particular frame structure nor processes frame headers. The data link and network layers are collapsed into the LTM MAC (LTM-MAC) layer that uses LTM as the transfer mechanism. The Transport Layer (TL) is responsible for the allocation and control of network resources, such as bands with necessary QoS. The Application Layer (AL) interacts with the TL requesting necessary classes of services.

Any traditional TL protocol currently used over STM or ATM may be used over the LTM-MAC directly with no changes other than adapting to the LTM-MAC SAPs (as explained later in Section 5.5.3). In addition, the signaling information from LTM-MAC can be used to enhance the functionality provided to provide real-time services (as explained later in Sections 5.5.4 and 5.5.5).

A feature unique to SPS is that synchronization signals (and not only data) may be reflected all the way to the AL. The network may thus inform its current status directly to applications that may then schedule its transmissions. For example, video traffic may be scheduled to generate frames when the proper band begins in each cycle.

In a more generic setting, SPS may be a portion of the overall protocol stack, as depicted in Figure 5.2. Stack layers need to be able to operate in real-time to handle signaling from the MAC-LTM. The protocol stack is divided in two portions: a lower real-time protocol stack that can handle signals from LTM, and an upper non-real-time protocol stack that does not have real-time service provision. The *SPS boundary* is the interface between both stacks. The interface is responsible for buffering requests to overcome operational lack of synchronization between both portions. Synchronization of operations with transmissions can be guaranteed only below the SPS boundary. The figure illustrates a typical data path from protocol stack A to B, using the signaling from the LTM network. Notice that signaling is not passed above the SPS limit.

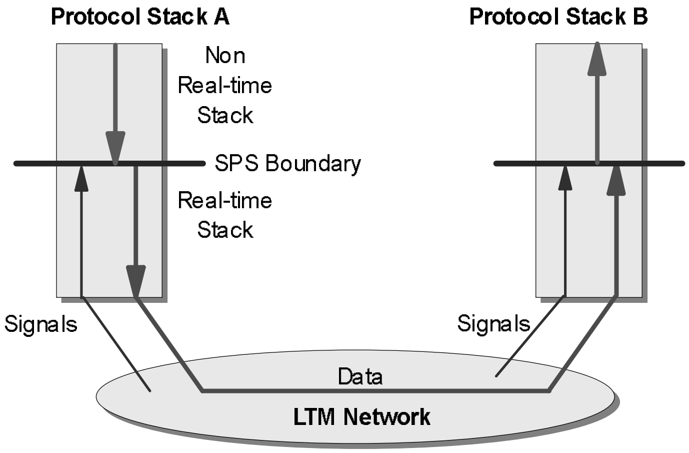


Figure 5.2. Handling signals in the protocol stack

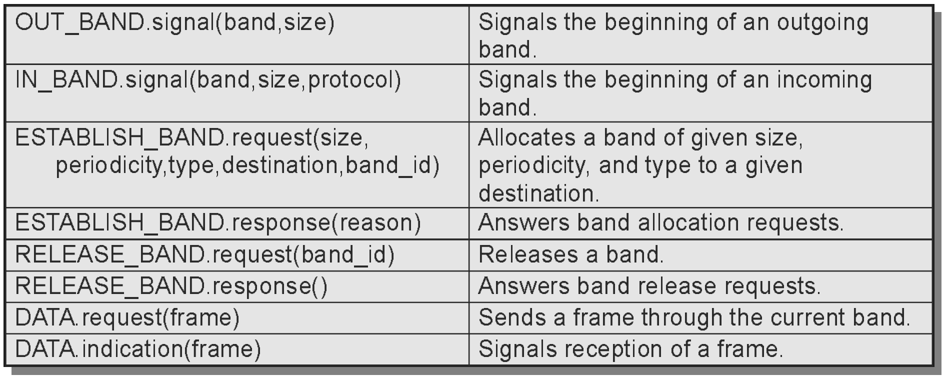
The extreme cases of this scenario are two. In the first, the SPS boundary is above the application layer, and thus LTM signals can be relayed up to applications. This can be the case when machines and operating systems provide real-time support, that is, when it is possible to predict upper bounds on execution times of operating system calls. In the second, the SPS boundary is at the interface with the LTM network and no signaling from the

LTM network is forwarded to upper protocol layers. This could be the case, for example, when traditional protocols are to be implemented on conventional machines without real-time support.

5.4 The LTM-MAC Service Access Points

This section summarizes the LTM-MAC SAPs. The LTM layer may be implemented using any technology as long as the LTM-MAC SAPs are kept unchanged. All protocol layers above the LTM-MAC can operate independently of the specific mechanisms used to implement the LTM.

The LTM-MAC SAPs are summarized in Figure 5.3. The first service is *OUT_BAND.signal(band, size)*. It is a signal from LTM to mark the beginning of an out-going band. The *band* parameter has the form *<band_id, type>* where *band_id* is an identifier for the band and *type* is one of the QoS associated with the band (contention, priority, or multicast). For example, *<5, c>* means that the band identifier is *5* and its type is contention. Other possible types are priority (p) and multicast (m). The *size* parameter is the length of the corresponding band in nanoseconds.



OUT_BAND.signal(band,size)	Signals the beginning of an outgoing band.
IN_BAND.signal(band,size,protocol)	Signals the beginning of an incoming band.
ESTABLISH_BAND.request(size, periodicity,type,destination,band_id)	Allocates a band of given size, periodicity, and type to a given destination.
ESTABLISH_BAND.response(reason)	Answers band allocation requests.
RELEASE_BAND.request(band_id)	Releases a band.
RELEASE_BAND.response()	Answers band release requests.
DATA.request(frame)	Sends a frame through the current band.
DATA.indication(frame)	Signals reception of a frame.

Figure 5.3. LTM-MAC SAPs

Similarly, *IN_BAND.signal(band, size, protocol)* signals the beginning of an incoming band. Additionally, since the MAC may multiplex multiple transport entities above it, the *protocol* parameter identifies the protocol that should service the band.

The next services are used to establish a band. *ESTABLISH_BAND.request(size, periodicity, type, destination, band_id)* requests the establishment of a band of a given *size, periodicity,* and *type* to the respective destination or destinations (in the case of multicast bands). The periodicity is the amount of time between occurrences of the same band. If it is 0, the band is allocated only once in a cycle. The band identifier for the connection is returned in the *band_id* field. The *destination* field denotes not only the destination machine address, but also the destination protocol used in the *IN_BAND.signal* SAP at the destination.

Notice that the naming structure for sources and destinations adopted so far is appropriate for the LTM-MAC layer because it deals with traffic that is already multiplexed. Upper layers of the protocol stacks may need do adopt new levels of naming to demultiplex the traffic streams. One example is to use ports at the transport level. The upper layer names are multiplexed (or de-multiplexed) to (or from) the LTM-MAC according to their protocol frame structures.

The manner in which the LTM-MAC is going to achieve band allocation is dependent on its internal operations. For example, contention bands can be allocated as a portion of the LTM supplied contention band to the given destination. To allocate priority or multicast bands, the signaling of the underneath transfer mode must be used to negotiate the allocation and inform all nodes involved. Priority bands can be allocated as a portion of the respective LTM supplied contention band. *ESTABLISH_BAND.response(reason)* indicates if the band was established or not (and in the latter case the *reason* for failure). Band requests may fail because not enough resources are available to allocate the requested QoS.

SAP *RELEASE_BAND.request(band_id)* is used to release previously allocated bands. *RELEASE_BAND.response()* is used by LTM to signal when the request is finished.

Finally, the last services are used to send and receive frames through LTM. *DATA.request(frame)* sends the user supplied *frame* through the network. *DATA.indication(frame)* is used by LTM to signal the arrival of a *frame*.

5.5 Transport Protocols

This section discusses how current transport protocols (in the IP stack, ATM stack, etc.) can use LTM-MAC SAPs directly to implement their functionality. Additionally, the signaling features enabled by LTM are used to show how such protocols can be extended to implement asynchronous, synchronous, and isochronous services.

5.5.1 Mapping Destination Addresses onto LTM-MAC Bands

Transport protocol addresses need to be mapped into appropriate band identifiers at the network periphery to enable transmissions through the LTM-MAC. Translation tables are used to this effect. The fields in a translation table are destination address and band identifier. For each reachable destination address, the corresponding band identifier (that is, the identifier for the band that routes to that destination) is given.

One interesting issue is how to set such translation tables initially. That is, TL entities need to be able to find band identifiers connected to desired destination addresses. A variation of the ARP protocol [Comer 91] stack is used to implement this function. The protocol works as follows. A special frame containing the address of the requesting TL entity is transmitted during a band to request the TL address of the associated destinations. The peer TL entities recognize the special frame and reply with their TL addresses, using the band to the requesting entity. If the band identifier of the requesting TL entity is not already known by a replying TL entity, its reply is sent during all bands in the cycle to cover all possible requesting entities. Notice that by sending such request frames during all bands in the cycle, all destinations are covered and the table in the requesting TL entity is accordingly initialized.

The address translation mechanism described may be implemented more efficiently (in terms of bandwidth use, that is, avoiding broadcasts of requests) by using a special name server accessible through a special band identifier. The server keeps the current mapping and answers requests for address resolution.

5.5.2 Mapping QoS onto LTM-MAC Bands

Transport protocol QoS also needs to be mapped into appropriate LTM-MAC band identifiers. Translation tables can be used to this effect as well. The fields in a translation table are band identifier and the associated QoS delivered by the band. This information can be acquired by broadcasting special request frames during all bands in a cycle, as explained in Section 5.5.1.

5.5.3 LTM Supports Asynchronous Traffic

Asynchronous traffic requires no time constraints or loss guarantees on frame delivery. Examples of applications that generate such traffic are electronic mail delivery, file transfers, etc. Since these applications do not have hard timing constraints, frame loss may be overcome by re-transmission. Asynchronous traffic can be directly supported on top of contention bands.

For example, to implement a file transfer application, a contention band can be used to transfer each portion of the file. When errors occur, they are detected at the destination TL entity that requests re-transmission using the reverse band to the source.

The Internet Protocol. The Internet [Comer 91] transport protocols, User Datagram Protocol (UDP) and Transmission Control Protocol (TCP), are examples of asynchronous communication protocols. One way to support them on LTM, is to first implement the Internet Protocol (IP) layer on LTM. UDP and TCP both use the IP layer to implement their services.

To implement IP, a contention band is established to each destination through the *ESTABLISH_BAND.request* SAP. Packets received from upper layer entities are buffered in the IP layer according to their band identifiers (which are computed from frame destination addresses). When the beginning of an outgoing band is signaled to the IP layer, it forwards the respective buffered frames. Similarly, when the beginning of an incoming band to an IP entity is signaled, the entity receives the frames and forwards them to upper layer entities.

5.5.4 LTM Supports Isochronous Traffic

In *isochronous traffic*, frames must be played-back (that is, used) with minimal jitter between them (that is, frame access should happen at constant intervals) and some loss may be tolerated.

Such services can be implemented on SPS by making the TL compile requests into two parameters available through *ESTABLISH_BAND.request*: priority band size and periodicity (that is, how many cycles apart should the band be allocated). The mapping is performed according to the QoS requested, that is, depending on the requested jitter and bandwidth. The priority band periodicity is determined by the jitter requirements and buffering capacity at the destination. The priority band size is then computed from the requested bandwidth, link capacity, and computed periodicity.

For example in a video transmission, if the cycle period is 125 μs, allocation can be implemented as follows. A priority band sized to fit one video frame can be allocated every 264 cycles (or 33 ms apart which is the sampling rate for video). Alternatively, a smaller priority band can be allocated for portions of the video frame with higher frequency, depending on the amount of buffers available at the destination. As long buffering space for 1 frame is available, the allocation can be done such that every 264 cycles 1 complete frame is delivered. A typical video transmission in this scenario is depicted in Figure 5.4. When an application receives a signal from the LTM-MAC, it is awaken and it transmits a video frame (or portion of a video frame). After that, the next video frame (or portion) is generated by the application that then goes to sleep waiting for the next signal. The signals from the network pace the application to generate isochronous frames.

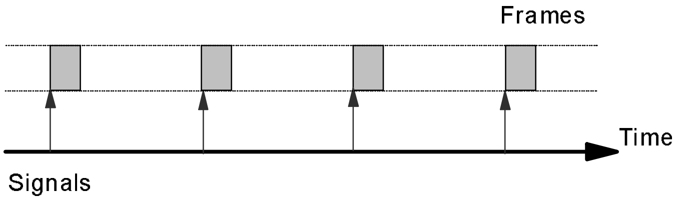


Figure 5.4. Isochronous transmissions

Another possibility is to profit from the fact that some loss may be tolerated in this kind of communication. The TL may then allocate two kinds of contention bands: one for asynchronous traffic and another for isochronous traffic. The contention band for isochronous services can be used according to distributed protocols that allocate resources by maximizing multiplexing constrained by the tolerable loss allowed by applications, as it is done in the context of ATM networks [De Prycker 93, Le Boudec 92]. That is, portions of the contention band for isochronous traffic are allocated not to guarantee loss-less communication, but to deliver low probability of loss. In this manner, the portion of the band to be allocated is smaller than what would be necessary to guarantee no loss. When sending frames, the TL always gives priority to isochronous traffic over asynchronous traffic.

After the band allocation phase is completed, the TL receives signaling information from LTM-MAC when corresponding bands begin. It then schedules signaling to applications when the corresponding priority bands are due. When signaled, applications may send data to TL that uses LTM-MAC to transmit them. Potentially, traffic generation may be scheduled to begin only when signaling is received from TL, thus minimizing buffering.

The ATM Adaptation Layer. The ATM Adaptation Layer (AAL) [De Prycker 93, Le Boudec 92] protocols can be implemented using two alternative *ESTAB-*

LISH_BAND.request options: priority or contention band. The ATM virtual path and virtual channel identifiers are translated into band identifiers. ATM services not requiring QoS are implemented on top of contention bands, in similarity to the IP protocol stack. QoS demanding services must be implemented on priority bands. The following overviews how each AAL protocol can be implemented.

The AAL 1 is intended to service constant bit rate applications such as uncompressed video transmissions. AAL 1 services can be directly implemented using a priority band with periodicity equal to the necessary sampling rate. If the sampling rate is too small and each sample contains more information than what can be allocated in one band, the sampling rate may be increased and the band size decreased. For example, to accommodate 100 Mbits/s video transmissions, a band of size 3.3 Mbits can be allocated every 33 ms or a band of size 100 Kbits can be allocated every 1 ms.

The AAL 2 is intended for variable bit services. Such services can be accomplished in several ways, depending on the error rate to be allowed in the communication. One possibility is to allocate two contention bands, one for normal contention traffic, and another to service variable bit rate services, as explained previously. Another possibility is to guarantee error-free delivery by allocating a priority band.

The AAL 3/4 and 5 are intended for data communications sensitive to loss, but not to delay. This is the ideal application for a contention band, as explained for IP.

Notice that all frame structures of the various AAL protocols can be sent directly to LTM-MAC, without adaptation. This is because LTM does not rely on any particular frame structure to perform its operations.

5.5.5 LTM Supports Synchronous Traffic

In *synchronous traffic*, it is necessary to guarantee maximum end-to-end delay (that is, the delay to the destination may fluctuate, but must be bound by a pre-negotiated value) and error-free communication. This kind of traffic is supported by allocating a priority band.

For example, a virtual high-speed multiprocessor machine can be implemented using a set of machines interconnected by a network such as Isochronets. This application requires sporadic exchange of small amounts of data for inter-process communication. The transfer, nevertheless, needs to be reliable (error-free) and done in a timely fashion due to the high-speed of the processors. Priority bands can be pre-allocated for this sporadic communication in every cycle. The bandwidth size is computed from the maximum bandwidth required between processors.

Most observations from Section 5.5.4 in the context of scheduling isochronous traffic generation according to the synchronization signals from LTM are applicable for synchronous traffic as well.

Synchronous IP. An important feature in SPS is that the signals that are input from the LTM-MAC can be used to extend existing TL protocols toward providing synchronous services. For example, the IP suite can be extended with new SAPs to the application layer to support synchronous transport. Such SAPs can be implemented using priority bands at the LTM layer.

5.5.6 Compiling Higher Level QoS Parameters

Higher level QoS parameters such as end-to-end delay, loss, and jitter need to be compiled into the elements made available by the LTM-MAC layer, that is, type of band, band size, and band periodicity. Such compilation is performed by TL protocols, depending on the high-level QoS parameters they offer. This section presents an example of how the translations can be performed.

In the example Protocol 5.1, two parameters are used by the application layer to request transport layer services: maximum end-to-end delay and bandwidth needed (Step 1). The goal is to accommodate the request into a priority band. The variable *P* in Step 2 is used to always allocate only a portion of the LTM supplied band, to avoid compromising the whole band with one request, if possible. The allocation begins backwards in Steps 3 and 4 searching for idle portions in the cycles from the deadline (*T*+*D*) up to the current time (*T*). In Step 5, the allocation is tested. If it was successful, that is, the first allocated priority band begins at least at time *T*+*O* (where *O* is the overhead necessary before the first frame can be sent), the application is informed about the allocation. If not, new allocations are tried with a new value for *P*. If, after all values for *P* have been tried, no feasible allocation exists, the failure is communicated to the application.

Two observations are important in the example described. First, optimizations can be performed, but were not adopted for simplicity. For example, *O* could be estimated to avoid the situation in which the allocation succeeds, but the feasibility test fails. Second, Protocol 5.1 is only one possibility for mapping end-to-end delay and bandwidth requests into priority bands. Each transport layer protocol may have its one translation algorithm, most suitable for the services it intends to provide.

Protocol 5.1: *Translates higher level QoS parameters (end-to-end delay and bandwidth) into an LTM-MAC priority band.*

1. Let the requested delay be *D*, and the requested bandwidth be *B*.
2. Assign 50% to *P*.
3. Mark location *T*+*D* (where *T* is the current time) in the time line.
4. Search each cycle backwards beginning from *T*+*D* and fill at most *P* percent of the idle portion of the priority band assigned for the source and destination pair. The search is performed by requesting the LTM-MAC to allocate a portion (of the given size) of the band to the destination. The search begins with the cycle in which the band ends before and closest to *T*+*D*. It ends with the cycle in which the band begins after and closest to *T*.
5. Let *E* be the instant in the time line when the first found portion of the band begins. If $E \geq T+O$ (where *O* is the overhead until the first transmission can happen), the allocation is feasible. Stop and inform the application. If $E < T+O$ or if *E* could not be found, the allocation is not feasible. Go to Step 6.
6. If *P* is less than 100%, add 10% to *P* and go to Step 3. If *P* is 100%, stop. The requested service cannot be delivered. Stop and inform the application.

Notice that the implementation of requests for QoS in terms of jitter and bandwidth can be accomplished using Protocol 5.1 by substituting the required maximum jitter for the maximum delay. For the jitter case, care must be taken to request a periodic allocation, instead of a single allocation (where the period is computed according to maximum jitter demands).

5.6 Examples of Transport Layer SAP

This section illustrates how SPS concepts can extend traditional TL SAPs to implement new real-time services.

Example 1 illustrates how the socket [Stevens 90] SAP can be extended to provide novel real-time services. The piece of code on top uses the extended socket SAP to forward frames in the network. While the condition is not true, it sleeps on the socket waiting for a loosely-synchronous signal. The *socket_sleep(s_skt, wake_up_time)* operation makes the process sleep on socket *s_skt* until *wake_up_time* nanoseconds before the incoming band signal. When the process is awaked, it proceeds to forward the frame through the socket and goes back to generate the next frame. Then, it sleeps waiting for the next signal.

The *socket_sleep* operation is implemented using the LTM-MAC SAPs. It loops forever to receive incoming band signals. When it receives one, it checks to see if it is the correct band. If it is, it waits the period to the next signal (the band size) minus the requested wake-up time.

Notice that the *wait_out_band()* operation can be implemented in terms of the *OUT_BAND.signal(band, size)* LTM-MAC SAP.

There could be other implementations of the same operations.

Example 4.1: *Shows how the socket SAP can be extended to provide real-time services using loosely-synchronous signals.*

```
...
While (!condition) {
    ...   /* generate next frame */
    socket_sleep (s_skt, wake_up_time);
    send (s_skt, frame); }
...
socket_sleep (synch_socket s_skt, time wake_up) {
    while (true) {
        response = wait_out_band ();
        if (check (response, s_skt.name)) {
            sleep (s_skt.next - wake_up);
            return ; }}}
```

5.7 Related Work

5.7.1 LTM between ATM and STM

This section compares LTM with STM and ATM as solutions for B-ISBN.

Plain STM generates a periodic fixed-size frame. The frame is divided in fixed-sized slots (usually of size 1 byte) that can be used by sources to transmit information. Once allocated, bandwidth is guaranteed for the connection, thus delivering strict QoS in terms of guaranteed end-to-end delay, and no jitter or loss. It is necessary, nevertheless, to keep a virtual global clock in order to synchronize all nodes in the network to the global frame and slots within the frame.

The main problems in adopting STM for B-ISDN are the lack of flexibility in the slot size and in supporting on-demand service allocation. Applications such as voice communications require small slots (usually 8 bits per frame), while video communication would best profit from large slots. If the slot size is defined too big, network bandwidth may be wasted while if it is too small, it may be difficult to allocate broadband services. Additionally, STM lacks provision for asynchronous traffic (i.e., on-demand PS). When such traffic needs to access the network, slots must be allocated in the whole path from source to destination with unacceptable end-to-end allocation delays.

Flexibility in bandwidth allocation is the main force pushing ATM as a solution for B-ISDN. In ATM, information is partitioned into fixed-size cells that are sent asynchronously to the destination. Destinations are recognized by using identifiers in the cells (as opposed to being identified by the location in a frame as is the case in STM) and, as a consequence, no global clock synchronization is required. Nonetheless, virtual connection (channel or path) establishment is necessary to allocate identifiers. Bandwidth can be flexibly allocated based on source demands.

The main problems of ATM are limited support for asynchronous or synchronous communication and the trade-off between guaranteed QoS and efficient network utilization. The main drawback of asynchronous communications over ATM is that they need to be preceded by the virtual connection establishment phase, which involves end-to-end delays. The connection establishment phase in many applications may take longer than the transfer phase, which makes asynchronous communications inefficient both in end-to-end delays and in resource utilization. Some work [Gerla et al. 92] has been done to overcome this problem by allocating permanent virtual channels for the purpose of sending asynchronous traffic, but these solutions may require complex management of virtual connection identifiers for all source/destination possibilities and may poorly use network resources. Synchronous communications are difficult due to the lack of synchronization at the ATM layer. Synchronization is relayed to upper layer protocols. But, the accuracy of such protocols is bound by the maximum end-to-end jitter, which may be difficult to predict exactly in ATM.

The QoS parameters are negotiated during connection establishment. Nevertheless, if all network resources are to be allocated to guarantee QoS, ATM will poorly use network resources, similarly to what happens in STM. For example, video coding algorithms usually generate variable bit rate outputs. To guarantee no loss during a video section, ATM would need to allocate resources for peak bit rate. But, the peak to mean bit rate ratio is usually high, which means that network utilization may become poor. Due to this problem, resource allocation in ATM networks is usually performed based on a lower QoS than the one requested. The idea is that multiplexing several connections and granting lower QoS to each may deliver high QoS for the multiplexed ensemble while accomplishing high network resource utilization. Unfortunately, it is not clear what is the actual QoS delivered to a particular connection under this regimen. Usually such QoS can only be characterized using a probability distribution, with a low (but existent) chance of severe service degradation for some connections.

LTM merges the flexibility in bandwidth allocation of ATM with the support for guaranteed QoS communications found in STM. As opposed to ATM or STM, where network resources have to adapt to traffic characteristics, in LTM traffic can adapt to network operations. That is, the network is in charge of informing sources about its current status so that sources may adapt its traffic generation accordingly. The unit of transfer in LTM is not pre-set to a fixed structure or size. QoS in terms of maximum delay, jitter, or loss may be guaranteed.

LTM enables scheduling of traffic generation to its signals and thus can minimize network buffering due to synchronization errors. Synchronous services may be achieved by local control exchange between the network and periphery nodes, without incurring end-to-end delays as it is necessary in traditional protocol stacks.

5.7.2 The SPS and Real-time Protocols

This section positions SPS with respect to proposed protocols that support real-time communications such as the Real Time Protocol (RTP) [Schulzrinne et al. 94], the Internet Stream Protocol version II (ST-II) [Topolic 90], and the XTPX Transport Protocol [Metzler et al. 92].

The real-time portions of these protocols try to accomplish two main functions:

- Request QoS to the underlying transport protocols; and
- Monitor and control the QoS delivered.

The first function is responsible to map application demands into concrete QoS parameters such as bandwidth necessary and supported loss rate in the communication. The second manages QoS to detect periods when the transport fails to deliver the requested QoS. Usually, frame headers include time stamps and sequence numbers to detect QoS violations.

Any real-time protocol can be placed in the SPS with the added advantage that its functionality can be significantly reduced. That is, the management portion can be reduced significantly because LTM effectively guarantees the QoS in the communication.

5.8 Future Work

The next challenge is to design and incorporate the extensions suggested in this chapter into an existing protocol stack to build an SPS and then evaluate its performance and functionality. One possibility is to extend the IP stack on RDMA using the implemented Isochronet switch (see Chapter 6).

5.9 Summary

This chapter introduced a novel transfer mode: LTM. LTM operates by signaling periphery nodes when destinations become available with specific QoS guarantees. It encompasses advantages of both STM and ATM. Similarly to STM, synchronous communications with guaranteed QoS can be supported directly on LTM. Bandwidth allocation flexibility, one great advantage of ATM, can be found in LTM as well. Nevertheless, many of the problems introduced by STM and ATM are overcome by LTM: (1) no frame structure is necessary for communication; (2) traffic adapts to network status (instead of the other way around); (3) buffering in the network may be significantly lowered by correlating traffic generation to network status; (4) strict QoS is supported directly; and (5) synchronization signals are provided to the protocol stacks at periphery nodes.

The SPS is a novel protocol stack that uses LTM as its MAC mechanism. Because synchronization signals flow in SPS from LTM upwards to the application, SPS may incorporate protocols to support asynchronous, synchronous, and isochronous communications. Traditional transport layer protocols may be supported unchanged in SPS. Additionally, such protocols can be extended to offer real-time and multicast services.

Chapter 6

Isochronet Switch Design and Implementation

6.1 Introduction

6.1.1 The Problem

The problem addressed in this chapter is to design and to implement an electronic Isochronet switch (Isoswitch) prototype. The switch must operate at giga bit per second rates and use only simple off-the-shelf components.

6.1.2 Main Results

Electronic switch design and implementation. An electronic Isoswitch has been designed and implemented. It has four input and four output ports each operating at 1 Gb/s. The design addressed the following challenges:

- Scalability with respect to number of channels and link speeds;
- Use of simple off-the-shelf components;
- Ease of integration with existing hardware;
- Configuration overheads do not disturb or slow on-going transmissions.

Interface card to a SPARC 1 machine. An interface card between the Isoswitch and a SPARC 1 machine has been built. The card has a nominal throughput of 22 Mb/s. It signals the beginning of bands to the SPARC processor and can thus be used to implement LTM and SPS.

Optical switch design. This chapter also proposes a preliminary design of an all-optical switch implementation. The implementation has a few advantages:

- No conversion of en-route signals to the electronic domain, which makes possible implementations potentially at hundreds of terabits per second.
- Fewer wavelengths are necessary than in pure WDM.
- RDMA- can be implemented guaranteeing no loss for priority traffic.

6.1.3 Chapter Organization

The reminder of this chapter is organized as follows. Section 6.2 discusses the main issues in building high-speed switches. Section 6.3 describes the electronic Isochronet switch design and implementation. Section 6.4 describes the design and implementation of an interface card between the switch and an attached workstation. Section 6.5 overviews the efficiency, complexity, and scalability of the switch. Section 6.6 evaluates the compatibility of the switch with current network devices. Section 6.7 identifies the main novel services the switch offers. Section 6.8 describes a potential all-optical Isochronet switch design. Section 6.9 overview future work. Finally, Section 6.10 summarizes.

6.2 High-speed Switching

Judging by the pace at which current optical transmission link technology is evolving, future HSNs will switch gigabits or even terabits of information per second. Besides being efficient, an adequate switching mechanism for HSNs must maximize bandwidth use while providing the necessary QoS that integrated data, voice, and video applications need.

Switching efficiency. Current switching techniques rely on the fact that processing efficiency significantly prevails over transmission efficiency. Unfortunately, the scenario is likely to change when HSN link speeds reach hundreds of gigabits or even terabits per second because it may become difficult to process frame headers or synchronize sources and destinations at these speeds.

QoS guarantees. HSN switches must be able to support a variety of QoS needs in terms of maximum end-to-end delay, jitter, loss, and bandwidth allocation. These parameters must be tunable by sources.

Scalability in speed. Link speeds have been increasing very rapidly in recent years and are expected to continue the trend with advances in optical transmissions. It is thus advisable not to base the switch design on peculiarity of specific transmission rates.

Scalability in the number of ports. Scalability in the number of ports has always been an issue in switch design. Network systems are continuously growing and switches must be scaled accordingly.

Implementation complexity and cost. The provision of multimedia services to a large market will significantly increase the complexity and size of HSNs. If switches are complex, their maintenance and cost will affect the number of end users supported and cost of services provided. It is thus desirable to employ simple components (including switches) in the network.

Interface complexity and cost. The engineering of the switch must not complicate the engineering of its interface. The simpler the interface, the more services can be directly connected to the switch at cheap cost. Additionally, current services may be easier to integrated if the interface is kept simple.

Compatibility with current network devices. Current network devices will not cease to exist in future integrated HSNs. Slower speed technology will coexist with highs-speed technology, especially during transition phases. The switching technology must be compatible with existing devices to assure wide acceptance.

Resource multiplexing. HSN must be able to multiplex resources when QoS can be traded for cheaper services.

Minimal re-configuration overhead. Switch re-confirmation must be performed without disturbing on-going transmissions.

Isochronets are a strong candidate architecture to accommodate these features. There is no frame processing in the Isoswitch, which has a few advantages:

- The switching function is a simple mapping from incoming links into outgoing links, based on the current enabled trees and contention resolution policy among contending links within the same tree. These functions are realized completely off-line with transmissions. This makes the design efficient in terms of internal delay, scaleable with respect to link speeds, and simple to implement.

- Multiple frame structures can be supported without adaptation because headers are not processed during switching. This makes the Isoswitch easy to integrate with existing network devices and protocols.
- The interconnection fabric between input and output ports is not controlled by any frame header field, and thus any of the current modular interconnection technologies [Adams et al. 87, Broomell and Heath 83, Feng 81] can be used to implement the fabric, controlled by the slow periodic changes in tree configurations. The use of such technologies makes it possible for Isoswitches to scale with the number of ports.

Switch control functions can be accomplished through one simple mechanism: band allocation. Because of this:

- The interface to the switch has to provide two simple functions: transfer of frames to and from the network, and provision of signals to attached nodes informing when new trees become available. This simplifies the interface implementation.
- The necessary configuration information must include only current tree configuration together with priority among contending sources. All this information can be computed off-line (using a general purpose machine) and downloaded into the switch periodically, which then functions based on the new configuration. The configuration overhead is thus reduced to downloading simple configuration tables.

Additionally, the band allocation mechanism can be tuned to satisfy a variety of QoS demands. On one hand, strict guaranteed QoS in terms of delay, jitter, and loss can be provided by priority bands. On the other hand, resource multiplexing can be provided by contention bands. By tuning priority and contention bands, QoS can be compromised for multiplexing in many different ways.

6.3 Isochronet Electronic Switch Design

This section describes the electronic RDMA+ Isoswitch design. Switching in the Isoswitch consists of two functions. The first is to configure the nodes periodically to form respective routing trees. The second is to select one of the contending sources when contention happens. The following sections take a bottom-up view to the switch design.

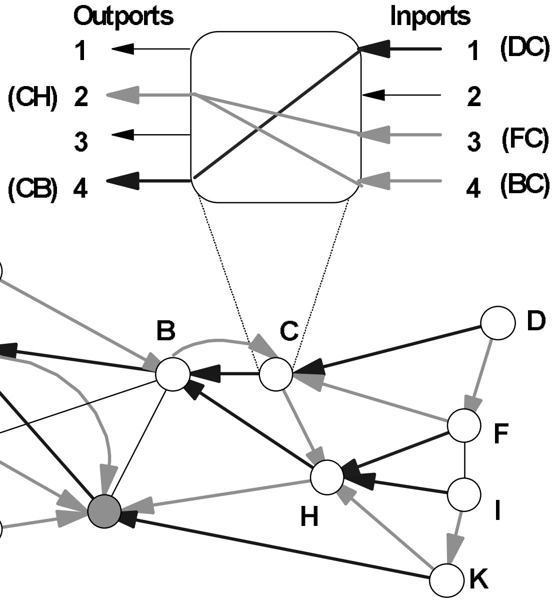


Figure 6.1: Tree allocation and node configuration

6.3.1 Configuration Table

A typical tree configuration is depicted in Figure 6.1. One of the nodes in the tree is shown with its input and output port connections necessary to accomplish the desired tree configuration for two non-interfering routing trees (depicted using different shades of gray). The connections can be described by maintaining a data structure at each output port that lists all the input ports connected to it. In addition, the data structure should identify which of the connected inputs (if any) has priority. This information is captured in the Configuration Tables (CTs) at each node, described in Figure 6.2.

For the reminder of this section, it is assumed that each node has *n* input ports (inports) and *m* output ports (outports). Each inport is identified by a number in the range [1,*n*]. Similarly, outport identifiers are in the range [1,*m*].

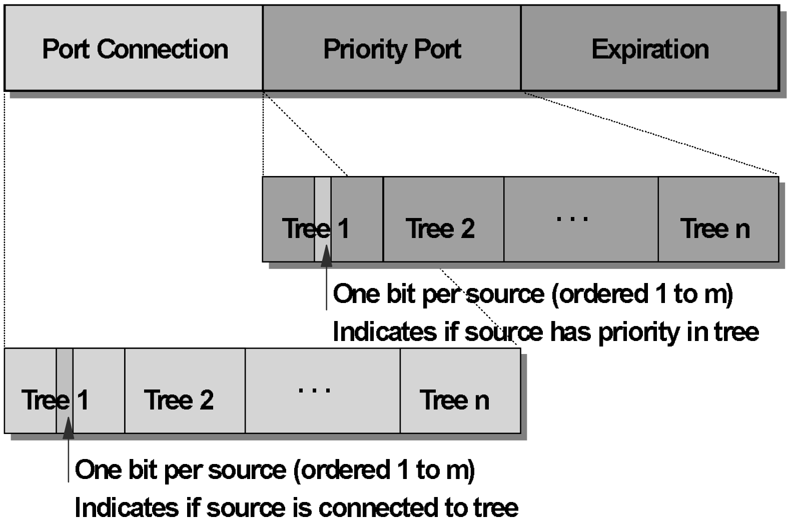

Port Connection
Priority Port
Expiration
Tree 1
Tree 2
. . .
Tree n
One bit per source (ordered 1 to m)
Indicates if source has priority in tree
Tree 1
Tree 2
. . .
Tree n
One bit per source (ordered 1 to m)
Indicates if source is connected to tree

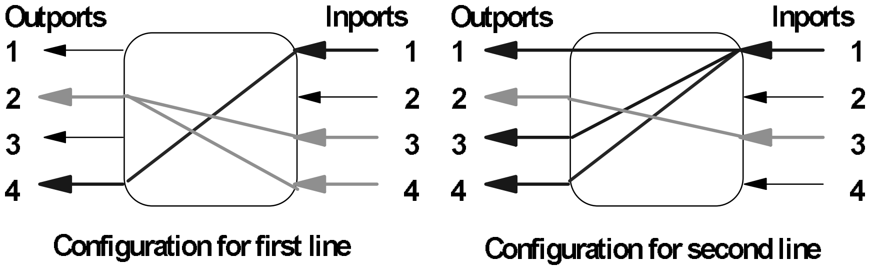
Figure 6.2: Configuration Table structure

The CT contains one line for each possible band in the cycle. Each line contains three fields: Port Connection, Priority Port, and Expiration. The Port Connection field describes the configurations of the trees that transverse the node. It contains a sequence of *m* binary words (one per outport) of size *n* bit (one per inport). Bit i ($1 \leq i \leq n$) in word j ($1 \leq j \leq m$) is 1 if and only if input i is connected to output j.

The priority ports field is again a sequence of *m* *n*-bit words. Bit *i* in word *j* is 1 if and only if *i* has priority to *j* during the current band. Notice that only one source can have priority to a particular output port within a band. This means that only one of the *n* bits can be 1 for each word.

The expiration field is a binary number that indicates how many clock ticks the current band should last.

Figure 6.3 illustrates the CT at a particular network node.



Port Connection	Priority Port	Expiration
…	…	…
0000 0011 0000 1000	0000 0010 0000 0000	100011110001
1000 0010 1000 1000	0000 0000 0000 0000	000111101100
…	…	…

Figure 6.3: Configuration Table in a particular network node

The CT is implemented as a RAM memory.

6.3.2 Arbitration Logic

The CT provides all information that is necessary to decide which input is granted access to the respective output at any given time. The Arbitration Logic (AL) is responsible for reaching such decisions. It is implemented as a combinational circuit.

Algorithm 6.1 is an abstraction of the AL. In Step 1, *Grant* is initialized, assuming that no input will be granted access to any output port. In Step 2, each output is evaluated to decide which input will be connected to it. Step 2.1 checks to see if there is a busy input that has priority to the output. If so, that input is granted access to the output. In Step 2.2, if no input has priority, there are two possibilities. In the first (Steps 2.2.1 and 2.2.2), there are many busy input ports connected to the output port being analyzed. One of them is picked randomly and granted access. In the second, no busy input is connected to the output and consequently the output is kept idle. This algorithm is repeated periodically to resolve input to output connectivity.

Algorithm 6.1: *Let n and m be the number of input and output ports, respectively. Let Con*[*i*,*j*] *be 1 if and only if input i is connected to output j* ($1 \le i \le n$, $1 \le j \le m$)*. Let Pri*[*i*,*j*] *be 1 if and only if input i has priority to output j. Let Busy*[*i*] *be 1 if and only if input i is busy. Let Grant*[*i*,*j*] *be 1 if and only if input i is granted access to output j. This algorithm computes Grant*[*i*,*j*].

```
1.  For all $1 \le i \le n$ and $1 \le j \le m$, make $Grant[i,j] = 0$.
2.  For all $1 \le j \le m$ do:
    2.1.    If $Pri[i,j] = 1$ and $Busy[i] = 1$ for some $i$ then $Grant[i,j] = 1$.
    2.2.    If $Pri[i,j] = 0$ for all $i$ then
            2.2.1.  Let $K$ be the set of all $i$ such that $Busy[i] = 1$ and $Con[i,j] = 1$.
            2.2.2.  If $K$ is not empty, choose a random $k$ in $K$ and make $Grant[k,j] = 1$.
```

6.3.3 Switch Engine

The switch consists of the following modules: Line Cards, Switching Fabric, and Control Unit. The Control Unit receive configuration data from a Host machine directly connected to it. These components are depicted in Figure 6.4.

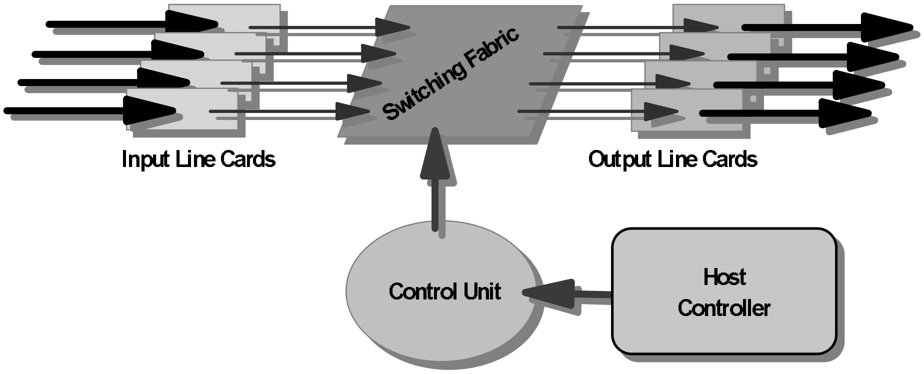


Figure 6.4: Switch organization

The Host machine computes band allocation information off-line with respect to data transmissions. Sporadically, it stores the new configurations in the CT inside the Control Unit. The Control Unit contains the AL that, based on the information in the CT, configures the Switching Fabric to provide the desired input and output connectivity. The Input Line Card receives bit-serial data from the network trunks and converts them into bit-parallel words that are routed to the proper output ports by the Switching Fabric. Similarly, the Output Line Cards converts bit-parallel words into bit-serial streams to be transmitted through the outgoing network trunks.

The Host machines in the network are connected by a traditional network such as a point-to-point modem connection, an Ethernet, or an Internet, depending on how far apart they are. They use these slow speed channels to exchange configuration and control information. This information is used to allocate and synchronize bands. The separation of the high-speed links from the control channels makes the network more reliable and easy to control. For example, loss of band synchronization is easier to detect and communicate using a separate control channel.

The following presents a more detailed look at each component.

Input Line Cards. Line cards are responsible for media conversion (electronic/optical), signal (parallel/serial) generation, and buffering. Figure 6.5 depicts the Input Line Cards. The Isoswitch uses 40-bit words internally.

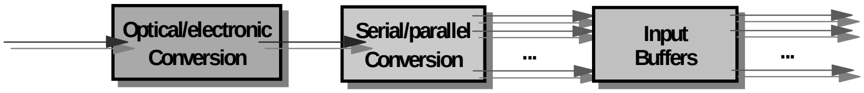


Figure 6.5: Input Card

Input queue buffering is performed after the serial to parallel conversion. In Isochronets, input or output queueing achieve the same performance. This is not usually the case in normal switching technologies due to the “Head of Line” (HoL) blocking effect [Murakami 91]. HoL blocking happens, for example, in PS technology with input queueing of packets. When the HoL is addressed to an output port that is busy, it must wait until the output becomes idle. Other packets that are behind the HoL might be destined to idle output ports, but are effectively blocked by the HoL, diminishing the potential throughput of the switch. The solution to this problem is to employ output queueing in which input packets are sorted according to their destination and placed in an output queue associated with the corresponding output port. Output queueing implementation is complex because it involves switching at *n* (where *n* is the number of input ports) times the rate of the ports. In Isochronets, the HoL effect does not exist because all packets queued at a given input port are being routed through the same routing tree and thus seek the same output port. Because of that, it can employ the less complicated input buffering solution without incurring loss in switch throughput.

Switching Fabric. The Switching Fabric provides the connectivity between the input and Output Line Cards. The structure used in the Isoswitch is depicted in Figure 6.6. It connects each output port to all the input ports. The building blocks for the Switching Fabric are multiplexers, one per output port. Each multiplexer is connected to all input lines. By using the multiplexer selection lines, the appropriate input/output connectivity is achieved based on the current switch configuration supplied by the AL. The implemented Isoswitch has four input and four output ports.

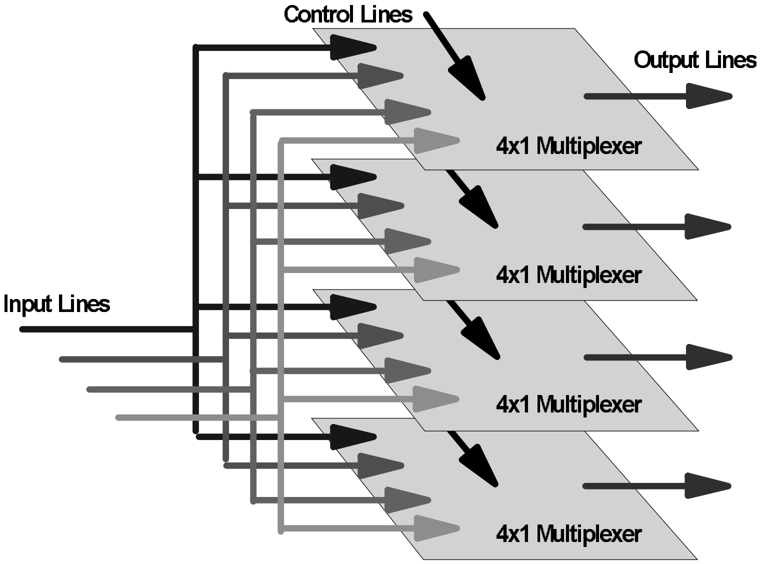


Figure 6.6: Switching Fabric

The Switching Fabric architecture is independent of Isochronets operations, and can be implemented using any available interconnection architecture. In particular, architectures that do not necessarily dependent on the contents of internal frame headers for switching decisions are best suitable for RDMA.

Multi-stage interconnection networks [Adams et al. 87, Broomell and Heath 83, Feng 81] are one such classes of architectures. They reduce the internal complexity of the fabric at the cost of extra stages (and thus added end-to-end delay). Such interconnection can replace the one in Figure 6.6, and can be controlled directly from the Control Unit. The AL decides how to control each stage in the interconnection based on the contents of the CT and the busy lines.

In the proposed design, a complete configuration is used to simplify the logic, since the number of ports is small and consequently the internal fabric complexity is small.

Control Unit. Configuration and arbitration are the main functions of the Control Unit, depicted in Figure 6.7. The CT (discussed in Section 6.3.1) is implemented in the Configuration Memory (CM), explained in more details later. The AL (discussed in Section 6.3.2) is implemented as a combinational circuit.

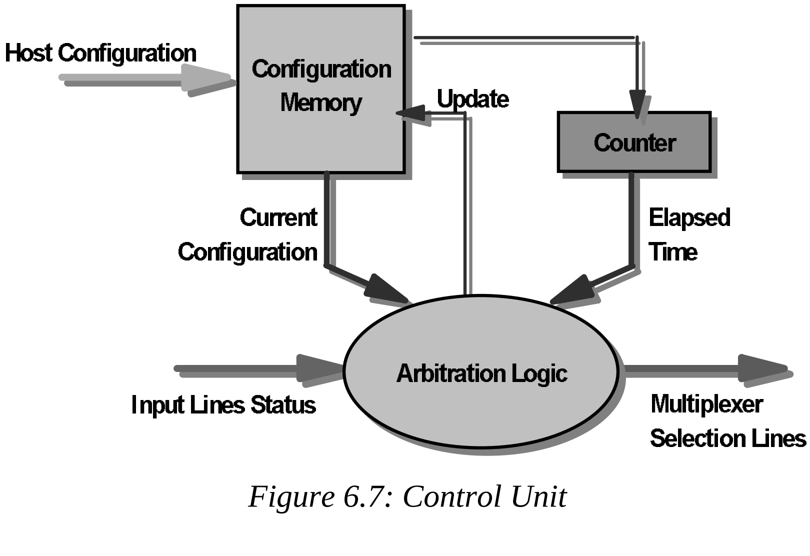


Figure 6.7: Control Unit

New configurations are loaded from the Host machine into the CM. It works similarly to a main memory in a computer. The CM contains switch configurations that are fetched in sequence, much like instructions are accessed in computers. When a new configuration is fetched, the duration for the configuration in the Expiration field is loaded in the Counter register. At each clock tick, the counter is decremented to reflect the elapsed time. When the Counter is 0, it is time to fetch a new configuration from the CM. The sequence of all configurations form the Isochronet cycle that is repeated forever.

During each configuration, the Port Connection and Priority Port fields in the current CT entry along with input line status information on which lines are busy are supplied to the AL that decides the configuration of the multiplexers in the Switching Fabric. The correct input and output connectivity is achieved by selecting the proper multiplexer control.

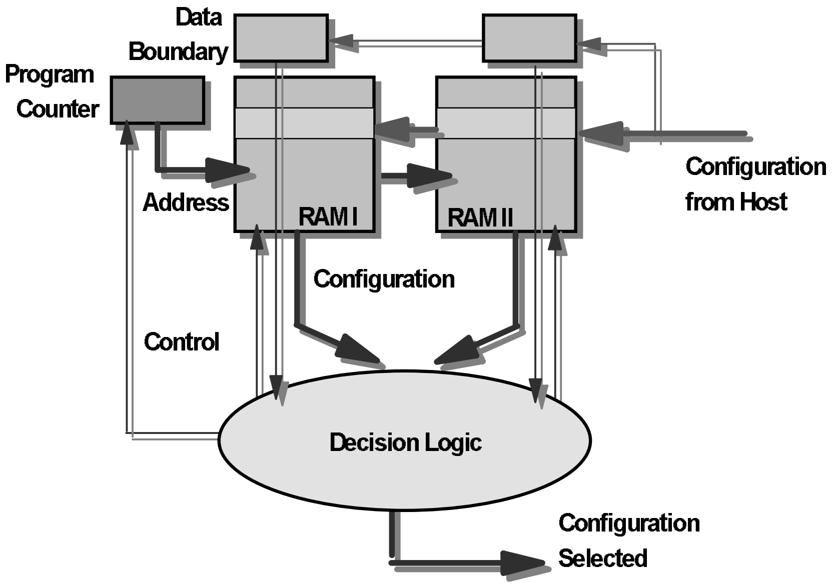


Figure 6.8: Configuration Memory

Figure 6.8 illustrates the CM. Two tandem memories (RAMs) are used to store CTs. Only one of the RAMs is being used by the AL at any given time. When a new CT needs to be loaded from the Host machine, the other RAM is used to store it. In this manner, a new CT may be loaded while the switch is still operating with the old CT. When the current cycle is finished, the switch may operate using the new CT stored in the other RAM.

The Program Counter points always to the address of the current configuration. The RAM containing the current CT is addressed by the Program Counter and supplies the configuration to the Decision Logic that selects the valid RAM. The Program Counter is incremented when the current entry of the CT expires, to point to the next CT. The Data Boundary register contains the address of the last valid configuration. The Program Counter is reset when it reaches this address to restart the cycle.

Meanwhile, the Host can directly address the other RAM to store the next CT concurrently. It also stores the boundary information in the respective Data Boundary register. Once finished, it signals the Decision Logic so that the latter will begin using the new CT after the current Isochronet cycle is finished.

An interesting characteristic of the current implementation is that it uses a special kind of RAM. The RAM is asynchronous and always outputs the current word pointed by the Program Counter. When the Program Counter changes, the time to fetch the next word and stabilize it in the RAM output lines is less than 40 ns in the current implementation. The Program Counter can change and the new configuration used by the AL in the same clock tick. In other words, the change in configuration does not take any overhead because it can occur in parallel within the delay of the clock tick used for arbitration. Other components, like the AL delay, define the size of the clock tick rather than the time to fetch words in the RAM.

Output Line Cards. Figure 6.9 depicts the Output Line Card. Besides performing the inverse function of the Input Line Card, it has a delay element before the parallel to serial bit conversion module. The main function of the Delay Module is to make the whole link propagation delay a multiple of the cycle period. In this manner, the cycles at each node begin a the same time and band synchronization becomes simple. More details on the protocols that use this feature can be found in Chapter 4.

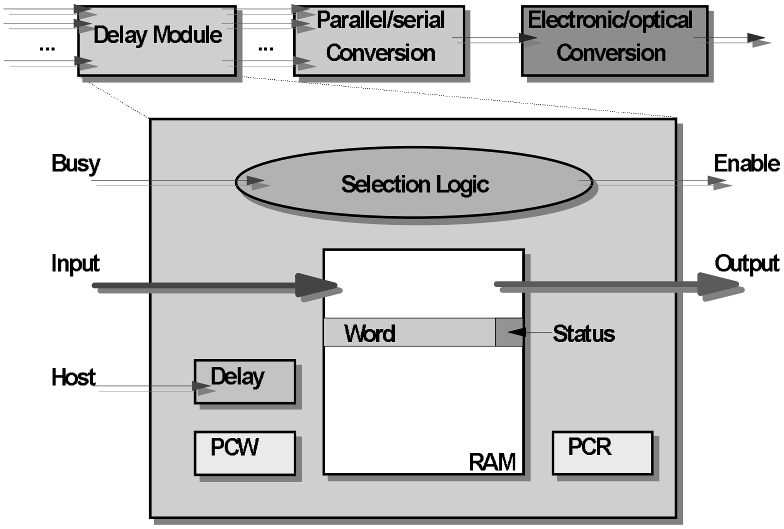


Figure 6.9: Output Card

The main function of the Delay Module is to delay transmissions according to a Host specified amount, between 0 and the cycle period. The Host monitors each link end-to-end propagation delay and then decides which value is to be set in the delay module. Such delay (in clock ticks) is placed in the Delay register. The dual port RAM can be written and read concurrently. The Program Counter Write (PCW) is used to write words to the RAM, while the Program Counter Read (PCR) is used to read words from the RAM. At each clock tick, a word is written into the RAM. If the input was busy, the word written is valid and the status bit is marked with 1. If the input was idle, the word is not valid and its status is marked with 0. Words are then fetched using PCR. If the word fetched from PCR has status 1, it is transmitted. If it has status 0, it is not transmitted. To achieve the delay effect, PCW is initially equal to the contents of Delay while PCR is equal to 0. In this way, the number of words stored in Delay elapse before the first transmission occurs. This assures the required delay.

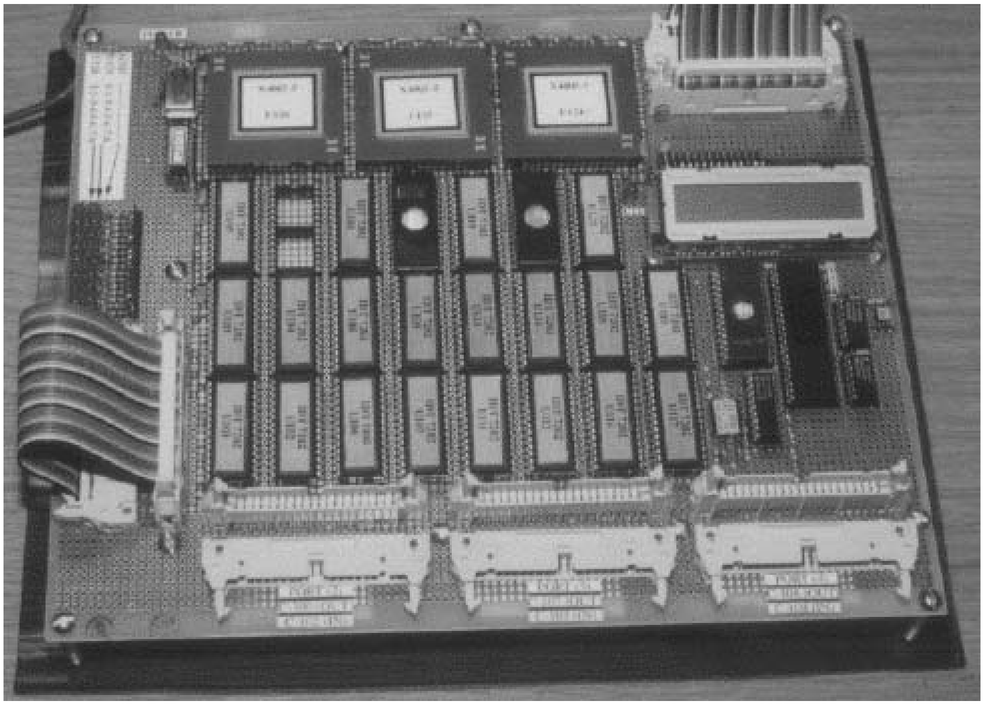


Figure 6.10: Isoswitch implementation

6.3.4 Operational Characteristics

The Isoswitch was implemented using the 4005H family Xilinx Logic Cell Arrays (LCAs). A picture of the prototype is shown in Figure 6.10. LCAs are relatively slow, but were chosen due to the ease of changing the implemented design. This feature is essential for a prototype design and implementation. To get a feel of how slow the adopted 4005H LCAs operate, the implementation of a single D flip-flop [Roth 92] in the chip can achieve a throughput of only 38.5 MHz and the minimal latency through a single combinational gate is 5 ns. Nevertheless, even using such slow components, it was possible to accomplish the goal of implementing a 1 Gb/s per port Isoswitch due to the simplicity in Isochronet operations.

The Isoswitch Control Unit operates at 3.125 MHz, that is, it reaches more than 3 million switching decisions per second. Internally, the data bus is 40-bit wide. In the absence of contention or queueing delays, it takes 320 ns from the time a frame arrives at the switch until it is selected for switching. At each clock tick, 8 40-bit data words are transferred through the switch. The nominal rate per output port is thus 3.125×8×40 Mb/s, or 1 Gb/s. Additionally, each data word takes 40 ns to cross the Switching Fabric from the input to the output port.

6.4 Isochronet Interfaces

The Isoswitch interface must provide, as basic functionality, signaling of network status and basic means for data transfer. The status signaling embody information such as current enabled destinations, priority sources, etc. This is a typical example of a loosely-synchronous network interface discussed in Chapter 5.

Figure 6.11 depicts the interface card to the Isoswitch. Buffers are placed in the interface for data transmission and reception. The objective of the Transmission Buffer is to gather data from the slower Host machine so that it can be sent at full speed through the switch. The Reception Buffer is used to store data received from the high-speed switch until the Host machine can access it.

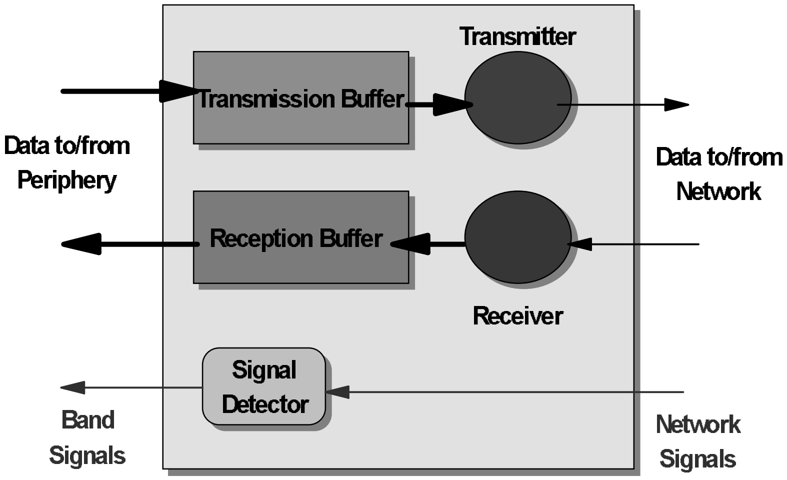


Figure 6.11: Isochronet interface to an end machine

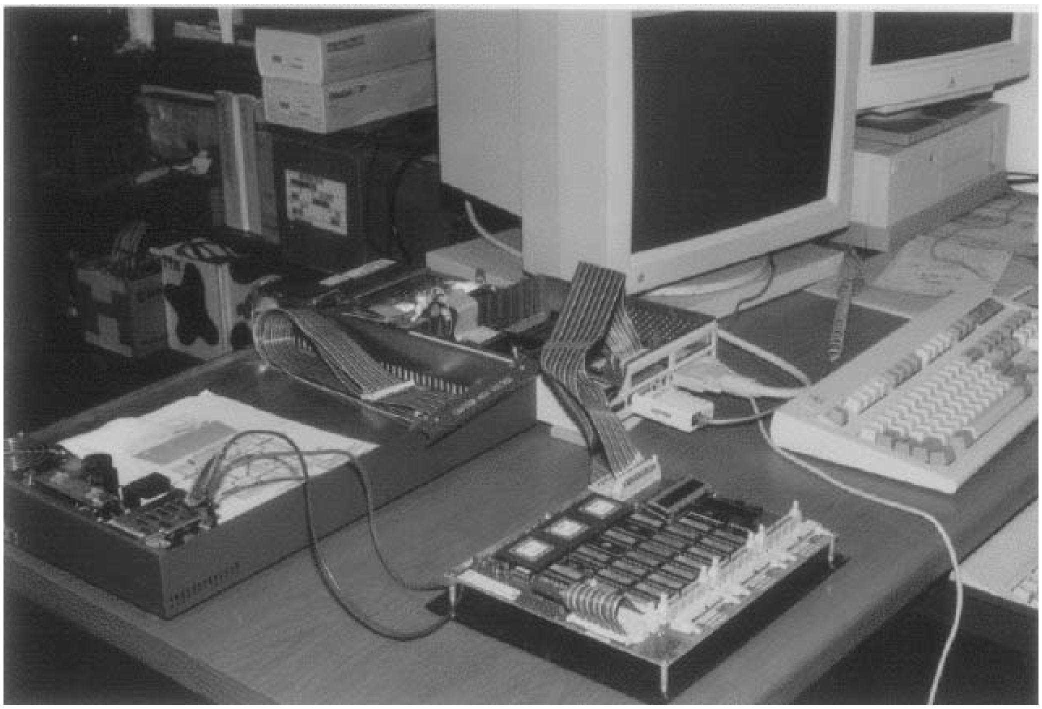


Figure 6.12: Interconnection between the Isoswitch and the Host machine

The card receives signaling information from the switch when the following events occur: cycle begins, bands begin, and data reception. These signals are forwarded as interrupts to the Host machine containing the interface card. A status register in the card can be read by the Host to retrieve the details of what event spawned the interrupt. Alternatively, the status register may be used to check the same events without enabling the interrupts. In this mode, the Host can poll the status registers to see when events occur. Finally, a control register in the card can be used by the Host to enable or disable some of the events or interrupts.

This line card was implemented as an interface between the switch and Sun SPARC [Frank and Lyle 90] machines. A picture showing the connection of the switch to the SPARC containing the card is shown in Figure 6.12. The card can send data to the switch at the peak 1 Gb/s rate once the Transmission Buffers contains at least 8 words. Similarly, data is received from the switch at 1 Gb/s. Nevertheless, the speed at which the SPARC can fill the card buffers is dependent on the protocol used by its bus (the SBus). Measures of the SBus transfer rate between processor registers and the implemented card showed a maximum throughput of 22 Mb/s.

6.5 Efficiency, Complexity, and Scalability

This section evaluates the switch efficiency and complexity, and how these values scale with the number of ports and the link speeds. These measures are evaluated for the Control Unit logic and the Switching Fabric logic.

Control Unit Logic. The Isoswitch switching functions is performed by the Control Unit. In order to find out the effect of increasing the number of switch ports it thus suffices to evaluate the AL complexity and latency in Algorithm 6.1. Step 1 is not really performed in the implemented hardware because the flip-flops that implement *Grant* have default value 0. The operations in Step 2 are performed in parallel, by *m* similar combinational logic circuits. Each of the circuits has as input a portion of *Pri* of size *n* plus *Busy* that is of size *n* as well (see steps 2.1 and 2.2). Thus, each of the *m* parallel circuits has complexity *O*(*n*). The overall AL complexity is thus *O*(*nm*). As for latency, since *m* parallel combinational circuits are processing *n* entries, and since each instruction has linear latency, the total latency is *O*(*n*). Notice that both complexity and latency are also optimal. Firstly, the switch has *n* input ports (and thus arbitration needs to at least read all *n* input status). Secondly, it has *m* output ports (and thus must decide *m* problems of size *n*).

The whole Control Unit (including the tandem memory components) is implemented in a single LCA. It occupies 92% of the LCA logic and 45% of its pins.

It is important to notice that the Isoswitch AL complexity and latency depend only on the values for *n* and *m*. They do not depend on other factors normally found in other switching architectures for HSNs such as processing per frame header. Processing and link transmission per frame affect traditional switch arbitration complexity as follows. If the link speed in *b* (in bits per second), frames have size *f* (in bits) and each frame takes *c* (in seconds) to have their headers processed, the switch needs computational power to process each input port at (*b/f*)*c* frames per second. For example, an ATM switch with 1 Gb/s lines needs to be able to process about 2.36 million cells in one second or one frame every 424 ns per port. Since in the Isoswitch *c=0,* the processing component does not affect the switch complexity.

Nonetheless, the link speed does affect the Control Unit logic speed in the Isoswitch. This dependency may be overcome due to the simplicity in Isoswitch operations, which makes it possible to run it at very high clock rates. The AL must make decisions at a rate of *b/f* per outport. Since each Isoswitch outport is processed independently, each of these circuits must have a maximum latency of *f/b*. For example, at 1 Gb/s and 53 bit frames, the latency must be 424 ns. Notice that such constraint is not a problem, since only simple arbitration is to be performed (no processing dependent on frame headers). For example, the arbitration latency for the implemented Isoswitch is 320 ns.

The number of ports *n* also affects the control logic speeds. When the number of ports is increased, the Control Unit needs to increase its arbitration decisions linearly. In reality both the constraints on link speed and on number of ports can be overcome by the following techniques applicable in RDMA.

In the first technique, each periphery source must send always *w* consecutive frames. Since arbitration is now performed for the whole block of *w* frames, its latency may be *wf/b* (that is, *w* times bigger than originally). Such technique cannot be used in any switching technique that needs to process frames because the switching decision for each of the *w* individual frames may be different. Furthermore, this technique does not affect the frame size (that is, it is different of creating a new network frame size). For example, if 8 ATM cells are always sent through the Isoswitch, the tolerated latency is 2.56 µs using this technique.

In the second technique, the internal data bus in the switch is increased. By increasing the bus size *d* times, the Control Unit can operate *d* times slower. Such technique also relies heavily on the fact that no frame-dependent processing occurs in the Isoswitch.

Switching Fabric Logic. The implementation complexity is reduced in Isochronets since no processing occurs in the fabric. The complexity in terms of the number of input to output lines is independent of the RDMA technique. Care must be taken to avoid a large number of connections in the fabric. Many existing solutions to reduce the number of lines can be adopted in the Isoswitch. For example, one may employ multi-stage interconnection networks, as discussed in Section 6.3.3.

The Switching Fabric implemented contains *m* tandem multiplexers with *n* inputs each (see 6.3.3). Thus, its complexity is *O(nm)* and its latency is *O(n)*. The Switching Fabric implementation occupies only 43% of the LCA chip logic, while is uses 92% of the pins. It is implemented in two LCAs. The total number of LCAs used in the switch is thus 3, 1 for the Control Unit and 2 for the Switching Fabric.

Another factor that increases the Isoswitch scalability is its modular design. Multiple switches may be interconnected to build an *Isohub*, that is, a node consisting of multiple Isoswitches, with higher number of ports. The interconnection of the multiple Isoswitches can be accomplished in many ways.

Finally, as mentioned in Section 6.3.3, the re-configuration overhead is absent in the Isoswitch due to the use of dual memory modules.

6.6 Compatibility with Current Network Devices

The end-to-end delay in interconnected HSNs will heavily depend on how efficiently component networks can operate. For example, when an Ethernet network is connected to an ATM network, a router must be placed at the interface between both networks. The router has to fragment (or assemble) Ethernet frames into (or from) ATM cells, allocate and free ATM virtual connections (circuit or path), prioritize multiplexed Ethernet traffic to deliver necessary QoS, etc. Such functions are not simple and it may very well be the future bottleneck in interconnected HSNs. It is thus fundamental for HSN switches to minimize such routing functions.

Isoswitches can significantly simplify inter-operability among networks because they do not rely on any particular frame structure. Consequently, frames may be forwarded through the switch without adaptation. Other protocol operations can be supported through the Isoswitch directly, with no changes.

6.7 Switching Services

This section analyzes the Isoswitch as a black box and identifies the fundamental services that it offers. The emphasis is on services that can be provided in addition to traditional switching services.

The first new service is the provision of synchronization signals. The Isoswitch issues synchronization signals that identify the beginning of bands and cycles. These signals may be used by periphery nodes to synchronize their local clocks. As a result, global synchronization can be achieved by local exchange between network switches and attached periphery nodes.

Synchronization signals can additionally be used in the protocol stacks at the end nodes to provide synchronous services. A whole new synchronous protocol stack, detailed in Chapter 5, can be implemented using this feature.

These signals are novel when compared to traditional STM synchronization in many respects. First, the necessary accuracy of Isoswitch signals is much lower. For example, an 8 bit slot in a STM frame at 2.4 Gb/s transmission rate lasts 3.3 ns. Typical bands last between a few hundreds of nanoseconds up to a few scores of microseconds. Second, no global network-wide synchronization is necessary among the periphery nodes. Third, the signals embody routing and QoS information that may be used by periphery nodes to schedule their activities. For example, these signals can be forwarded to applications that can schedule their activities to the signals.

The second new service is direct frame forwarding. That is, the Isoswitch does not rely on any particular frame structure to operate. On the contrary, any protocol frame may be directly forwarded through the switch without adaptation thus significantly simplifying work at periphery nodes.

The third new service is guaranteed (as opposed to statistical) QoS provision. Isoswitches can provide guarantees through priority bands. Once signaled at the beginning of a priority band, sources can transmit to the respective destinations and be assured that no loss or delay due to contention will occur in the switch.

The interface is simple, as explained in Section 6.4. The main functionality is frame forwarding, reception, and signal forwarding. All these functions can be accomplished with simple circuitry.

6.8 All-optical Implementation

This section describes the all-optical RDMA- Isochronet design. An all-optical realization of Isochronets must avoid buffering at intermediate switches. The design proposed in this section uses Wavelength Division Multiplexing (WDM) to allocate one wavelength for each band. Figure 6.13 illustrates how wavelengths are used in each band when there is only one tree per band. The figure displays three such trees atop the network topology. Notice that each tree is allocated to a different band because they all share link (F,E) in the same direction. The idea is to use a different wavelength for each tree. In this case, three wavelengths cross link (F,E), one for each tree. Contention among trees is thus resolved by WDM.

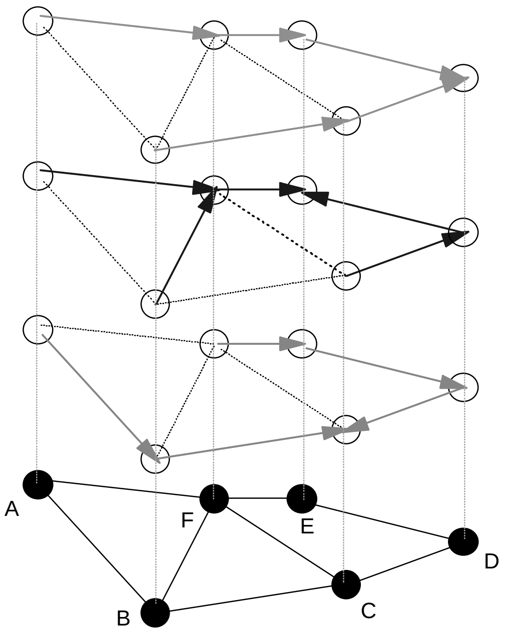


Figure 6.13: WDM allocation to routing trees

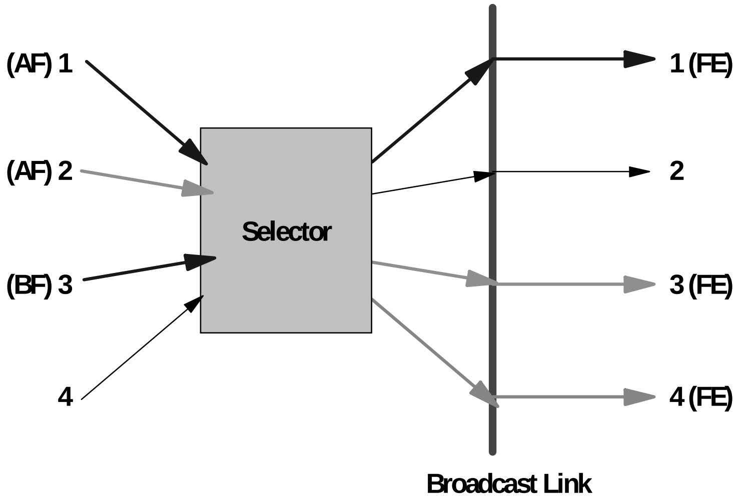


Figure 6.14: All-optical switch implementation: one tree per band

The architecture for Node E in Figure 6.13 is shown in Figure 6.14, where each input and output port is labeled with the corresponding link. Each wavelength is depicted using a different shade of gray. Incoming wavelengths are first fed into a Selector box (explained later) and then multiplexed through a single optical Broadcast Link (the interconnection fabric) connecting all source and destination links. At each output link, a slowly-tunable receiver picks the wavelength of the trees sharing the link. The receiver is directly connected to a slowly-tunable transmitter that regenerates the wavelength in its output link.

Contention in the all-optical implementation is resolved by discarding all but one of the contending frames. Contention occurs when two or more of the input links are sending frames using the same wavelength through the Broadcast Link. In this circumstance, one of the sources must be enabled while the others must be stopped, otherwise all data may be lost. This functionality is achieved through the Selection box, the only electronic component in this architecture, depicted in Figure 6.15. Its function is to detect incoming signals from the links and immediately grant access to one of them, shutting the others.

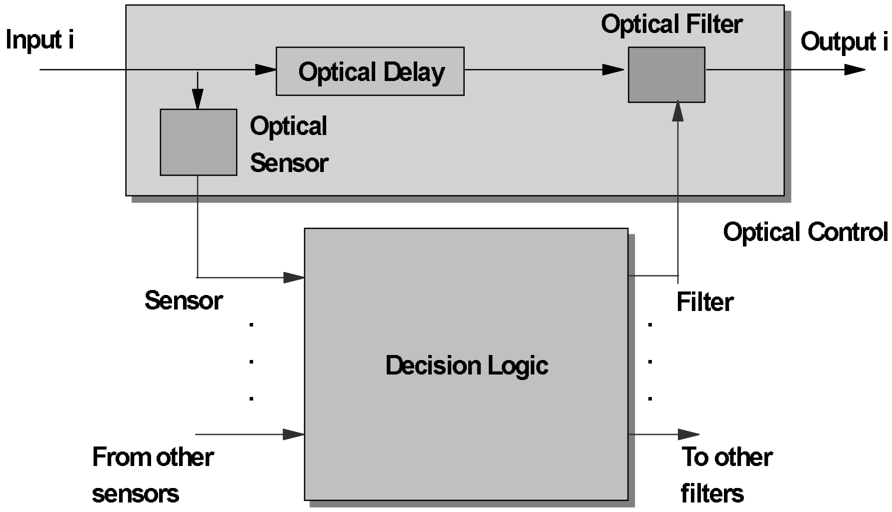


Figure 6.15: Selection box

Each input line is extended inside the selection box to increase the delay from the input to the output ports. This extension is the Optical Delay in the figure. At each entrance, an optical sensor detects incoming light. The idea is to only allow one input per wavelength into the Broadcast Link. The Decision Logic decides which of the inputs should be allowed to proceed and which should be shut. The decision is reached using the AL with inputs from the CT and sensors. The decision is input to the filter at the exit of the line, which passes or block light according to AL instructions. The size of the extension inside the selection box must be big enough to allow the AL reach its decision before the optical data reaches the end of the extension.

Priority bands are implemented as in the electronic Isochronet implementation, using time division inside the selection box. The CT contains the priority data information and the AL uses it when deciding which sources should proceed in each band. Priority bands exist during portions of a cycle, whereas contention bands are always “opened”.

Multiple trees per band are implemented by extending the architecture in Figure 6.14 into multiple Broadcast Links. The idea is to propagate different routing trees in the same Broadcast Link using separate wavelengths, but still be able to reuse wavelengths in separate Broadcast Links. In this manner, the total number of wavelengths necessary is reduced. Notice that trees in the same band do not share any link in the same direction, so they will not interfere at any link if they use the same wavelength.

The implementation of the multiple Broadcast Link scheme works as follows. Each input link is connected to all the Broadcast Links, but optical filters are placed at the interface between each input link and each Broadcast Link to pass or block wavelengths. The filters are set so that, after the filtering phase, no two trees using the same wavelength are broadcast through the same Broadcast Link. After this phase, operations within each Broadcast Link work as explained for the single Broadcast Link case.

The optical Isochronet implementation has many advantages when compared with traditional WDM. First, a small number of wavelengths (at most *n*, where *n* is the number of switches in the network) are needed. Second, no pre-allocation of wavelength or frame processing is necessary for communication. Most recent schemes (see [Ramaswami 93] for a survey of such schemes) either need to provide a special control channel for the reservation of wavelength prior to communication or need to process frames. The reservation schemes suffer drawbacks such as round-trip allocation delay, necessity for rapidly-tunable receivers/transmitters, and dedicated bandwidth. The schemes that switch frames incur the added delay for media conversion and frame processing. Third, the implementation described can use slowly tunable transmitters and receivers since they only need to tune when nodes or links fail, which occurs at much slower rates than the speed of incoming frames.

It is important to notice that all bands are opened all the time, avoiding synchronization of bands.

Frame loss may occur in RDMA- during contention bands. It would be interesting for an all-optical implementation to implement slots in each link to decrease the probability of frame loss. In the following, the frame-loss probability for the slotted implementation when arrivals are Poisson is analyzed. Also, an extension to the basic scheme to reduce the frame-loss probability is suggested.

Let us assume that the input rate to the system is between 0 and 1 (it is simple to extend the following analysis for generic input rates). Let $\frac{\lambda}{n}$ be the input rate ($0 \leq \lambda \leq 1$) of each input link to a particular switch, and n be the number of input links to the switch. The probability of no transmission from a source link during a slot is $1-\frac{\lambda}{n}$. Thus, the average successful transmission rate during a slot is $1-\left(1-\frac{1}{\lambda}\right)^{n}$ (that is, if at least one source transmits). As $n \to \infty$, the successful transmission rate becomes $1-e^{-\lambda}$. The expected success probability is $\frac{\left(1-e^{-\lambda}\right)}{\lambda}$. Finally, the expected loss probability is $1-\frac{\left(1-e^{-\lambda}\right)}{\lambda}$. When $\lambda \to 1$ (loaded system), the expected loss is e^{-1} (less than 37%).

It will be shown in Chapter 7 that the loss rate is bound to 50% when the source stream transmissions are not slotted.

It is possible to improve the performance of this scheme. Multiple copies of the same frame may be sent, thus decreasing the loss probability for the frame. For example, assuming that losses are uniformly distributed among the contending frames, each frame can be repeated m times, resulting in a maximum loss probability of e^{-m} per frame (that is, all m frame copies would have to be lost to lose the frame and, in addition, the input load in the system including the copies would be 1). Thus, m may be computed from the maximum loss rate r that can be tolerated in the system: $r \leq e^{-m}$ so that $m \geq -\ln r$. For example, $m = 4$ insures less than 2% loss rate when the system is heavily loaded and the number of input sources is big.

Notice that when m copies are generated, the maximum input load to the system before copying is at most $1/m$. Thus, $1-(1/m)$ of the potential bandwidth is lost due to the copying process. For example, if $m = 4$, the loss in bandwidth is 75%. This may be reasonable in all-optical networks where the rates are in the order of scores of terabits per second.

To complete the design using the analysis above, a filter is placed at the traffic sources (before the traffic enters the network), which disturbs the input traffic frame inter-arrival times to the network and makes them exponentially distributed (thus generating a Poisson arrival process to the network). Each source sends m copies of the same frame, where m is computed from the tolerated loss rate.

One can observe that the same design used for the electronic design could have been used for the optical RDMA- design as well. That is, bandwidth could be time divided among trees in the optical design as well. The Selection box in Figure 6.15 would be necessary in the optical design to detect and resolve contention among frames. The Control Unit in Figure 6.7 could be used unchanged in the all-optical design. This design using time division is an alternative to the one proposed using wavelength division.

Both have advantages and disadvantages that should be evaluated according to the operational goals of the network. The time division version is cheaper to implement in terms of transmitters and receivers per node and interface cards; it only requires one receiver and one transmitter. Once a band is granted, the whole optical spectrum is available for transmissions. The wavelength version does not require band synchronization. Additionally, the optical spectrum is so rich in bandwidth that it is unlikely that a single source will be able to generate alone such an amount of traffic. If this is the case, wavelength division may be more reasonable.

This chapter chose to elaborate on the wavelength version to show two different design possibilities.

6.9 Future Work

One of the most important extensions to this work is to pursue the implementation of an all-optical switch. The switch has the potential of running at terabit per second rates.

Another interesting experiment is to implement a few Isoswitches to test a full-size network. In a first stage the switches can be implemented as a hub, that is, all nodes in the same box. As a second stage, they could be implemented in a distributed fashion using the delay elements proposed in this chapter.

The study of efficient interfaces to the network is not complete. What is the ideal general purpose interface to Isochronets? The interface proposed in this work is simple and relies to the attached Host queueing and traffic scheduling responsibilities. One interesting extension would be to build a more complex interface where these responsibilities would be embedded in the interface card.

6.10 Summary

The fact that Isochronets reduces all network-layer functionality to the MAC layer significantly simplifies the implementation of its switch. Among other features in Isochronets: (1) no frame header processing is required in the network, (2) there is no need for adaptation layers at the network interface, and (3) internetworking is simplified.

All these features guide the design of the Isoswitch. Because switching is independent of frame headers, both an electronic and an all-optical implementation of the Isoswitch are possible. This work developed both designs in details and described an electronic Isoswitch implementation.

The implemented Isoswitch has four input and output channels operating at 1 Gb/s. The design is modular and scaleable with respect to increases in the number of channels and transmission speeds. Inter-operability with other switching techniques is simplified. Finally, it offers novel services: (1) synchronous signaling, (2) no necessity for adaptation, and (3) guaranteed QoS provision.

Chapter 7

Route Division Multiple Access Performance Evaluation

7.1 Introduction

7.1.1 The Problem

This chapter addressed the problem of evaluating the performance of RDMA when compared with more traditional switching techniques: Packet Switching (PS) and Circuit Switching (CS). The study is carried using both analytical and simulation models.

7.1.2 Main Results

Analytical approximations. The time-dependent behavior of RDMA complicates the performance study. Usual analytical performance analysis techniques ignore time-dependent (or transient) behaviors when simplifying models in search for tractable solutions. This chapter presents approximate closed-form formulas for end-to-end delay and loss in RDMA-, RDMA+, and RDMA++. The analytical approximations are very accurate when checked with the simulation results.

Simulation studies. The natural rescue when analytical techniques fail is simulation. Even though completely realizable, simulation studies must be carefully implemented to avoid extremely long executions and big state-spaces. This chapter presents simulation results on RDMA-, RDMA+, and RDMA++ performance.

Performance study. RDMA is compared favorably with respect to PS and CS for Poisson arrivals and bursty on-off arrivals. For low input loads, PS outperforms RDMA and CS because there is no admission delay waiting for a band or a circuit. When the load increases, frame header processors become saturated and then RDMA outperforms PS. CS always performs worst than RDMA because of the penalty incurred waiting for a circuit that is larger than the one waiting for a band. QoS parameters such as minimal delay and no jitter can be guaranteed with RDMA by using priority bands. Finally, when the propagation delay is large (in wide area communications), the time waiting for a band is negligible when compared with the propagation time. In this scenario, the performance advantages of PS with respect to RDMA become negligible.

7.1.3 Chapter Organization

The reminder of this chapter is organized as follows. Section 7.2 analyzes the performance of RMDA for a single node. Section 7.3 simulates the performance of RDMA for a single node. Section 7.4 simulates the performance of RMDA for a network of nodes. Section 7.5 describes future work. Finally, Section 7.6 summarizes.

7.2 Analysis: Contention for Outgoing Links

This section studies how RDMA resolves contention for an outgoing link. The situation is described in Figure 7.1 where a set of incoming links is competing for an outgoing link. The incoming links are partitioned in classes such that two links are in the same if and only if they are part of the same routing tree. In the figure, there are T routing trees and tree i , $1 \le i \le T$, contains n_i links. Each link is labeled L[*x*,*y*], where *x* is the respective routing tree and *y* is the link order within the tree. The outgoing link needs to be time divided among the T trees. The goal of this section is to build a performance understanding of this scenario using analytical techniques.

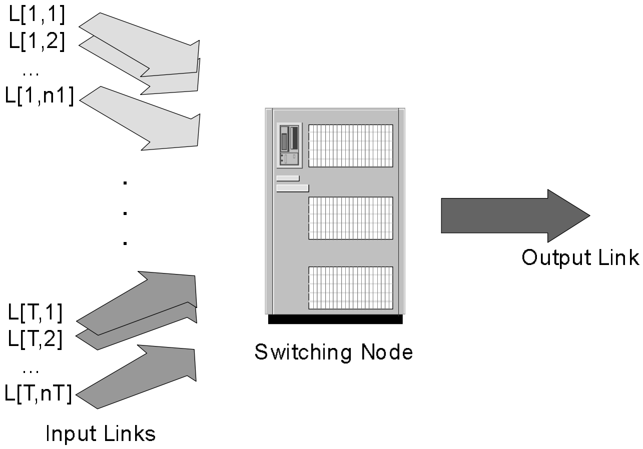


Figure 7.1: Contention for an outgoing link in RDMA

One interesting characteristic of RDMA is that each tree behaves independently of the others. From the perspective of a particular tree, the output link is available to it for a given period U and then becomes unavailable during a period V. The link is effectively a server that can service frames at a given link rate. The server cycle between two states, an operational state of duration U and a vacation state of duration V. The cycle is $C = U + V$. From this point onwards, the study will concentrate on one such independent trees.

It is assumed that the tree under study contains n links. The arrival process is Poisson with mean $\frac{\lambda}{n}$ in each link. It can be thus concluded [Kleinrock 75] that the merged input stream has a Poisson distribution $A(t)$ with mean λ. The Poisson assumption in each link can be justified by noticing that incoming traffic to a contention band is usually random and does not require QoS guarantees.

The frame sizes are distributed according to a generic distribution $B(x)$ with mean μ. The whole tree can be modeled as an M/G/1 system with input rate λ and service rate μ that cycles between an active period of size U and a vacationing period of size V. The system utilization is $\rho = \frac{\lambda}{\mu}$. Notice that, for the system to be stable, $\rho < \frac{U}{U+V}$.

7.2.1 The RDMA- System

Under RDMA- regimen, it is assumed that sources will generate traffic according to the Poisson distribution during each band. If the traffic sources are not Poisson in nature, a filter can be placed between the sources and the input to the system to decrease the chances of burst losses. The filter will output traffic according to the Poisson distribution from any generic input traffic. The filter is thus a G/M/1 system and can be analyzed accordingly [Kleinrock 75] in order to find its contribution in the overall end-to-end frame delay.

If the band size U is large compared to the mean transmission rate μ, then one can assume that the RDMA- system operates like a Type I Counter described in [Karlin and Taylor 75] and summarized here. The Type I Counter registers instantaneous signals. Signals arrive according to a generic arrival distribution $A(t)$ and take some time to be recorded also generic with distribution $B(x)$. While a signal is being recorded, other signals are lost. The similarity with RDMA- is apparent if one replaces the transmission process for the recording process. While a packet is being transmitted in RDMA-, other arriving packets are lost.

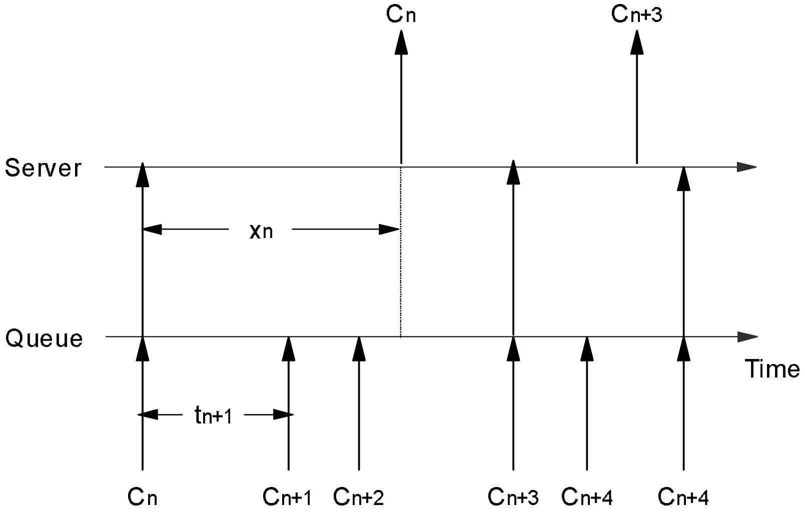


Figure 7.2: RDMA- model

Figure 7.2 illustrates the model for the RDMA- systems. C_n is the n-th frame to arrive to the system, t_{n+1} is the interval between the arrival of frames C_n and C_{n+1}, and x_n is the time to transmit frame n. The figure depicts three stages of the process: frames arriving to the queue, frames leaving the queue and immediately entering the server (that is, starting transmissions), and frames leaving the server. The arrows represent frames and the horizontal lines the queue and the server. If the link is not transmitting any information, the arriving frame is immediately serviced. Otherwise, it is lost. In the figure, only frames C_n, C_{n+3}, and C_{n+4} are not lost.

Following [Karlin and Taylor 75], let z_i denote the interval of time between the i-th and (i+1)-th successful transmissions, γ_t be the residual life of t_i at time t, $S_k = \sum_{i=1}^{k} t_i$ be the time of the *k*-th arrival, and $N(t) = \sup\{n: S_n \le t\}$ be the number of arrivals to the system by time *t*. The residual life γ_t of t_i is the time remaining for the arrival of the next frame at time *t*. Then z_1 can be computed as:

$$z_1 = x_1 + \gamma_{x_1} = S_{N(x_1)+1} \tag{7.1}$$

That is, the interval between the first and second successful transmissions is equal to the time to transmit the first frame plus the residual time for the arrival of the next frame after the first finished being transmitted.

Example 7.1: Distribution of the inter-departure time when the service has a deterministic distribution.

If the service time is always $\dot{x}$, then $B(x) = \begin{cases} 1 \text{ if } x \ge \dot{x} \\ 0 \text{ if } x < \dot{x} \end{cases}$, where $\frac{1}{\mu} = \dot{x}$.

Then, if $z > \dot{x}$:

$$\Pr\{z_1 \le z\} = \int_0^{z-\dot{x}} \lambda e^{-\lambda x} dx = 1 - e^{-\lambda(z-\dot{x})}$$

and if $z \le \dot{x}$:

$$\Pr\{z_1 \le z\} = 0.$$

This example shows the inter-departure distribution when ATM [De Prycker 93, Le Boudec 92] cells are being transmitted. The time to transmit an ATM cell is $\dot{x}$.

Since the processes $\{t_n\}$ and $\{x_n\}$ are independent, the law of total probabilities gives:

$$\Pr\{z_1 \le z\} = \int_0^z \Pr\{x + \gamma_x \le z | x_1 = x\} dB(x) \tag{7.2}$$

Defining the distribution of γ_t as $\hat{A}_w(t) = \Pr\{\gamma_t > w\}$ (an explicit formula for it can be found in [Karlin and Taylor 75]):

$$\Pr\{z_1 \le z\} = \int_0^z \{1 - \hat{A}_{z-x}(x)\} dB(x) \tag{7.3}$$

Since the arrival process is Poisson, the memory-less property of the interarrival exponential distribution leads to the conclusion that γ_t follows an exponential distribution. Thus for RDMA-:

$$\Pr\{z_1 \le z\} = \int_0^z B(z-x)\lambda e^{-\lambda x} dx \tag{7.4}$$

and:

$$\Pr\{z_1 > z\} = 1 - \int_0^z B(z-x)\lambda e^{-\lambda x} dx \tag{7.5}$$

But:

$$\int_0^z \lambda e^{-\lambda x} dx = 1 - e^{-\lambda z} \Leftrightarrow \int_0^z \lambda e^{-\lambda x} dx + e^{-\lambda z} = 1 \tag{7.6}$$

Replacing the number 1 in Equation 7.5 by the number 1 in Equation 7.6:

$$\Pr\{z_1 > z\} = \int_0^z \lambda e^{-\lambda x} dx + e^{-\lambda z} - \int_0^z B(z-x)\lambda e^{-\lambda x} dx \tag{7.7}$$

$$\Pr\{z_1 > z\} = e^{-\lambda z} - \int_0^z \{1 - B(z-x)\}\lambda e^{-\lambda x} dx = e^{-\lambda z} - \int_0^z \overline{B}(z-x)\lambda e^{-\lambda x} dx \tag{7.8}$$

Finally, one can conclude that Equation 7.8 is valid for any z_n because $\{z_n\}$ are the inter-arrival times of a renewal process [Karlin and Taylor 75].

Examples 7.1 and 7.2 illustrate how Equation 7.4 can be used to compute the distribution of the inter-departure times of successfully transmitted frames.

Example 7.2: *Distribution of the inter-departure time when the service has a Poisson distribution.*

In this case, $B(x) = 1 - e^{-\mu x}$.

Then:

$$\Pr\{z_1 \le z\} = \int_0^z (1 - e^{-\mu(z-x)})\lambda e^{-\lambda x} dx = \int_0^z \lambda e^{-\lambda x} dx - \lambda e^{-\mu z} \int_0^z e^{(\mu-\lambda)x} dx$$

$$\Pr\{z_1 \le z\} = 1 - e^{-\lambda z} - (\lambda e^{-\mu z} \frac{e^{(\mu-\lambda)z} - 1}{\mu - \lambda}) = 1 - e^{-\lambda z} - \lambda \frac{e^{-\mu z} - e^{-\lambda z}}{\lambda - \mu}$$

$$\Pr\{z_1 \le z\} = 1 - \frac{\lambda e^{-\mu z} - \mu e^{-\lambda z}}{\lambda - \mu}$$

This example shows the inter-departure distribution when ATM cells are being transmitted in batches of random sizes.

The reminder of this section uses Equation 7.8 to derive the mean loss rate in RDMA-. One can observe that frames are being transmitted at rate $\frac{1}{E[z]}$ while they are being received at rate $\frac{1}{E[x]} = \lambda$. Thus, the packet loss rate L is:

$$L = 1 - \frac{1/E[z]}{1/E[x]} = 1 - \frac{1}{\lambda E[z]} \qquad (7.9)$$

The expectation can be computed as:

$$E[z] = \int_0^\infty \Pr\{z_1 > z\} dz = \int_0^\infty e^{-\lambda z} dz + \int_0^\infty \int_0^z \overline{B}(z-x)\lambda e^{-\lambda x} dx dz \qquad (7.10)$$

But:

$$\int_0^\infty e^{-\lambda z} dz = \frac{1}{\lambda} \qquad (7.11)$$

and:

$$\begin{aligned}\int_0^\infty \int_0^z \overline{B}(z-x)\lambda e^{-\lambda x}dxdz &= \int_0^\infty \int_x^\infty \overline{B}(z-x)\lambda e^{-\lambda x}dzdx \\ &= \int_0^\infty \lambda e^{-\lambda x}\int_x^\infty \overline{B}(z-x)dzdx \\ &= \int_0^\infty \lambda e^{-\lambda x}\frac{1}{\mu}dx \\ &= \frac{1}{\mu}\end{aligned} \tag{7.12}$$

Replacing Equations 7.11 and 7.12 in 7.10, one obtains:

$$E[z] = \frac{1}{\lambda} + \frac{1}{\mu} = \frac{\lambda+\mu}{\lambda\mu} \tag{7.13}$$

Finally:

$$L = 1 - \frac{1}{\lambda(\frac{\lambda+\mu}{\lambda\mu})} = \frac{\lambda}{\lambda+\mu} = \frac{\rho}{\rho+1} \tag{7.14}$$

An interesting conclusion that can be derived from Equation 7.14 is that under RDMA- policy, when $\rho \to 1,\ L \to 0.5$. That is, the upper bound on the loss rate for Poisson arrivals and generic service time is 50%.

The delay in the node in RDMA- is equal to the service time of a frame. The mean delay D is thus:

$$D = \frac{1}{\mu} \tag{7.15}$$

7.2.2 The RDMA++ System

RDMA++ is a challenging system to analyze because of its time dependent behavior. From the point of view of an end node, the server is the routing tree. It is active during its band and then goes on vacation during other bands. A complete state description of such a system must include the current server status and how long it will last. References [Federgruen and Green 86, Ott 87] study systems with deterministic time dependent vacations and can provide only limited insight on the system behavior.

This section presents an approximation for the average delay and loss in the RDMA++ system when the queueing discipline is M/G/1 and the system load is low to medium.

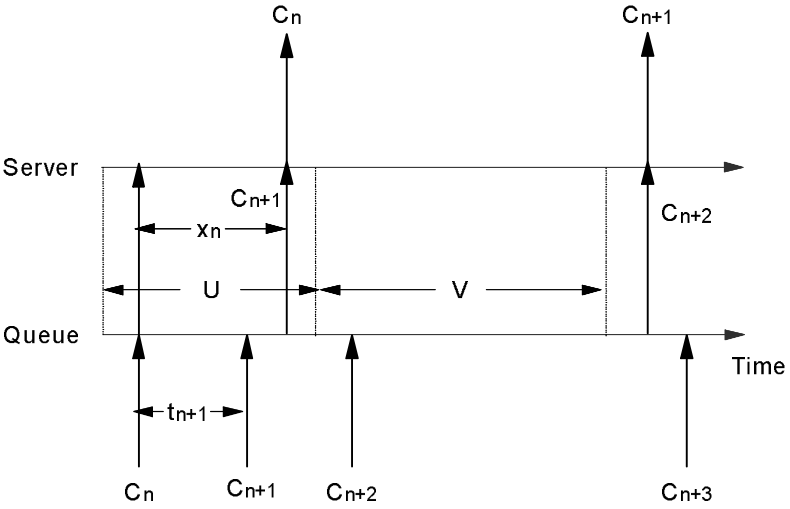


Figure 7.3: RDMA++ model

Figure 7.3 illustrates the model for the RDMA++ system. The terminology is the same as in Figure 7.2. Frame C_n arrives to an empty system and is serviced immediately. C_{n+1} waits until C_n is finished and then begins to be serviced. During its service time, the server goes on vacation and then returns to complete the service. Only then is C_{n+2} admitted into service.

Following [Bertsekas and Gallager 92], let w_i be the waiting time in the queue for the i-th frame, q_i be the number of frames found waiting in the queue by the i-th frame upon arrival, and r_i the residual service time seen by the i-th arriving frame. The residual service time is the service time remaining for the frame in service upon i's arrival. Similarly, v_i is the residual vacation time seen by frame i upon arrival. Then:

$$w_i = r_i + v_i + \sum_{j=i-q_i}^{i-1} x_j + \left\lfloor \left(r_i + \sum_{j=i-q_i}^{i-1} x_j \right) \Big/ U \right\rfloor V \tag{7.16}$$

That is, the waiting time for frame i upon arrival is equal to the sum of four components: (1) residual service time for the frame in service, (2) the residual vacation time, (3) the time to service all frames in the queue, and (4) the vacations incurred while servicing the frames in front of the arriving one. This last component is vacation period V multiplied by the total number of cycles necessary to service the frame in service upon arrival plus the ones in the queue. The number of cycles in question is the floor of the total time to service the frames in the queue and the one in service upon arrival divided by the total operational time in each cycle.

The next step is to take the expectation of the terms in Equation 7.16. Nonetheless, expectation of the last term of the summation in the right side is difficult to compute. One can recur to the following approximation for low loads. From the point of view of the arriving frame, it is very likely that all work in front of it will be finished during one active period because the load in the system is small. In this case, $\left\lfloor \frac{1}{U} \left(r_i + \sum_{j=i-N_i}^{i-1} x_j \right) \right\rfloor = 0$.

The approximate expectation of the terms in Equation 7.16 the becomes:

$$E[w_i] = E[r_i] + E[v_i] + E[\sum_{j=i-q_i}^{i-1} E[x_j | q_i]] \tag{7.17}$$

$$E[w_i] = E[r_i] + E[v_i] + \frac{1}{\mu} E[q_i] \tag{7.18}$$

Taking the limit as $i \to \infty$:

$$E[w] = E[r] + E[v] + \frac{1}{\mu} E[q] \tag{7.19}$$

But by Little's Law [Kleinrock 75], $E[q] = \lambda E[w]$ and so:

$$E[w] = E[r] + E[v] + \frac{1}{\mu} \lambda E[w] \tag{7.20}$$

Finally:

$$E[w] = \frac{E[r] + E[v]}{1 - \rho} \tag{7.21}$$

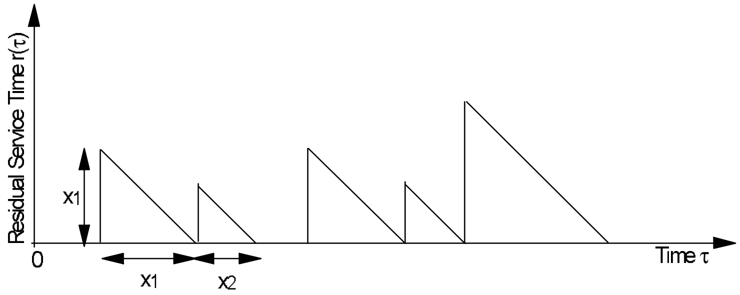


Figure 7.4: Residual service time

The mean residual service and vacation times can be computed using a graphical argument [Bertsekas and Gallager 92]. The residual service time is depicted in Figure 7.4. The residual time at *t*, *r*(*t*) is the time left to complete the frame transmission. When a new frame arrives, *r*(*t*) becomes equal to the service time for the frame. It then decreases linearly with time until it vanishes. The behavior is repeated for the next frame and so on.

The mean residual time at *t* can be computed from the time average of *r*(*t*), given that the number of frames serviced by time *t* is *M*(*t*), as:

$$\frac{1}{t} \int_0^t r(\tau) d\tau = \frac{1}{t} \sum_{i=1}^{M(t)} \frac{1}{2} x_i^2 \tag{7.22}$$

That is, the time average sum of the areas of all triangles in Figure 7.4 by time *t*. Equation 7.22 can also be written as:

$$\frac{1}{t}\int_0^t r(\tau)d\tau = \frac{1}{2}\frac{M(t)}{t}\frac{\sum_{i=1}^{M(t)} x_i^2}{M(t)} \tag{7.23}$$

Taking the limits in Equation 7.23:

$$\lim_{t\to\infty}\frac{1}{t}\int_0^t r(\tau)d\tau = \frac{1}{2}\lim_{t\to\infty}\frac{M(t)}{t}\cdot\lim_{t\to\infty}\frac{\sum_{i=1}^{M(t)} x_i^2}{M(t)} \tag{7.24}$$

The first term in the right hand side multiplication is the one half of the input rate while the second is the expectation of the service time squared. The left hand side is the expectation of the residual service time. Formalizing this observation:

$$E[r] = \frac{1}{2}\lambda E[x^2] = \frac{1}{2}\lambda\overline{x^2} \tag{7.25}$$

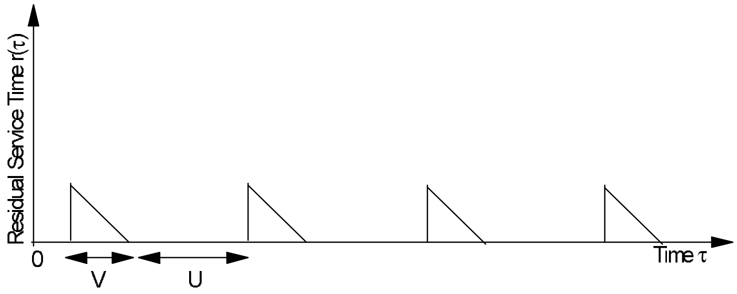


Figure 7.5: Residual vacation time

The mean residual vacation can be computed similarly to the mean residual service time. Figure 7.5 depicts the residual vacation time, where the interpretation is similar to the one for Figure 7.4. The average residual vacation is:

$$\frac{1}{t}\int_0^t v(\tau)d\tau = \frac{1}{t}\cdot\left\lfloor\frac{t}{V+U}\right\rfloor\cdot\frac{1}{2}V^2 \tag{7.26}$$

That is, the time average of the areas of the triangles in Figure 7.5 by time *t*. Taking the limits in Equation 7.28:

$$E[v] = \lim_{t\to\infty} \frac{1}{t}\int_0^t v(\tau)d\tau = \lim_{t\to\infty} \frac{1}{t}\cdot\frac{t}{V+U}\cdot\frac{1}{2}V^2 = \frac{1}{2}\frac{V^2}{V+U} \tag{7.27}$$

Replacing Equation 7.25 and 7.27 in Equation 7.21:

$$E[w] = \frac{1}{2}\frac{\lambda\overline{x^2}}{(1-\rho)} + \frac{1}{2}\frac{V^2}{(U+V)}\frac{1}{(1-\rho)} \tag{7.28}$$

From which it can be concluded that:

$$D = \frac{1}{2}\frac{\lambda\overline{x^2}}{(1-\rho)} + \frac{1}{2}\frac{V^2}{(U+V)}\frac{1}{(1-\rho)} + \frac{1}{\mu} \tag{7.29}$$

Notice that Equation 7.28 can be interpreted as follows. The first component of the summation is the normal waiting time of an M/G/1 system. The second component represents the penalty due to the residual vacation time. It is equal to the mean vacation period ($\frac{V}{2}$) times the probability of being on vacation ($\frac{V}{U+V}$) divided by the mean idle period ($1-\rho$) to obtain the total waiting time due to residual vacations.

Finally, there is no loss in RDMA++ and thus:

$$L = 0 \tag{7.30}$$

An interesting conclusion that can be derived from the RDMA++ analysis is that the same approximation for mean delay can be used in CS. CS operates similarly to RDMA++. The circuit comes periodically within a cycle, in similarity to a band arriving within a cycle. In conclusion, Equations 7.29 and 7.30 are also valid approximations for CS where *U* should be interpreted as the circuit duration.

7.2.3 The RDMA+ System

If the band size U is large compared to the mean transmission rate μ, then one can assume that RDMA+ operates in similarity to a M/G/1 queueing system with periodic losses. Figure 7.6 illustrates the typical behavior of an RDMA+ system. Frames behave like in RDMA++, except when the band ends. At that point, all frames in the system are lost. For example, in the figure, frame C_{n+1} is lost.

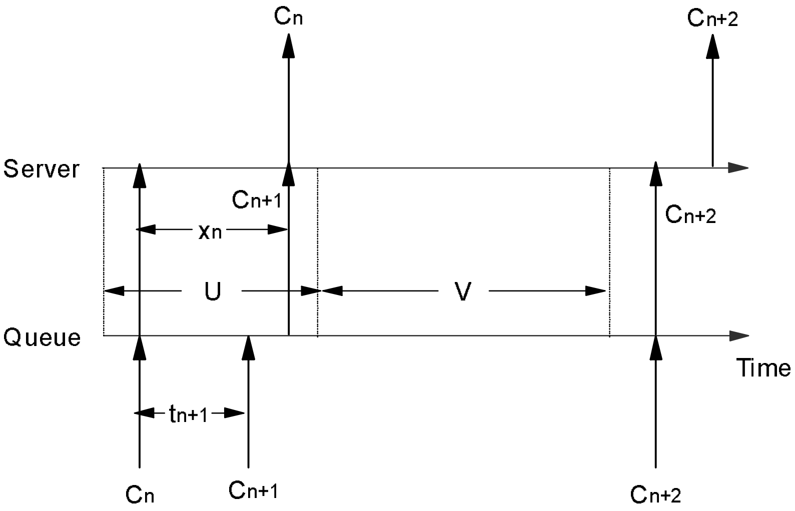


Figure 7.6: RDMA+ model

The model adopted here is just an M/G/1 queueing system. Delays are computed directly using M/G/1 delay formulas [Kleinrock 75]. For example, the mean delay is:

$$D = \frac{\lambda \overline{x^2}}{2(1-\rho)} + \frac{1}{\mu} \tag{7.31}$$

The delay distribution can be computed similarly using the M/G/1 formulas.

The loss probability can be approximated observing the following. If the band is big enough such that the system eventually becomes stable, then the probability that n frames will be lost at the end of the band is equal to the probability that n frames are in the system at any time. This is the distribution of the number in the system for an M/G/1 system. The average number in the system is:

$$\overline{N} = \frac{\lambda^2 \overline{x^2}}{2(1-\rho)} + \rho \tag{7.32}$$

From which it can be concluded that the average loss in a band of size U is:

$$L = \frac{\frac{\lambda^2 \overline{x^2}}{2(1-\rho)} + \rho}{\lambda U} = \frac{\lambda \overline{x^2}}{2U(1-\rho)} + \frac{1}{\mu U} \tag{7.33}$$

It is interesting to notice in RDMA+ that when $U \to \infty$, $L \to 0$.

7.2.4 Network of Nodes

Somewhat surprisingly, the results in the previous sections can be used to approximate the behavior in a network of nodes. This section gives some insight on why the approximations are also valid for a network.

In RDMA, the dynamics in each routing tree are independently of the ones in other routing trees. For this reason, one can take each tree in isolation and analyze it. Additionally, each tree can be seen as a server with multiple incoming links, as in Figure 7.1. There are only a few reasons why both models, the tree model and the node model, are not equivalent. All of them have to do with the number of servers and links that each frame crosses and with the order in which frames are serviced.

The first is that the order of service can vary. In the tree model, sources that are close to the destination are serviced before sources that are distant from the destination. Nevertheless, the order of service is not important to compute both the mean delay and loss, so this difference can be ignored.

The second is that the number of servers that each frame crosses varies. This can be accounted for by analyzing each outgoing link in each tree separately. Then, the results (for delay and loss) may be combined using proper weights (based on the load of each link) to find the end-to-end delay.

The third is that the total propagation delay per frame may also vary according to how many links it crosses. This can be accommodated similarly by computing the average propagation delay taking into account the percentage of frames that will incur the given delay.

Finally, if the service distribution is not Poisson, traffic being input from a node will not present a Poisson distribution when input to the next node. One can approximate the result by assuming that the input to each link is indeed Poisson at each stage of the tree.

One can observe that these are common practices in analyzing network. For example, Reference [Kleinrock 76] makes similar assumptions and can be used to understand how the analysis on a single node can be extended to a network of nodes using the guidelines presented in this section.

7.3 Simulation: Contention for Outgoing Links

The goal of this section is twofold. Firstly, it validates the analysis in Section 7.1 with the simulation of the system depicted in Figure 7.1. Secondly, it compares the behavior of RDMA with PS and CS in a node.

The physical system studied is the one in Figure 7.1 with the following features. There are 10 input links to the node and 1 outgoing link that is shared among the inputs. Each link operates either at 2.4 Gb/s or at 24 Gb/s and transmits ATM cells (each 53 bytes in size). The arrival process is Poisson. The frame size is either deterministic or distributed according to an exponential distribution. For the RDMA simulations, the cycle period is 125 μs. The band size varies in each case. Finally, the network topology is such that there are 5 bands in each cycle.

7.3.1 RDMA++

The purpose of the study is to show how closely the approximated and simulated RDMA++ mean frame delays match each other. As expected, the approximation works best for low to medium network loads in all experiments conducted.

Figure 7.7 illustrates mean delay when the frame size is fixed at 53 bytes, and the maximum link rate is 2.4 Gb/s. For this figure and the following, the study varies the input load up to 90% of its maximum capacity to avoid unstable behaviors as $\rho \rightarrow \frac{U}{U+V}$. The input load axis is measures in thousands of frames per second (Kframes). For example, at 73.584906 Kframes/s, the nominal input rate is 31.2 Mb/s in each input link. Because there are 10 input links, the aggregate input to the outgoing link is 312 Mb/s or 13% of the maximum rate of 2.4 Gb/s. Notice that, because the band size is 20% of the cycle, the maximum aggregate input rate to the outgoing link must be less than 20% or else the system will be unstable.

In Figure 7.8, the maximum link rate is 2.4 Gb/s, and the frame size is exponentially distributed with mean 53 bytes.

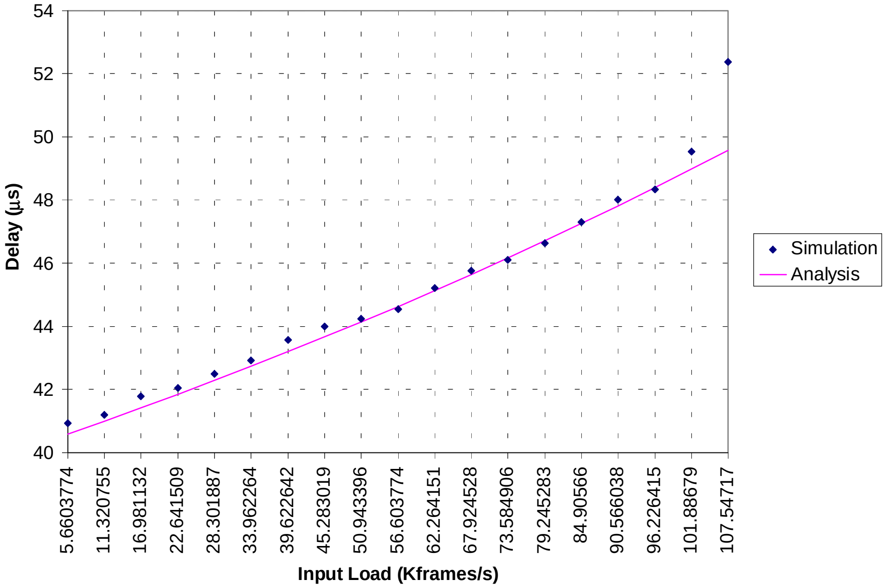


Figure 7.7: Mean frame delay (in µs) for Poisson arrivals and deterministic service under RDMA++ (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 25 µs, the cycle period 125 µs)

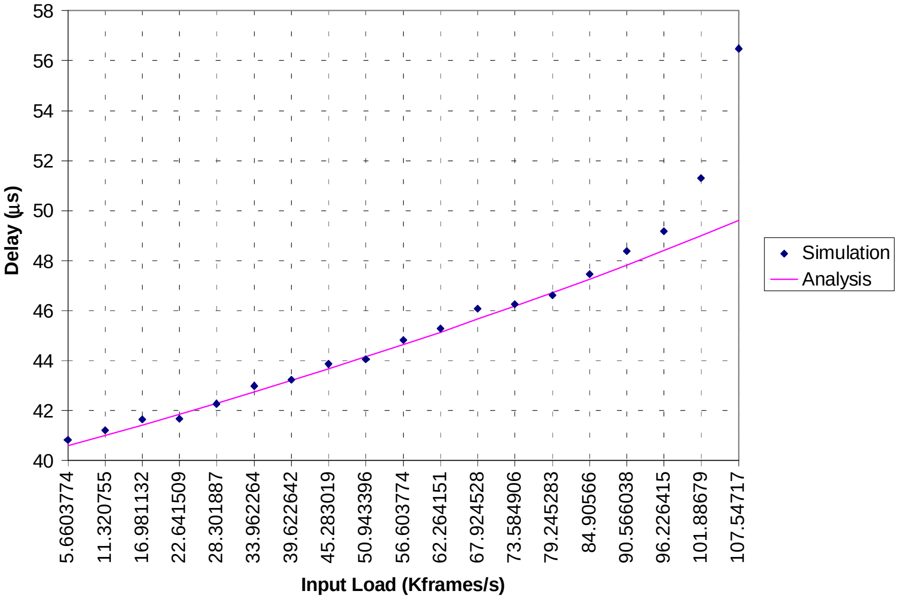


Figure 7.8: Mean frame delay (in µs) for Poisson arrivals and Poisson service under RDMA++ (the mean frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 25 µs, the cycle period 125 µs)

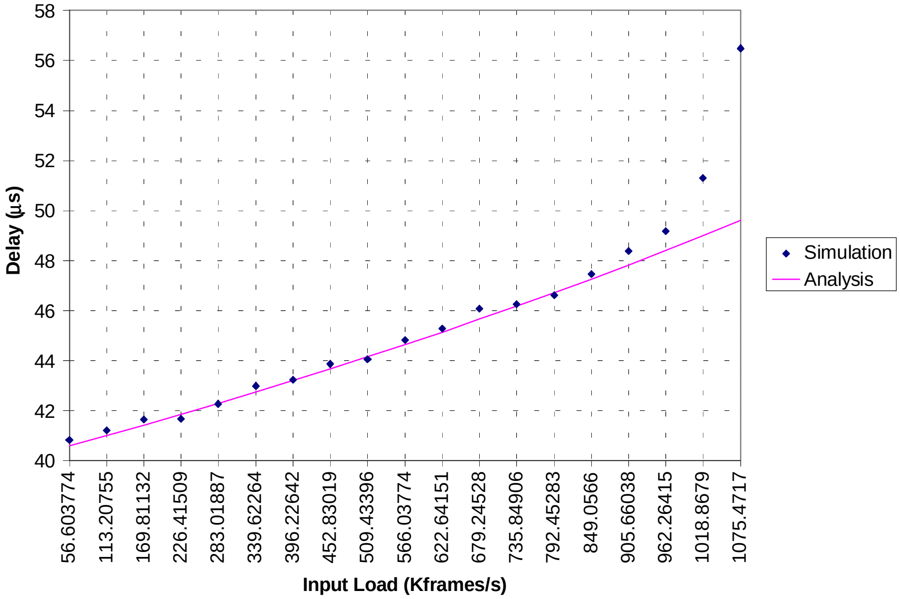


Figure 7.9: Mean frame delay (in µs) for Poisson arrivals and Poisson service under RDMA++ (the mean frame size is 530 bytes, the link speed 24 Gb/s, the band size 25 µs, the cycle period 125 µs)

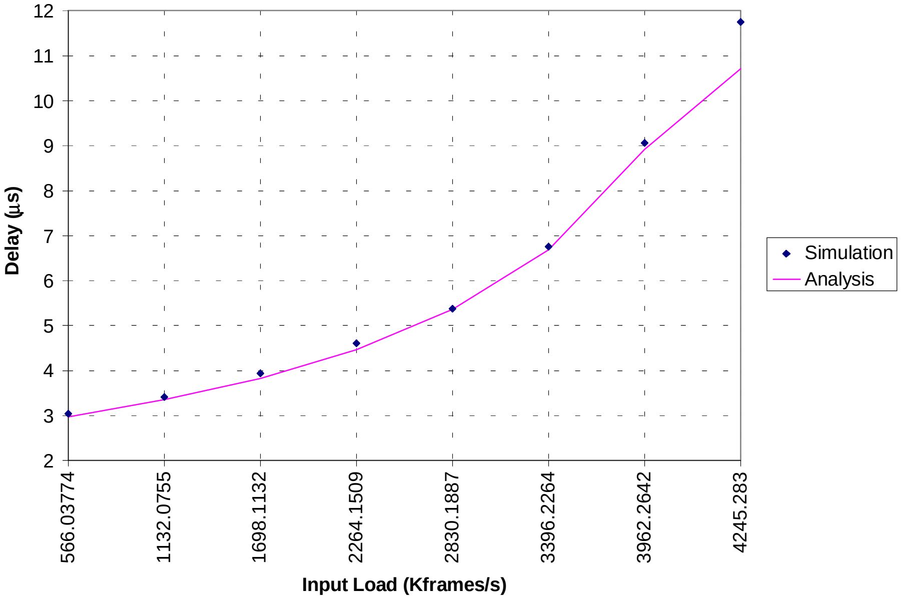


Figure 7.10: Mean frame delay (in µs) for Poisson arrivals and deterministic service under RDMA++ (the frame size is 530 bytes, the link speed 24 Gb/s, the band size 100 µs, the cycle period 125 µs)

Figure 7.9 repeats an experiment similar to the one in Figure 7.8 but with a faster link speed of 24 Gb/s and a mean frame size of 530 bytes. One possible interpretation for this experiment is that arrivals are bursty with mean burst size of 10 ATM cells. The figure displays the mean delay per burst.

Figure 7.10 depicts an experiment similar to the one in Figure 7.9 but with a bigger band size of 100 μs, or 80% of the cycle. Again, one can see the same trend as in the other experiments.

Finally, Figure 7.11 depicts the case of a CS system with fixed size ATM cells and 2.4 Gb/s maximum link rate. Each circuit lasts 2.5 μs and the cycle period is 125 μs. One can see that again the approximation matches the simulation for low to medium loads.

7.3.2 RDMA-

Figure 7.12 depicts the loss rate in RDMA- when the cycle period is 125 μs, the band size 25 μs, the arrival is Poisson within a band, and the frame size fixed at 53 bytes. Similar results can be obtained for exponentially distributed frame sizes. The input load axis indicates the ratio of the total input load in each input link during a band. For example, the ratio 0.8 indicates that the input links are transmitting at 1.92 Gb/s. One can see that the simulation closely matches the analysis and that the maximum loss in the system is 50%, as expected.

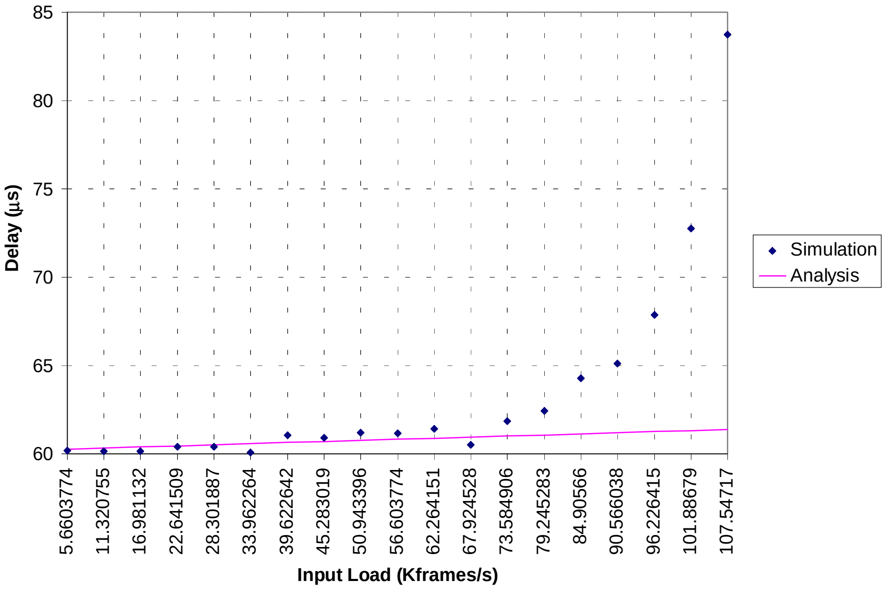


Figure 7.11: Mean frame delay (in µs) for Poisson arrivals and deterministic service under CS (the frame size is 53 bytes, the link speed 2.4 Gb/s, the circuit size 2.5 µs, the cycle period 125 µs)

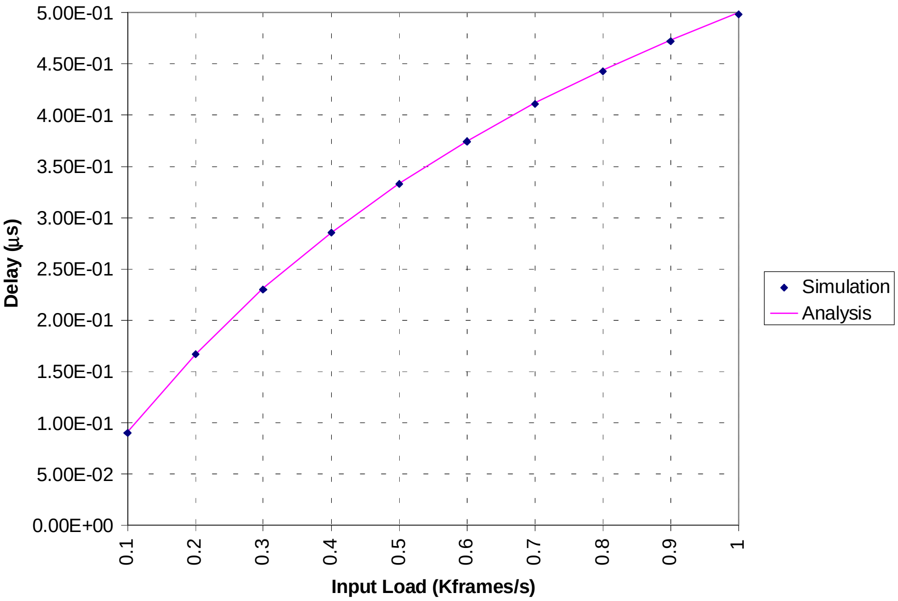


Figure 7.12: Mean loss rate for Poisson arrivals and deterministic service under RDMA- (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 25 µs, the cycle period 125 µs)

7.3.3 RDMA+

Figure 7.13 illustrates the mean delay for RDMA+ with Poisson arrivals within a band. One can see that the analysis is a good approximation of the delay in the system, especially at low loads.

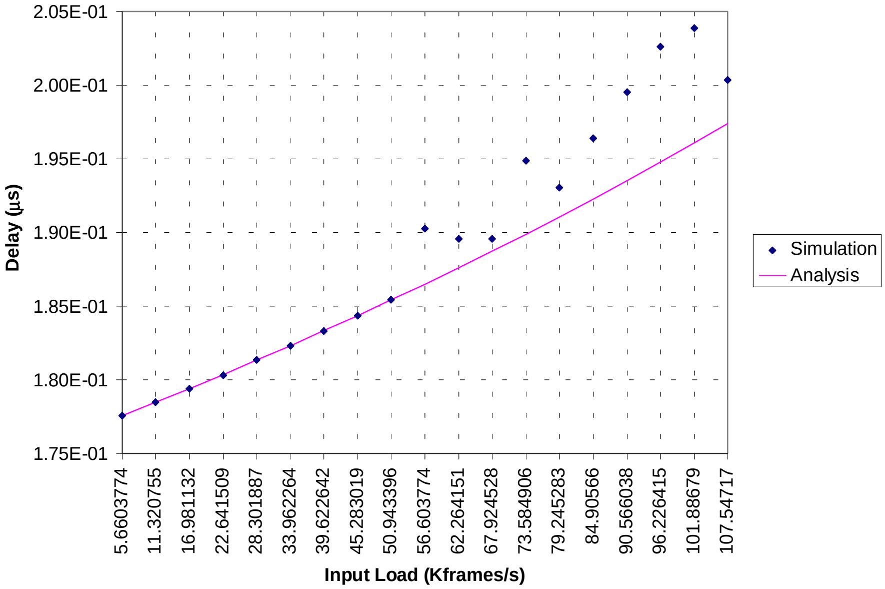


Figure 7.13: Mean frame delay (in µs) for Poisson arrivals and deterministic service under RDMA+ (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 25 µs, the cycle period 125 µs)

7.3.4 Comparison

The goal of this section is to compare RDMA++ performance with PS and CS.

The CS simulation works as follows. There is a cycle of 125 µs within which end-to-end connections can be allocated. Each connection lasts 2.5 µs. This ensures a fair comparison with RDMA++ where a band of size 25 µs is shared among 10 sources.

CS is expected to perform always worst than RDMA++ because the time waiting for an end-to-end connection to arrive is bigger than the time waiting for a band. In the particular case being simulated, for instance, each band lasts 25 μs and is repeated every 125 μs while each end-to-end circuit lasts 2.5 μs and is also repeated every 125 μs.

The PS simulation works as follows. There is a processor operating at the same nominal speed of one of the input links. This means, for example, that if 50 instructions are necessary to process each ATM cell, the simulated processor operates at 283 MIPS, which is an optimistic assumption. Additionally, there could be contention for processor use from frames crossing the node to other destinations. Such interference does not happen in RDMA or CS. It is reflected in the simulation by making all input traffic from each link incur some delay to be processed at the node. The number of possible destinations for each input frame is 10, and so the overall input traffic to be processed is 10 times larger than the traffic that will eventually follow the correct output link in Figure 7.1.

It is expected that PS will perform better than RDMA until its processing resources are exhausted at which point RDMA will outperform PS.

The input load axis is actually the load from each input link to the output link. For the PS simulation, the input load to the processor from each input link is 10 times this value. The reason is that the PS simulation generates traffic to all 10 possible destinations and all frames need to cross the processor to be switched.

Figure 7.14 and Figure 7.15 show the mean delay for the three systems. In the first, the link speed is 2.4 Gb/s and the frame size fixed at 53 bytes. In the second, the link speed is 24 Gb/s and the frame size exponentially distributed with mean 530 bytes. As it can be verified, both experiments are in accordance with the expectations.

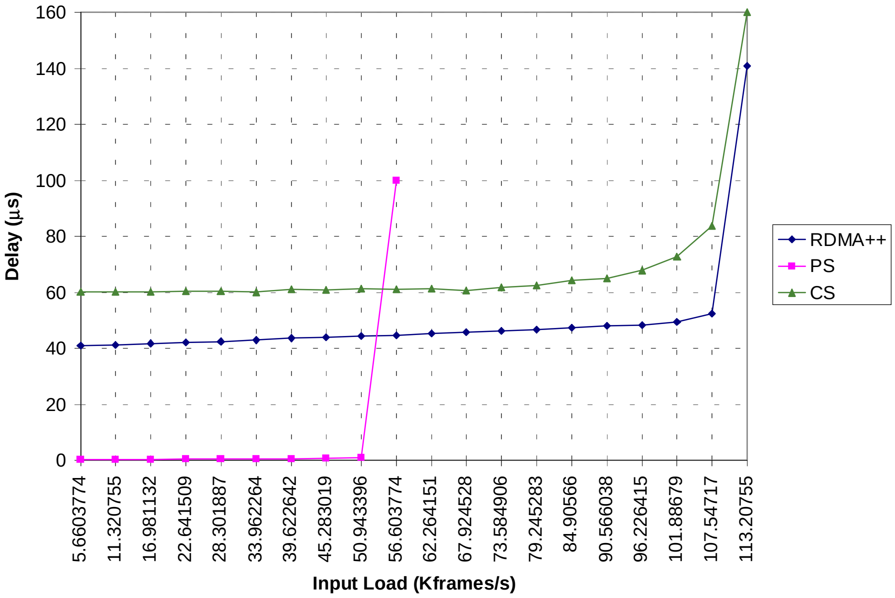


Figure 7.14: Mean frame delay comparison (in µs) for Poisson arrivals and deterministic service (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 25 µs, the cycle period 125 µs)

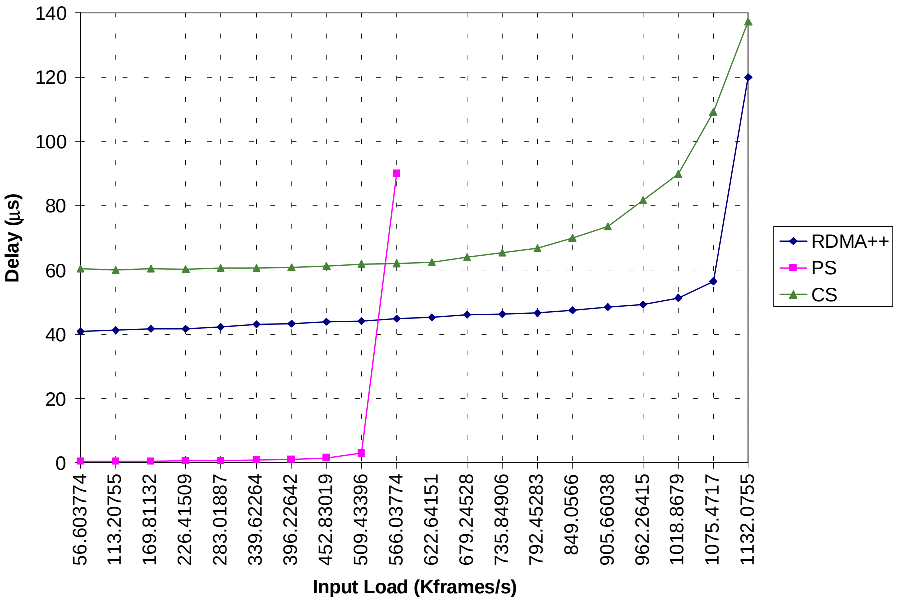


Figure 7.15: Mean frame delay comparison (in µs) for Poisson arrivals and Poisson service (the frame size is 530 bytes, the link speed 24 Gb/s, the band size 25 µs, the cycle period 125 µs)

Notice that PS becomes unstable in both simulations when the input load in each link is about 10% of its maximum capacity. At this point, the traffic to the output link in Figure 7.1 is 566.03774 Kframes/s (as depicted in Figure 7.14 and Figure 7.15) or 1% of the maximum capacity. The remaining 9% are directed to other destinations. The overall input to the processor from the 10 links is 100% of the processor capacity. That is, only 10% of the traffic is directed to the output link, but the remaining 90% still consume processor cycles to be switched. CS and RDMA++ become unstable at 2% of the input link capacity at which point the aggregate traffic for the output link is 20%, which is also the portion of the cycle allocated to the output link.

7.4 Simulation of Networks

This section extends the performance studies of RDMA to a network of nodes. As it will be seen, the results are similar to the ones obtained in the previous section.

The topology studied is depicted in Figure 7.16. It is a symmetric configuration that allows the overlapping of 3 non-interfering trees. An example of 3 non-interfering trees to destinations 1, 6 and 8 is depicted in Figure 7.17. These destinations thus can share a band. Two additional bands are sufficient to serve the 6 trees of the other destination nodes.

The simulation model works as follows. Each node generates ATM cells according to a Poisson process. Destinations are assigned to cells according to a uniform distribution. The link speed is 2.4 Gb/s, that is, each cell needs 177 ns to be transmitted. The cycle period is 125 µs. The propagation delay in each link is negligible (equivalent to 1 cell transmission delay). The bands to all destinations are of the same size. Each cell waits for the proper destination band at the source nodes and then moves through the network down the respective tree. The goal is to give a comparison of PS, CS, and RDMA++ performance.

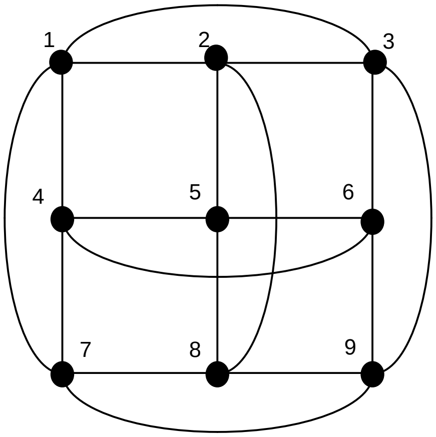


Figure 7.16: Simulated network topology

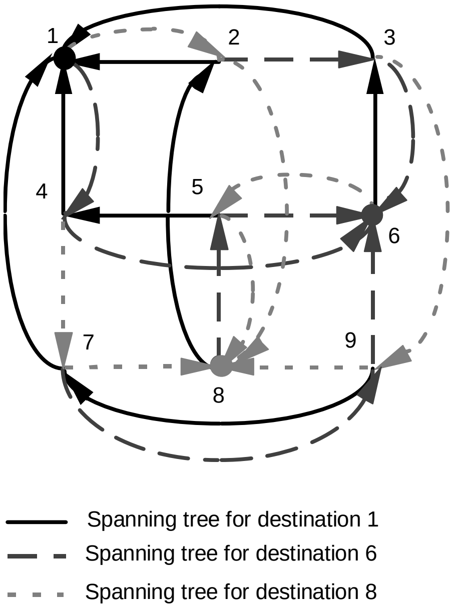

1
2
3
4
5
6
7
9
8
Spanning tree for destination 1
Spanning tree for destination 6
Spanning tree for destination 8

Figure 7.17: Allocation of trees in one band

The PS and CS simulations operate similarly to the ones in Section 7.3, use the same trees allocated for RDMA++ (see Figure 7.16), and cut-through technology [Kermani 78].

The CPU at each switch operates at the same rate of one input link. This means, for example, that if 50 instructions are necessary to process each ATM cell, the simulated processor operates at 283 MIPS, which is an optimistic assumption.

The simulation was run for periods where 10,000 ATM cells per input node were generated. After each of these periods, all statistics were saved and reset. Two RDMA++ experiments were conducted. In the first, all traffic had the same priority (RDMA++<c>). In the second, priority traffic was generated as follows. Each band was equally partitioned into priority sub-bands, one for each input node (RDMA++<p>).

Figure 7.18 depicts the mean packet delay (in µs) for the experiments. The input traffic load is given as a percentage of the 2.4 Gb/s maximum input rate at each node. PS has a steady performance until the input load 50% saturates the CPU capability at the nodes with delays growing unbounded. CS has a similar behavior. The unstable point is 30%. The reason is that this CS simulation allocates all nodes in its path (including the destination). It becomes unstable before RDMA++ because the latter allocates the destination node for the whole tree and not for a single circuit. An alternative CS operation could potentially be improved to allocate only links (and not nodes). In this case, the comparison is the same reported in Figure 7.11.

RDMA++ has a stable performance. Both RDMA++<c> and RDMA++<p> have the same mean packet delay characteristics, as expected from queueing analysis [Kleinrock 75], and thus overlap in the figure. The “Priority” curve plots the mean delay for priority traffic generated for the RDMA++<p> experiment. Priority was assigned randomly to ATM cells according to a uniform distribution. Priority traffic was scheduled to access the network during its priority band, thus not incurring admission delays. The delay incurred by the priority traffic is only the propagation delay and contention with other cells scheduled at the beginning of the priority band.

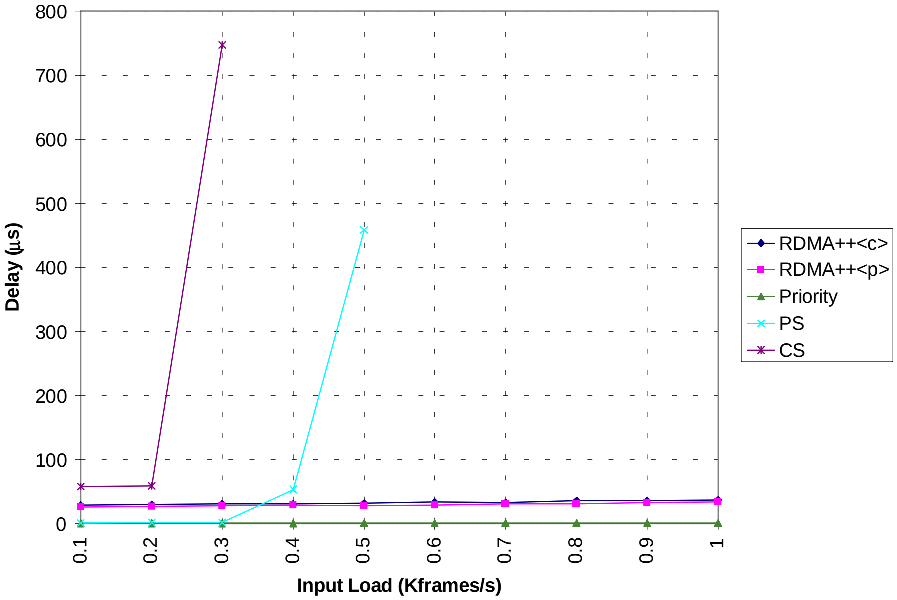


Figure 7.18: Mean frame delay comparison (in µs) for Poisson arrivals and deterministic service (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 41.67 µs, the cycle period 125 µs)

Figure 7.19 shows the network behavior when sources generate bursty traffic according to an on/off model, where the on and off periods are geometrically distributed in the number of cells. The mean on period is 10 ATM cells.

Figure 7.20 displays a multimedia experiment. Source 9 sends motion picture frames to Destination 1. All other sources send normal data traffic generated according to a Poisson process at the load specified in the load axis. The video traffic is scheduled to be generated during the source's priority band, which is of size 10 cells every 125µs cycle. As it can be seen, the network provides high-quality service to the video source while non-QoS demanding traffic proceeds normally. The isolation of both traffic types can be accomplished by simple band tuning (in microseconds).

Figure 7.21 shows the comparison of analysis and simulation results for the RDMA++ mean packet delay. As it can be seen, the results are in good agreement.

One can observe in Figure 7.17 that each tree has 4 links arriving at the final destination. Additionally, the final destinations in the simulation did not incur transfer delays. For this reason, one of the trees seen as a server has a throughput 4 times larger than the transmission rate at each link. This fact must be input into the formulas of the analysis by making the server 4 times faster than the maximum links rate. Additionally, the maximum load to any of the trees is 25% (and thus low). This is the reason why the analysis (that approximates better the system for low loads) closely matches the simulation results.

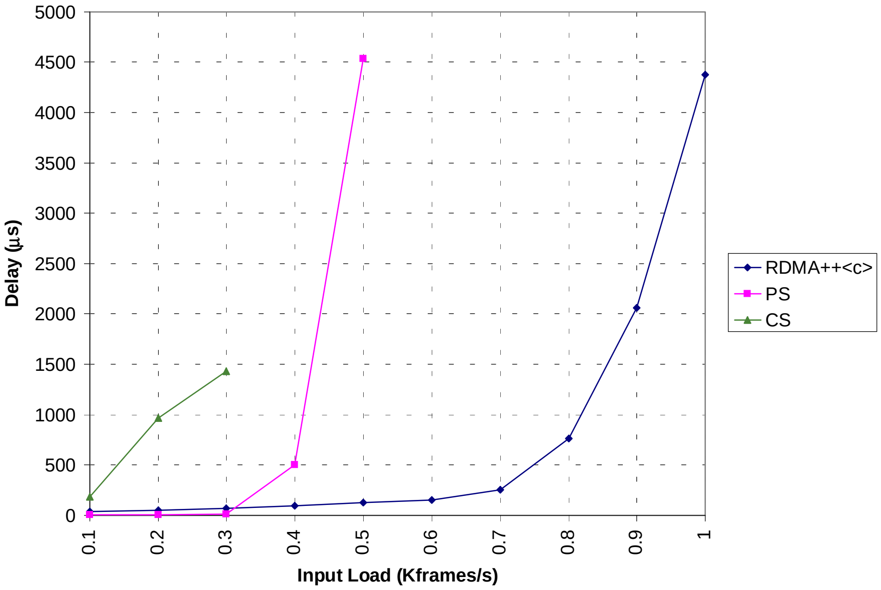


Figure 7.19: Mean frame delay comparison (in μs) for Poisson arrivals and Poisson service (the frame size is 530 bytes, the link speed 2.4 Gb/s, the band size 41.67 μs, the cycle period 125 μs)

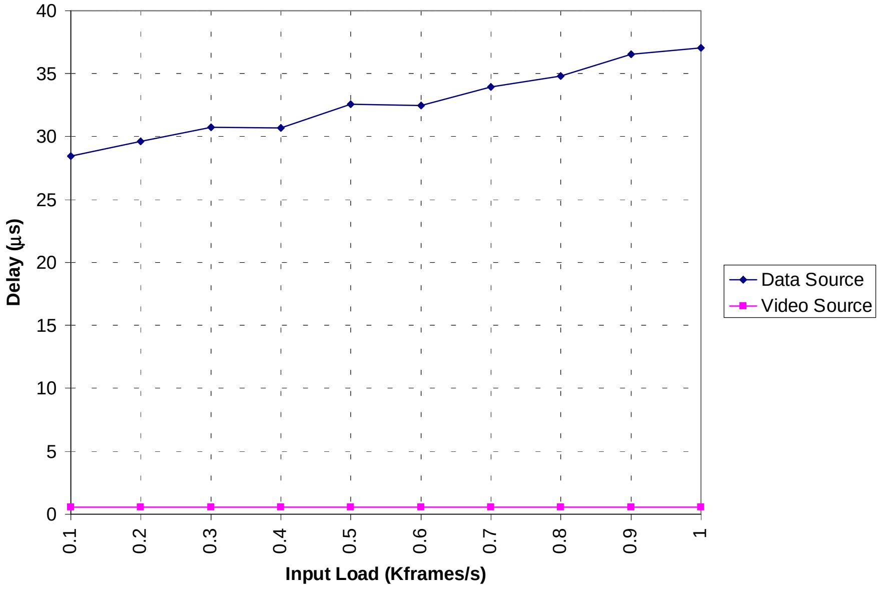


Figure 7.20: Mean frame delay comparison (in µs) for Poisson and deterministic arrivals and deterministic service under RDMA++ (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 41.67 µs, the cycle period 125 µs)

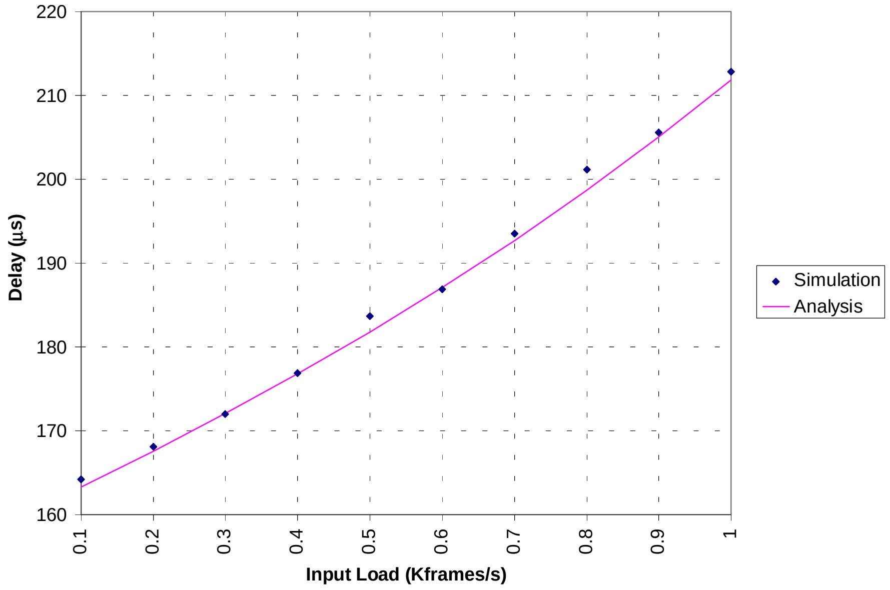


Figure 7.21: Mean frame delay (in µs) for Poisson arrivals and deterministic service under RDMA++ (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 41.67 µs, the cycle period 125 µs)

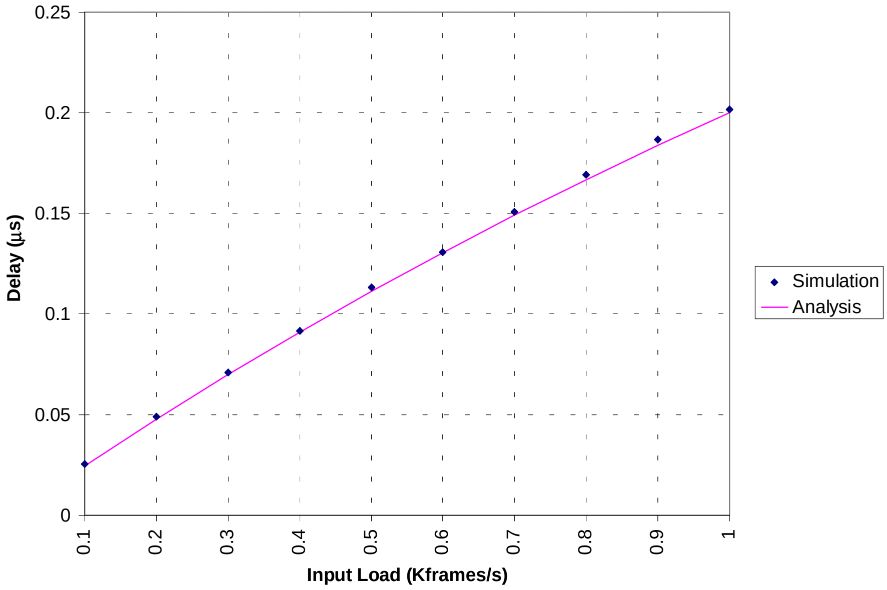


Figure 7.22: Mean loss rate for Poisson arrivals and deterministic service under RDMA- (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 41.67 µs*, the cycle period 125* µs*)*

One can observe in Figure 7.17 that each tree has 4 links arriving at the final destination. Additionally, the final destinations in the simulation did not incur transfer delays. For this reason, one of the trees seen as a server has a throughput 4 times larger than the transmission rate at each link. This fact must be input into the formulas of the analysis by making the server 4 times faster than the maximum links rate. Additionally, the maximum load to any of the trees is 25% (and thus low). This is the reason why the analysis (that approximates better the system for low loads) closely matches the simulation results.

Figure 7.22 compares the simulation and analysis results for the RDMA- mean packet loss rate. Each tree is active only 1/3 of the cycle. Thus the input load in the analysis formulas varies between 0 and 1/3 . This is why the loss rate never reaches the 50% loss upper bound.

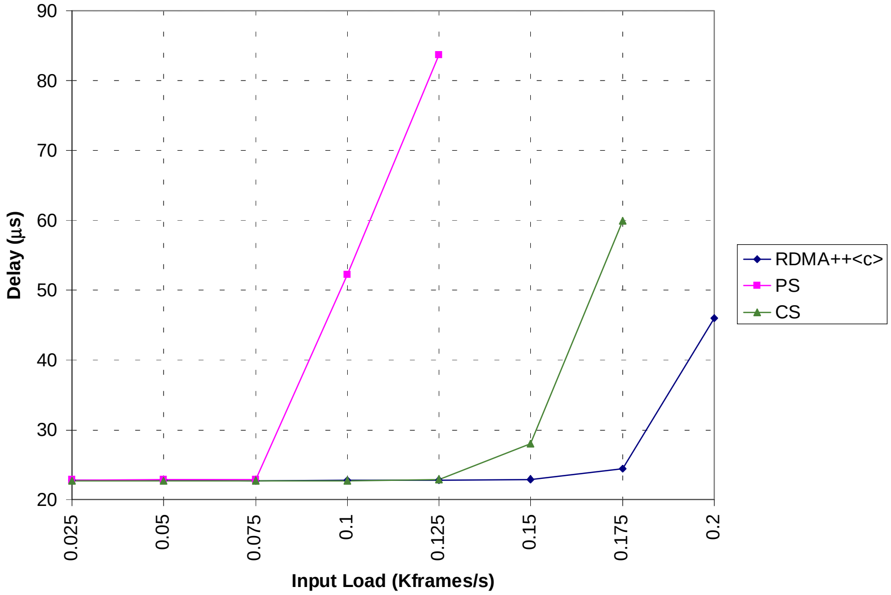


Figure 7.23: Mean frame delay comparison (in µs) for Poisson arrivals and deterministic service (the frame size is 53 bytes, the link speed 2.4 Gb/s, the band size 17.8571 µs, the cycle period 125 µs)

Finally, Figure 7.23 depicts the mean delay when the NSF T3 backbone network operates under RDMA++. The propagation delays are approximated since the exact measures are not available. The link speeds were upgraded to 2.4 Gb/s. The CPU speed was 100 MIPs for the PS simulation and some 50 instructions were necessary to process each ATM cell. It is important to notice that the topology of the NSF backbone is not particularly suitable for RDMA since only two trees can coexist in each band. Nevertheless, the performance advantage of RDMA is clear in the figure.

An important observation that can be derived from the experiment in Figure 7.23 is that, when applied to wide area networks, the time incurred waiting for a particular band is negligible when compared to the propagation delays. For instance, the waiting time for the band in the simulated NSF backbone is at most 125µs (a complete cycle), but the cross-country propagation delay is of the order of 30 ms (240 times larger). Thus, the time waiting for a band in RDMA++ is a negligible component of the total frame delay, which in this case is very close to the PS frame delay for low loads.

7.5 Future Work

The initial performance assessment of RDMA in this chapter concentrated on understanding basic RDMA performance and on comparing it with PS and CS. The next step is to study policies to allocate RDMA bands and to study their performance. One possible outcome of such study, given specific demands, is the best allocation policy and what is the performance that it can deliver.

The analysis in this chapter had to settle for approximations due to the difficulties in finding closed-form solutions. A natural extension for such analysis is to use mechanisms that rely on numerical algorithms to resolve expressions numerically. Potentially, the same mechanisms could be used to analyze higher moments of the delay and the loss in the network.

7.6 Summary

This chapter provided initial analysis of mean RDMA delay and loss. The goal of the analysis was to understand RDMA performance and compare it with PS and CS.

The initial step was to provide an analytical understanding of RDMA behavior with Poisson arrivals on a single node. A closed form solution was found for RDMA- with generic frame size distribution. It was shown that the loss rate will never exceed 50%. The mean delay for RDMA++ was approximated for low to medium input loads. The mean delay for RDMA+ could be approximated as well. The loss rate in RDMA+ was shown to be negligible when the band size is large compared to the mean frame size.

All the analytical results were validated through simulations. The simulations were also used to compare RDMA performance with PS and CS in two network topologies. It was found that PS has better performance for low loads. When the load in the system saturates the frame header processors, RDMA outperforms PS. It is important to notice that the PS simulation did not incur admission control delays that may deteriorate PS performance. Additionally, it was shown that, if the propagation delay in the network is large, the time waiting for bands in RDMA is negligible. In such scenario, the final end-to-end delay is comparable to PS even for low system loads. RDMA always outperforms CS because the mean time waiting for a band is smaller than the mean time waiting for a circuit. Additionally, the destination is allocated to a tree and not a circuit, which has the potential of further increasing RDMA performance when compared to CS.

Chapter 8

Future Work in Isochronets

8.1 Introduction

Isochronets present several open challenges. Some important open problems are natural extensions of initial findings reported in this thesis. They were presented at the end of previous chapters. This chapter will describe other important future directions in Isochronets.

8.2 Large Isochronets

8.2.1 The Problem

How to design Isochronets with large number of nodes? If the number of nodes in the network is very large, it may be necessary to partition the clock cycle into many bands. As a consequence, each band may become very small. Small bands may under-use network resources and increase the time waiting for bands. For example, if the network contains 1,000 nodes and the network topology only allows two trees per band, the mean band duration with a 125 µs cycle is only 125 ns. Links operating at 1 Gb/s take 177 ns to transmit an ATM cell, more time than the mean band size. Additionally, the average time waiting for a band is more than 62 µs.

8.2.2 Preliminary Solution Overview

The proposed solution is to cluster nodes and use Isochronets both within and between clusters. For example, a two level solution would use Isochronets among the nodes in the same cluster (*in-cluster Isochronet*) and between clusters (*inter-cluster Isochronet*), as depicted in Figure 8.1. An illustrative Isochronet network is shown for Cluster E. It contains one or more gateway machines to other clusters, that is, machines that contain links connected to gateways in other clusters.

The following is a typical scenario to send frames from a cluster. In Figure 8.1, let us assume that a node *n* in Cluster E needs to send a frame. If the destination also resides in Cluster E, *n* uses the in-cluster Isochronet band within Cluster E to the destination. Otherwise, *n* sends the frame to the gateway machines *g* in Cluster E. Gateway *g* parses the frame header and schedules its transmission to the destination cluster using the inter-cluster Isochronet. Once the frame reaches the final cluster, the cluster gateway parses its header and then identifies the ultimate destination machine. It schedules the frame for transmission within the in-cluster Isochronet band to the destination machine.

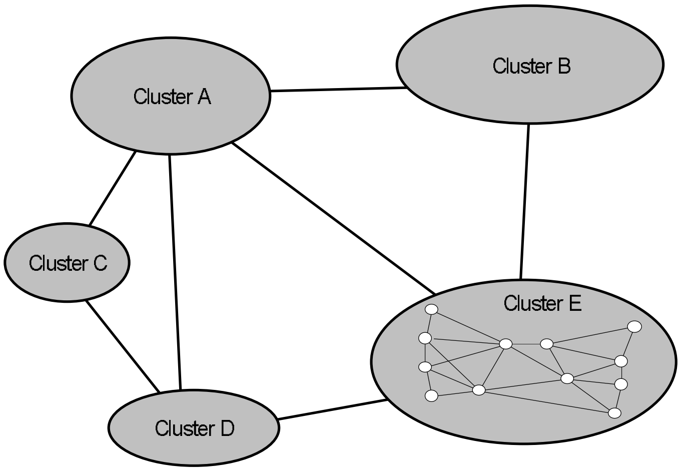


Figure 8.1: Two level hierarchical clustering of Isochronets

This architecture can be extended to multi-cluster hierarchies, where, in each level, Isochronets would run within a cluster and between the clusters.

This solution may offset many of the Isochronets advantages. For example, frame processing is necessary at gateways. The solution can be seen as a compromise between providing advantages of Isochronets on one hand and dealing with large networks on the other hand. Further study must be pursued to best implement a balanced compromise between these goals. The following are a few open issues regarding the proposed solution:

- *How to cluster the machines in order to minimize frame processing?* On the one extreme, one cluster will imply no frame processing while, on the other extreme, too many clusters will imply considerable processing at gateways.
- *How to assign and resolve addresses?* Gateways need to map addresses into clusters and machines efficiently. Nevertheless, it might be necessary to maintain addresses compatible with other networks such as, for example, Internet.
- *How to minimize multi-protocol processing?* One of the important aspects of Isochronets is the absence of multi-protocol adaptation. On the one extreme, an internal uniform frame structure could ease frame parsing at gateways. At the other extreme, multiple protocol structures may complicate gateway processing.

8.3 Reliability in Isochronets

8.3.1 The Problem

How to overcome link and node failures? Link failures in Isochronets will disrupt routing trees and thus disable some connections. Node failures will block all connections passing through them. In such scenario, new routing trees need to be dynamically setup to overcome the disruptions.

8.3.2 Preliminary Solution Overview

Isochronets are particularly suitable for duplication techniques. One possible technique to ensure reliability is to allocate multiple trees to the same destinations. Different trees to the same destination are assigned different bands. That is, each destination would have multiple distinct routing trees assigned to different bands. Also, the routing trees would be allocated so that a pre-defined number of link (or node) failures would not disrupt all the routing trees to any destination. In this fashion, the network would still operate to all destinations by using the band that contains a valid routing tree.

Figure 8.2 illustrates two non-interfering routing trees to the same destination. In a more general case, some of the links may be shared. The idea is that, if each tree is assigned to a different band, the destination is reachable using both bands. If one of the links or nodes fails, it may still be possible to reach the destination using one of the bands.

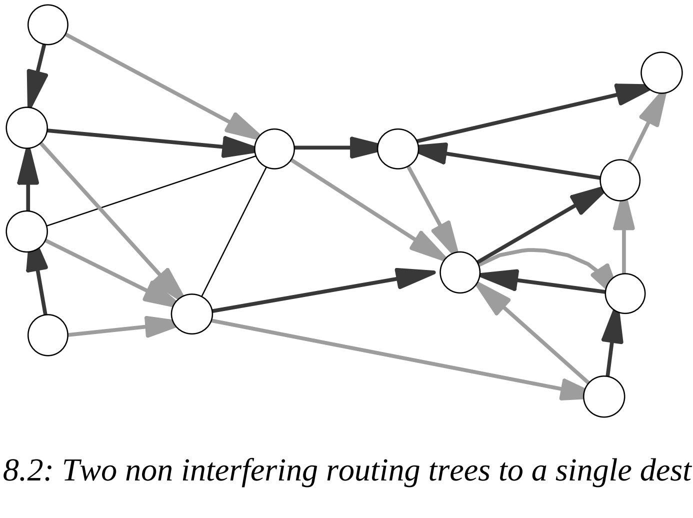


Figure 8.2: Two non interfering routing trees to a single destination

The goal is to find allocations such even if *k* links or *k* nodes fail, there is still a path from all sources to a given destination in some band.

Bands will contain two kinds of trees. The original trees to each destination and backup trees that may or not be used. If a band consists only of backup trees, its size may be shrunk to 0. When failures occur, it can progressively increase to overcome the failure.

The following are a few open issues regarding the proposed solution:

- *How should the trees be allocated?* This problem generalizes the tree allocation problem in Chapter 4 to cover all possible *k* link or node failures.
- *Given a generic network, what are the maximum node and link failures that may be supported?* This is an indication of how reliable the topology is.
- *What is the minimal number of bands that is needed to cover all possible k link or node failures?* The more bands are necessary, the more difficult the problem of deciding band sizes for original and backup trees becomes.
- *How should the band sizes be affected by failures?* A failure in a link may direct considerable traffic to other bands.

8.4 Reconfiguration

8.4.1 The Problem

How should band and tree allocations change when routes change? Routes may change due to the introduction or removal of nodes or links. Also, it is possible to change the routes with alternative more efficient options.

8.4.2 Discussion

Upon the introduction of new nodes or links in the network, it is necessary to recompute tree and band allocations. The ideal solution would involve incremental updates to existing allocations.

8.5 Topology Design

8.5.1 The Problem

Suppose one is to design a network that will operate according to the Isochronets architecture. *What is the most suitable topology taking into account multiple constraining issues such as cost, node locations, etc.?*

8.5.2 Discussion

Many techniques have been applied to generic optimization problems as the one proposed here. One of the difficult questions is how to define the optimization criteria that would best accommodate each scenario.

Chapter 9

Conclusions

This dissertation addressed the problem of building a novel architecture for High-Speed Networks (HSNs) that solves two fundamental limitations of existing architectures:

- Scaling with respect to link speeds;
- Strict guarantees of the Quality of Service (QoS) in the communications.

Isochronets is the architecture that solves both limitations. The main advantages of Isochronets are:

- *Support of end-to-end transport of multi-protocol frames.* Isochronets do not rely on nor need to adapt to any frame structure because frames do not need to be processed inside the network to be switched. Multiple frame structures (for example, IP, ATM, etc.) can coexist in the network without adaptation.
- *Support of tunable guaranteed QoS of end-to-end transport.* Through band allocation, sources can be given priority in a tree. When transmitting in this mode, traffic has guaranteed bandwidth and does not experience any jitter, loss, or delay introduced by contention with other sources. Consequently, QoS is strictly guaranteed through tunable bands.
- *Support of asynchronous, synchronous and isochronous traffic streams.* Synchronization signals are forwarded to network interfaces when bands begin and can

be used to synchronize traffic motions. Nevertheless, within a band, motion is asynchronous. Depending on the kind of service a band provides (priority or contention), the end-to-end motion may attain every synchronization need.

- *Minimization of processing and queueing latency in the transport path.* Switching is accomplished by re-configuring nodes over time. Frames are not processed to be switched. As a consequence, nodes do not need to process frames at incoming high-speed rates. Rather, nodes need only re-configure themselves periodically. Additionally, queueing only happens to access outgoing links and can be finely controlled, and even avoided completely by assigning priorities to sources in a band.

The following challenges in designing, building, and analyzing Isochronets were addressed in this work.

1. Initial protocols for tree allocation, band allocation, and band synchronization were developed.
    - All protocols can be implemented off-line and execute at slow speed relative to transmissions.
    - The tree allocation problem is solved by an exhaustive search algorithm that searches all possible combinations of non-interfering trees to find the one that leads to the minimal number of bands. This is a reasonable solution because tree allocations happen only when the network topology changes.
    - The band synchronization problem is solved by adjusting the link propagation delays in the network to make the final delay a multiple of the clock period.

  This fact trivializes synchronization because all that is necessary, after link delay adjustment, is to start and end each band exactly at the same moment within each cycle.

- The band allocation problem sizes the bands by using a dynamic algorithm based on the Least Recently Used (LRU) memory allocation policy. It operates by shrinking band sizes for destinations that are not accessed frequently and then increasing band sizes of destinations that need higher demands.

2. A novel scheme for real-time service provision at periphery protocol stacks was developed.

- The novel Loosely-synchronous Transfer Mode (LTM) was created to support real-time services. It issues loosely-synchronous signals to periphery nodes indicating current network status, that is, destinations that are reachable and their associated QoS. As a consequence:
    * Sources and destinations are synchronized with each other and with the network using the loosely-synchronous signals.
    * Multiple protocol frames can be transferred without fragmentation and reassembly because there is no pre-defined frame structure in the communication.
    * QoS can be controlled and guaranteed because the loosely-synchronous signals coordinate interactions between sources to avoid contention at intermediate nodes.
- The Synchronous Protocol Stack (SPS) was defined. SPS is any stack ex-

tended with the loosely synchronized signals provided by the LTM. An SPS forwards LTM synchronization signals through its layers up to the applications. In the SPS:

    * Existing protocol stacks are supported unchanged.
    * Existing stacks are extended with novel source and destination synchronization to support a variety of application synchronization needs.

3. The feasibility of designing and implementing an electronic and an optical Isochronet switch (Isoswitch) was demonstrated.
    - An electronic Isoswitch has been designed and implemented. It has four input and four output ports each operating at 1 Gb/s. The design addressed the following challenges:
        * The design is scaleable with respect to number of channels and link speeds.
        * The implementation is easy to integrate with existing hardware.
        * The Isoswitch can provide novel LTM synchronization services to periphery protocol stacks.
    - An interface card between the Isoswitch and a Sun SPARC 1 machine has been built. The card has a nominal throughput of 22 Mb/s. The card signals the beginning of band to the SPARC processor and can thus be directly used to implement LTM and SPS.
    - A preliminary design of an all-optical switch implementation is proposed. The design proposed uses Wavelength Division Multiplexing (WDM) to allocate one wavelength for each band. The design has a few advantages:

    - No conversion of en-route signals to the electronic domain, which makes possible implementations potentially at hundreds of terabits per second.
    - Fewer wavelengths are necessary than in pure WDM, namely one per band. In the worst case, the number of necessary wavelengths is the number of destinations in the network.

4. An initial performance study of Isochronets was developed.

    - An analytical model was developed to study mean delay and loss in RDMA. It was assumed that frame arrivals happen according to a Poisson process.
        - The mean loss in RDMA- was approximated by an approximate closed-form expression. It was used to show that the maximum loss in RDMA- is bound to 50%.
        - The mean delay under RDMA++ when the input load is low to medium was approximated by a closed-form expression.
        - The mean delay in RDMA+ was approximated by a closed-form expression. The loss rate was shown to be negligible if the band size is large compared to the mean frame size.
    - Simulation studies validated the analytical expressions and compared RDMA end-to-end delay performance with Packet Switching (PS) and Circuit Switching (CS). The general performance trend is the following. For low input loads, PS outperforms RDMA and CS because there is no admission delay waiting for a band or circuit. When the load increases, the processor becomes saturated and then RDMA outperforms PS. CS always performs worst than

RDMA because of the penalty incurred waiting for a circuit, which is larger than the one waiting for a band.

Bibliography

[Acampora and Karol 89]
A.S. Acampora and M.J. Karol.
An Overview of Lightwave Packet Networks.
IEEE Network Magazine, vol. 3, no. 1, pp. 29–41, January 1989.

[Adams et al. 87]
G.B. Adams III, D.P. Agrawal, and H.J. Siegel.
A Survey and Comparison of Fault-tolerant Multistage Interconnection Networks.
Computer, vol. 20, no. 6, pp. 14–27, June 1987.

[Ahmadi and Denzel 89]
H. Ahmadi and W.E. Denzel.
A Survey of Modern High-performance Switching Techniques.
IEEE Journal of Selected Areas in Communications, vol. 7, no. 7, pp. 1091–1103, September 1989.

[Amstutz 83]
S.R. Amstutz.
Burst Switching—a Method for Dispersed and Integrated Voice and Data Switching.
In *Proceedings of the International Conference on Communications*, pp. 288–292, Boston, Massachusetts, USA, June 1983. IEEE.

[Behzad et al. 79]
M. Behzad, G. Chartrand, and L. Lesniak-Foster.
Graphs & Digraphs.
Wadsworth International Group, 1979.

[Bertsekas and Gallager 92]
D. Bertsekas and R. Gallager.
Data Networks.
Prentice Hall, Inc., Englewood Cliffs, NJ, USA, Second Edition, 1992.

[Brackett 91]
C.A. Brackett.
Dense Wavelength Division Multiplexing Networks: Principles and Applications.
IEEE Journal of Selected Areas in Communications, vol. 8, no. 6, pp. 948–964, August 1990.

[Broomell and Heath 83]
G. Broomell and R. Heath.

Classification Categories and Historical Development of Circuit Switching Topologies.
Computing Surveys, vol. 15, no. 2, pp. 95–133, June 1983.

[Chao 91] H.J. Chao.
A Recursive Modular Terabit/second ATM Switch.
IEEE Journal of Selected Areas in Communications, vol. 9, no. 8, pp. 1161–1172, October 1991.

[Comer 91] D.E. Comer.
Internetworking with TCP/IP. Volume I: Principles, Protocols, and Architecture.
Prentice Hall, Inc., Englewood Cliffs, NJ, USA, Second Edition, 1991.

[De Prycker 93] M. De Prycker.
Asynchronous Transfer Mode: Solution for Broadband ISDN.
Ellis Horwood Limited, Hemel Hempstead, Hertfordshire, England, Second Edition, 1993.

[Dono et al. 90] N.R. Dono, P.E. Green, K. Liu, R. Ramaswami, and F. Tong.
A Wavelength Division Multiple Access Network for Computer Communications.
IEEE Journal of Selected Areas in Communications, vol. 8, no. 6, pp. 983–994, August 1990.

[Eng et al. 89] K.Y. Eng, M.J. Karol, and Y.S. Yeh.
A Growable Packet (ATM) Switch Architecture: Design Principles and Applications.
In *Proceedings of GLOBECOM*, pp. 1159–1165, Dallas, TX, USA, November 1989. IEEE.

[Federgruen and Green 86] A. Federgruen and L. Green.
Queueing Systems with Service Interruptions.
Operations Research, vol. 34, no. 5, pp. 1161–1172, September–October 1986.

[Feng 81] T.-Y. Feng.
A Survey of Interconnection Networks.
Computer, vol. 14, no. 12, pp. 12–27, December 1981.

[Florissi and Yemini 94] D. Florissi and Y. Yemini.

Protocols for Loosely Synchronous Networks.
In *Proceedings of the 4th International IFIP Workshop on Protocols for High Speed Networks*, pp. 69–83, Vancouver, BC, Canada, August 1994. IFIP.

[Frank and Lyle 90] E.H. Frank and J. Lyle.
SBus Specification B.0.
Sun Microsystems, Inc., 1990.

[Gerla et al. 92] M. Gerla, T.-Y. Tai, and G. Gallassi.
LAN/MAN Interconnection to ATM: a Simulation Study.
In *Proceedings of INFOCOM*, pp. 2270–2279, Florence, Italy, May 1992. IEEE.

[Giacopelli 91] J.N. Giacopelli, J.J. Hickey, W.S. Marcus, W.D. Sincoskie, and M. Littlewood.
Sunshine: a High-performance Self Routing Broadband Packet Switch Architecture.
IEEE Journal of Selected Areas in Communications, vol. 9, no. 8, pp. 1289–1298, October 1991.

[Halsall 92] F. Halsall.
Data Communications, Computer Networks and Open Systems.
Addison-Wesley Publishing Company, Reading, MA, USA, Second Edition, 1992.

[Haselton 83] E.F. Haselton.
A PCM Switching Concept Leading to Burst Switching Network Architecture.
In *Proceedings of the International Conference on Communications*, pp. 1401–1406, Boston, Massachusetts, USA, June 1983. IEEE.

[Huang and Knauer 84] A. Huang and S. Knauer.
Starlite: a Wideband Digital Switch.
In *Proceedings of GLOBECOM*, pp. 121–125, Atlanta, Georgia, USA, November 1984. IEEE.

[Humblet et al. 92] P.A. Humblet, R. Ramaswami, and K.N. Sivarajan.
An Efficient Communication Protocol for High-speed Packet-switched Multichannel Networks.
In *Proceedings of SIGCOMM*, pp. 2–13, Baltimore, Maryland, USA, August 1992. ACM.

[Karlin and Taylor 75]
S. Karlin and H.M. Taylor.
A First Course in Stochastic Processes.
Academic Press, Second Edition, 1975.

[Kermani 78]
P. Kermani.
Switching and Flow Control Techniques in Computer Communication Networks.
PhD Thesis, Technical Report UCLA-ENG-7802, Computer Science Department, University of California Los Angeles, February 1978.

[Kleinrock 75]
L. Kleinrock.
Queueing Systems. Volume I: Theory.
John Wiley & Sons, 1975.

[Kleinrock 76]
L. Kleinrock.
Queueing Systems. Volume II: Computer Applications.
John Wiley & Sons, 1976.

[Le Boudec 92]
J.Y. Le Boudec.
Asynchronous Transfer Mode: a Tutorial.
Computer Networks and ISDN Systems, vol. 24, no. 4, pp. 279–309, May 1992.

[Lee 90]
T.T. Lee.
A Modular Architecture for Very Large Packet Switches.
IEEE Transactions on Communications, vol. 38, no. 7, pp. 1097–1106, July 1990.

[Metzler et al. 92]
B. Metzler, I. Miloucheva, and K. Rebensburg.
Multimedia Communication Platform: Specifications of the Broadband Transport Protocol XTPX.
CIO, RACE Project 2060, 60/TUB/CIO/DS/A/002/b2, September 1992.

[Mills 91]
D.L. Mills.
Internet Time Synchronization: the Network Time Protocol.
IEEE Transactions on Communications, vol. 39, no. 10, pp. 1482–1493, October 1991.

[Mills et al. 90]
D.L. Mills, C.G. Boncelet, J.G. Elias, P.A. Schragger, and A.W. Jackson.
Highball: a High Speed, Reserved-Access, Wide Area Network.

Technical Report 90-9-1, Electronic Engineering Department, University of Delaware, September 1990.

[Murakami 91] G.J. Murakami.
Non-blocking Packet Switching with Shift-register Rings.
Technical Report UIUCDCS-R-91-1711, Department of Computer Science, University of Illinois at Urbana-Champaign, October 1991.

[Nojima 87] S. Nojima.
Integrated Services Packet Network Using Bus Matrix Switch.
IEEE Journal of Selected Areas in Communications, vol. 5, no. 8, pp. 1284–1291, October 1987.

[Ofek 94] Y. Ofek.
Generating a Fault Tolerant Global Clock using High-speed Control Signals for the MetaNet Architecture.
IEEE Transactions on Communications, vol. 42, no. 5, pp. 2179–2188, May 1994.

[Ofek and Yung 90] Y. Ofek and M. Yung.
Principles for High Speed Network Control: Loss-less and Deadlock-freeness, Self-routing and a Single Buffer per Link.
In *Proceedings of the 9th Annual ACM Symposium on Principles of Distributed Computing*, pp. 161–175, Quebec City, Quebec, Canada, August 1990. ACM.

[Oie et al. 91] Y. Oie, T. Suda, M. Murata, D. Kolson, and H. Miyahara.
Survey of Switching Techniques in High-speed Networks and their Performance.
International Journal of Satellite Communications, vol. 9, no. 5, pp. 285–303, May 1991.

[Ott 87] T.J. Ott.
The Single-server Queue with Independent GI/G and M/G Input Streams.
Advances in Applied Probability, vol. 19, no. 1, pp. 266–286, March 1987.

[Pattavina 90] A. Pattavina.
A Multistage High-performance Packet Switch for Broadband Networks.
IEEE Transactions on Communications, vol. 38, no. 9,

pp. 1607–1615, September 1990.

[Ramaswami 93] R. Ramaswami.
Multiwavelength Lightwave Networks for Computer Communication.
IEEE Communications Magazine, vol. 31, no. 2, pp. 78–88, February 1993.

[Roth 92] Charles H. Roth, Jr.
Fundamentals of Logic Design.
West Publishing Company, St. Paul, MN, USA, 1992.

[Schulzrinne et al. 94] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson.
RTP: a Transport Protocol for Real-Time Applications.
Internet-draft, July 18, 1994.

[Sidi et al. 89] M. Sidi, W. Liu, I. Cidon, and I. Gopal.
Congestion Control through Input Rate Regulation.
In *Proceedings of GLOBECOM*, pp. 1764–1768, Dallas, Texas, USA, November 1989. IEEE.

[Stern 90] T.E. Stern.
Linear Lightwave Networks: How Far can They Go?.
In *Proceedings of GLOBECOM*, pp. 1866–1872, San Diego, California, USA, December 1990. IEEE.

[Stevens 90] W.R. Stevens.
Unix Network Programming.
Prentice Hall, Inc., Englewood Cliffs, NJ, USA, 1990.

[Suzuki 77] M. Suzuki.
Group Theory I.
Springer-Verlag, 1977.

[Tanenbaum 88] A.S. Tanenbaum.
Computer Networks.
Prentice Hall, Inc., Englewood Cliffs, NJ, USA, Second Edition, 1988.

[Tobagi 90] F.A. Tobagi.
Fast Packet Switching Architectures for Broadband Integrated Services Digital Networks.
Proceedings of the IEEE, vol. 78, no. 1, pp. 133–167, January 1990.

[Tobagi et al. 91] F.A. Tobagi, T. Kwok, and F.M. Chiussi.
Architecture, Performance, and Implementation of the Tandem Baynyan Fast Packet Switch.
IEEE Journal of Selected Areas in Communications, vol. 9, no. 8, pp. 1173–1193, October 1991.

[Topolic 90] C. Topolic.
Experimental Internet Stream Protocol: Version 2 (ST-II).
Internet Request for Comments RFC1190, October 1990.

[Turner 86] J.S. Turner.
Design of an Integrated Service Packet Network.
IEEE Journal of Selected Areas in Communications, vol. 4, no. 8, pp. 1373–1379, October 1986.

[Turner 88] J.S. Turner.
Design of a Broadcast Packet Switching Network.
IEEE Transactions on Communications, vol. 36, no. 6, pp. 734–743, June 1988.

[Venkatesan 92] R. Venkatesan.
Balanced Gamma Network—a New Candidate for Broadband Packet Switch Architectures.
In *Proceedings of INFOCOM*, pp. 2482–2488, Florence, Italy, May 1992. IEEE.

[Widjaja and Leon-Garcia 92] I. Widjaja and A. Leon-Garcia.
The Helical Switch: a Multipath ATM Switch which Preserves Cell Sequence.
In *Proceedings of INFOCOM*, pp. 2489–2498, Florence, Italy, May 1992. IEEE.

[Yeh et al. 87] Y.S. Yeh, M.G. Hluchyj, and A.S. Acampora.
The Knockout Switch: a Simple, Modular Architecture for High-performance Packet Switching.
IEEE Journal of Selected Areas in Communications, vol. 5, no. 8, pp. 1274–1282, October 1987.

[Yemini and Florissi 93] Y. Yemini and D. Florissi.
Isochronets: an architecture for high-speed networks.
In *Proceedings of INFOCOM*, pp. 740-747, San Francisco, CA, USA, March 1993. IEEE.

[Yum and Leung 92] T.S. Yum and Y.W. Leung.
A TDM-based Multibus Packet Switch.

In *Proceedings of INFOCOM*, pp. 2509–2515, Florence, Italy, May 1992. IEEE.