On the Design and Implementation of the Isoswitch, a Novel Multi-protocol High-speed Switch[*]

Danilo Florissi
Departamento de Informática
Univ. Federal de Pernambuco
PO Box 7851, Recife, PE 50732-970
Brazil
dafl@di.ufpe.br

and

Yechiam Yemini
DCC Lab
Columbia University
450 Computer Science Bldg., New York, NY 10027 USA
yy@cs.columbia.edu

Resumo

Este artigo resume o projeto e implementação da versão eletrônica de um comutador, baseado na arquitetura para redes de alta velocidade Isochronets, capaz de suportar vários protocolos, de operar a gigabits por segundo e de se adaptar a várias configurações de número de portas e de velocidades de linhas de transmissão. Isochronets não precisam processar os cabeçalhos de mensagens que passam por seus comutadores e, conseqüentemente, podem se adaptar facilmente às velocidades das linhas, aceitam múltiplas estruturas de quadros e são adequadas para implementações totalmente óticas. A versão eletrônica do comutador tem custo baixo e utiliza componentes simples, amplamente disponíveis no mercado. A interface do comutador também é simples e provê novos serviços tais como Qualidade de Serviço (QoS) garantida e propagação de sinais de sincronização para os vários protocolos da rede, incluindo aplicações. O comutador permite negociação de QoS e, ao mesmo tempo, possibilita o compartilhamento de recursos de forma flexível. Através da utilização de tecnologias de implementação mais rápidas, a versão eletrônica pode potencialmente operar a centenas the gigabits por segundo, enquanto que uma versão toda ótica tem o potencial de operar a terabits por segundo.

Abstract

This paper overviews an electronic design and implementation of a scaleable gigabit per second multi-protocol switch based on the Isochronets high-speed network architecture. Isochronets do not require frame header processing in the switches and, consequently, they are scaleable with link speeds, support multiple frame structures, and are suitable for all-optical implementations. The electronic switch has low cost and uses simple off-the-shelf components. The switch interface is simple and provides novel services such as guaranteed Quality of Service (QoS) and propagation of synchronization signals to upper protocol layers, including applications. The switch allows the negotiation of QoS while promoting flexible resource sharing. Using faster implementation technology, the electronic design can reach scores of gigabits per second, while an all-optical design can potentially operate at terabits per second.

[*] This work was performed while the first author was pursuing his Ph.D. in Computer Science in the Distributed Computing and Communications (DCC) Lab at Columbia University.

1 Introduction

Judging by the pace at which current optical transmission link technology is evolving, future High-Speed Networks (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 Quality of Service (QoS) that integrated data, voice, and video applications need. Current switching techniques, such as Packet Switching (PS) and Circuit Switching (CS) [2, 11], may not appropriately fulfill these goals. For example, PS has limited support for guaranteed QoS, while CS enables limited bandwidth sharing. The goal of this section is to introduce the main challenges in designing and implementing a HSN switch.

- *Switching efficiency.* Current switching techniques rely on the fact that processing efficiency significantly prevails transmission efficiency. Unfortunately, the scenario is likely to change when HSN link speeds reach hundreds of gigabits or even terabits per second. For example, at 2.4 Gb/s, a switch has only 177 ns to process an ATM [3] cell (53 bytes). It is important to build switching technologies that can operate at these rates by minimizing frame header processing.
- *Tunable QoS guarantees.* New multimedia distributed applications require strict guarantees on the QoS delivered by the network in terms of maximum end-end delay, jitter, loss, etc. Switches must enable QoS tuning to satisfy an array of application demands while supporting effective resource use.
- *Implementation complexity and cost.* The provision of multimedia services to a large market will significantly increase the complexity and size of HSNs. It is thus reasonable to expect that the components (including switches) must be engineered to be simple.
- *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.
- *Compatibility with current network devices.* Current network devices will not cease to exist in future integrated HSNs. The switching technology must be compatible with such devices to assure wide acceptance.
- *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 rate families.
- *Scalability in the number of ports.* Scalability in the number of ports has always been an issue in switch design. Network systems grow continuously and switches must be scaled accordingly.
- *Minimal re-configuration overhead.* Switch re-confirmation must be performed without disturbing on-going transmissions.

This paper introduces the design and implementation of a novel low-cost multi-protocol high-speed scaleable switch based on the Isochronets [6, 14] switching architecture for HSNs that addresses the challenges above. Isochronets is designed to provide flexible control over QoS and multiplexing needs. Isochronets switch by Route Division Multiple Access (RDMA) — a technique that divides bandwidth over time among routes, as opposed to packets (as in

PS) or circuits (as in CS). In RDMA, a clock periodically distributes bandwidth or time bands among routing trees, enabling them. When a routing tree is enabled, frames move to the tree root. Switching is solely based on current tree configuration, being independent of frame header contents. RDMA can transfer any protocol frame without adaptation inside the network. In addition, RDMA is independent of any particular transmission technology and can scale with any transmission speed.

This work describes an electronic Isochronet Switch (Isoswitch) design and implementation using off-the-shelf components. The implementation consists of four input and four output channels operating at 1 Gb/s rates, despite the use of relatively low-speed Logic Cell Arrays (LCAs) [10]. These speeds can be increased considerably by using customized circuitry and by employing more parallel data lines inside the switch.

The Isoswitch presents novel features especially suitable for HSNs. 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 potentially contention resolution among contending links in the same tree. These function 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 the structure is only parsed by end nodes, being transparent during switching. This makes the Isoswitch easy to integrate with existing network components.
- The interconnection fabric between input and output ports is not controlled by any frame content, and thus any of the current modular interconnection technologies [12, 13] controlled by the slow periodic changes in tree configurations can be used to implement the fabric. Furthermore, Isoswitches can be interconnected creating hubs with potentially any number of ports.

Switch control functions can be accomplished in the Isoswitch through one simple mechanism: bandwidth allocation to routing trees. Because of this:

- QoS can be tuned to a number of application requirements using one simple mechanism. At one extreme, QoS demanding applications can allocate priorities for network resources to strictly guarantee QoS while, at the other extreme, non-QoS demanding applications can favor resource multiplexing.
- 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 includes 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.

Due to space constraints, this document will not overview the all-optical Isoswitch design that can be found in [6].

This paper is organized as follows. Section 2 overviews Isochronets and RDMA. Section 3 is a detailed description of the electronic Isoswitch design. Section 4 describes the implementation of an interface between a workstation and the Isoswitch. Section 5 overviews the efficiency, complexity, and scalability of the Isoswitch. Section 6 evaluates the compatibility of the Isoswitch with current network devices. Section 7 describes the services provided by the Isoswitch. Finally, Section 8 concludes.

2 Isochronets Background

2.1 Architecture and Principles of Operations

Isochronets divide network bandwidth among routing trees, that is, network switches are configured periodically to build different routing trees. Each configured tree lasts a time band (*green band*) after which the next time band and corresponding tree replaces it. A sequence of bands building trees that cover each destination once and only once form a *cycle*, which is repeated periodically.

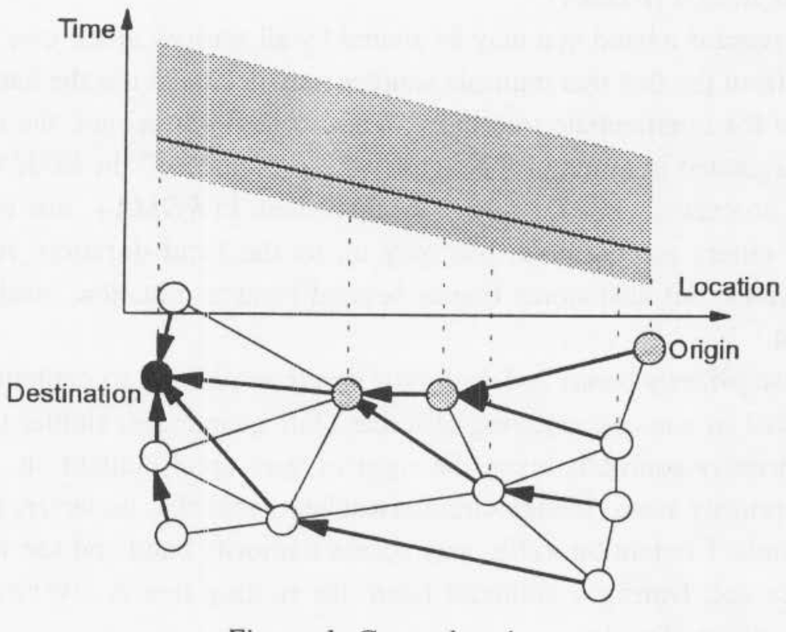


Figure 1. Green-band.

Isochronets seek to provide flexible control of contention to accomplish desired QoS by routing network traffic along the routing trees leading to respective destination nodes. Bandwidth is time-divided among, and synchronized along routing trees by way of the time band mechanism. Bands are usually of different sizes and may be adapted according to the traffic load in each tree. Figure 1 depicts a network topology with the routing tree (marked with directed thick links) leading to the dark node. The graph on top plots traffic motion from source to destination through the gray nodes. The Location-axis shows the location of a given frame at the time marked in the Time-axis. During the green-band (shaded area in the graph), a frame transmitted by a source will propagate down the routing tree to the destination root (a typical frame motion is depicted using a line within the shaded area). If no other traffic contends for the tree, the frame will move uninterrupted, as depicted by the straight line.

The green band is maintained by 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 configuration of switch resources 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.

Bands need not occupy the same width throughout the network. Indeed, one can view a green band as a resource that is distributed by a node to its up-stream sons (as long as the bands allocated to sons are scheduled within the band of the parent). In particular, if the bands allocated to two sons 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.

One may view these mechanisms to schedule traffic motions by way of band allocations as a media-access 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) [2, 11]. This technique, accordingly, is called *Route Division Multiple Access (RDMA).*

A *contention band* is a band that may be shared by all sources in the tree simultaneously. Its name is derived from the fact that multiple sources may decide to use the band at the same time and thus *contend* for intermediate tree links. When contention occurs, the collision resolution mode used is designated in terms of the signs "–", "+", and "++". In *RDMA–*, only one of the colliding frames proceeds, while the others are discarded. In *RDMA+*, one colliding frame proceeds while the others are buffered, but only up to the band duration. *RDMA++* operates similarly to RDMA+, but also stores frames beyond band termination, rescheduling them during the next band.

Isochronets use *priority bands* and *multicast bands* 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 upon contention, by switches on its path, during its priority band. Unlike circuit-switched 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 multicast to any subset of nodes.

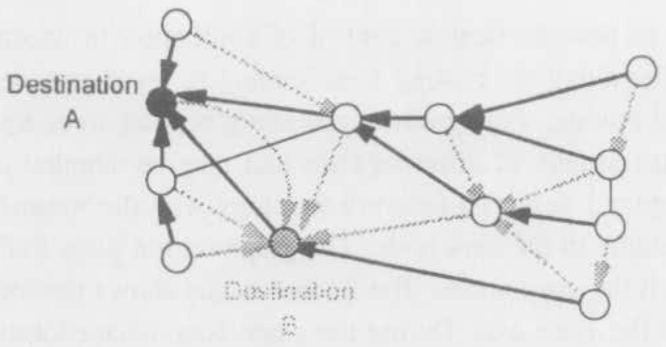


Figure 2. Multiple non-interfering trees.

Bands are thus shared resources that may be passed from intermediate nodes to subtrees. Nodes may decide to pass only portions of their bands to their sons. Also, the portions may be of different sizes. Thus, the final band allocation scheme may be designed taking advantage of the rich structure enabled by the band allocation possibilities.

A few observations regarding Isochronets are in order. Multiple simultaneous routing trees can schedule transmissions in parallel (have simultaneous green bands), depending on the network topology. Figure 2 shows two non-interfering routing trees.

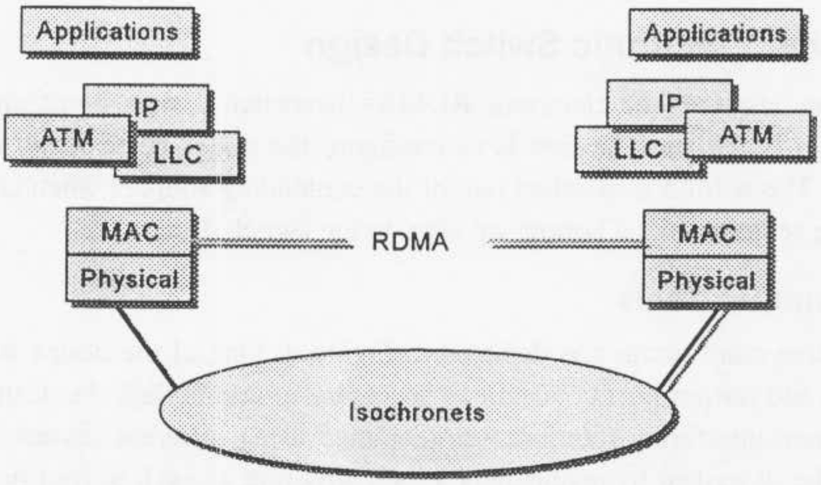


Figure 3. Multiple protocol stacks in Isochronets.

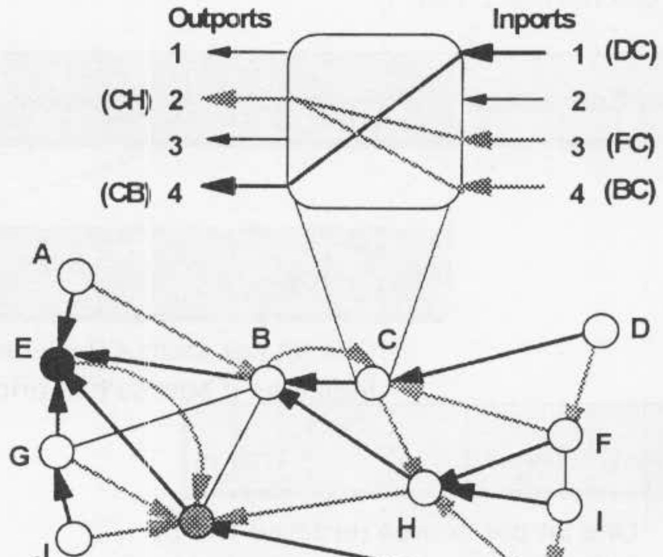


Figure 4: Tree allocation and node configuration.

No header processing is necessary in Isochronet nodes. Frames on incoming links can be mapped into the corresponding outgoing link based solely on the current routing tree structure which, in turn, may be derived from the current time. Thus switching can be accomplished without any processing that is dependent on frame contents. This means that Isochronets may operate at any link speed.

Since no frame processing is performed at intermediate nodes, all stack layers above the media-access layer are delegated to interfaces at the network periphery. That is, Isochronets may transport any frame structure without adaptation because frames do not need to be parsed to derive routing information. A typical stack organization for Isochronets is depicted in Figure 3. Interconnection of Isochronets can be accomplished by way of media-layer bridges using extensions of current well-understood technologies.

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.

3.1 Configuration Table

A typical tree configuration is depicted in Figure 4. 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 5.

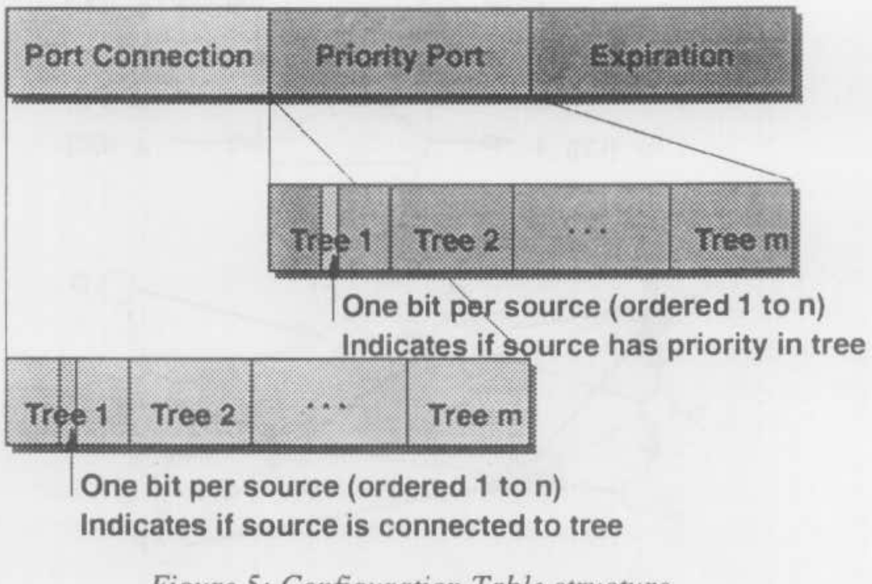
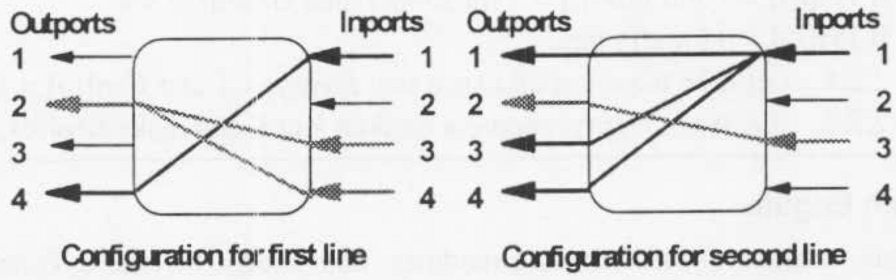

Port Connection
Priority Port
Expiration
Tree 1
Tree 2
...
Tree m
One bit per source (ordered 1 to n)
Indicates if source has priority in tree
Tree 1
Tree 2
...
Tree m
One bit per source (ordered 1 to n)
Indicates if source is connected to tree

Figure 5: Configuration Table structure.

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 $[1,m]$.

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 bits (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.



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: Configuration Table in a particular network node.

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

Figure 6 illustrates the CT at a particular network node. The CT is implemented as a RAM memory.

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 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 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 $Grant[k,j] = 1$.
```

3.3 Switch Engine

The switch consists of the following modules: Line Cards, Switching Fabric, and Control Unit. The Control Unit receives configuration data from a Host machine directly connected to it. These components are depicted in Figure 7.

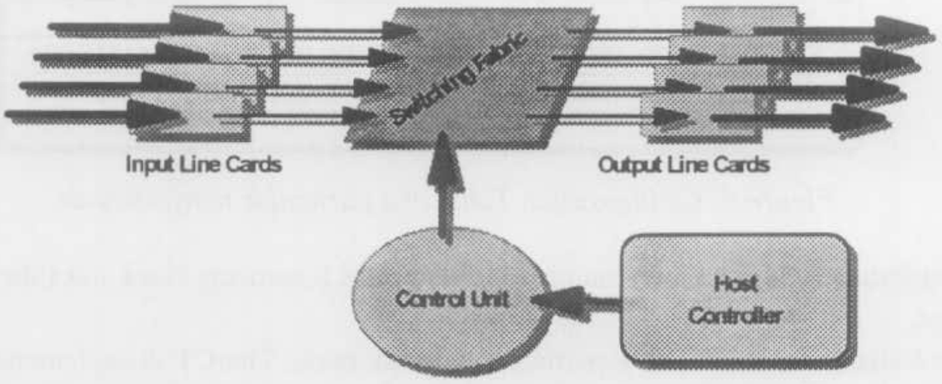


Figure 7: Switch organization.

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 8 depicts the Input Line Cards. The

Isoswitch uses 40-bit words internally.

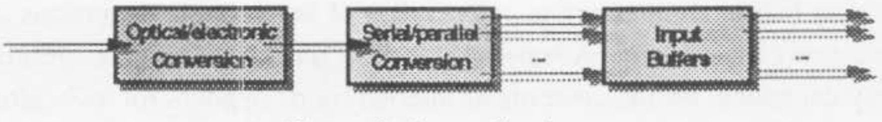


Figure 8: Input Card.

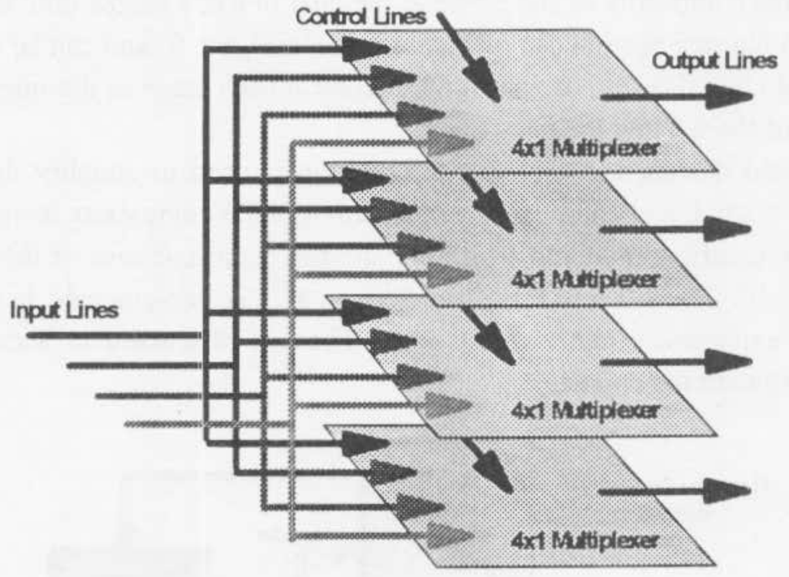


Figure 9: Switching Fabric.

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 [2]. 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 9. 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.

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 [12, 13] 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 9, 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 10. The CT (discussed in Section 3.1) is implemented in the Configuration Memory (CM), explained in more details later. The AL (discussed in Section 3.2) is implemented as a combinational circuit.

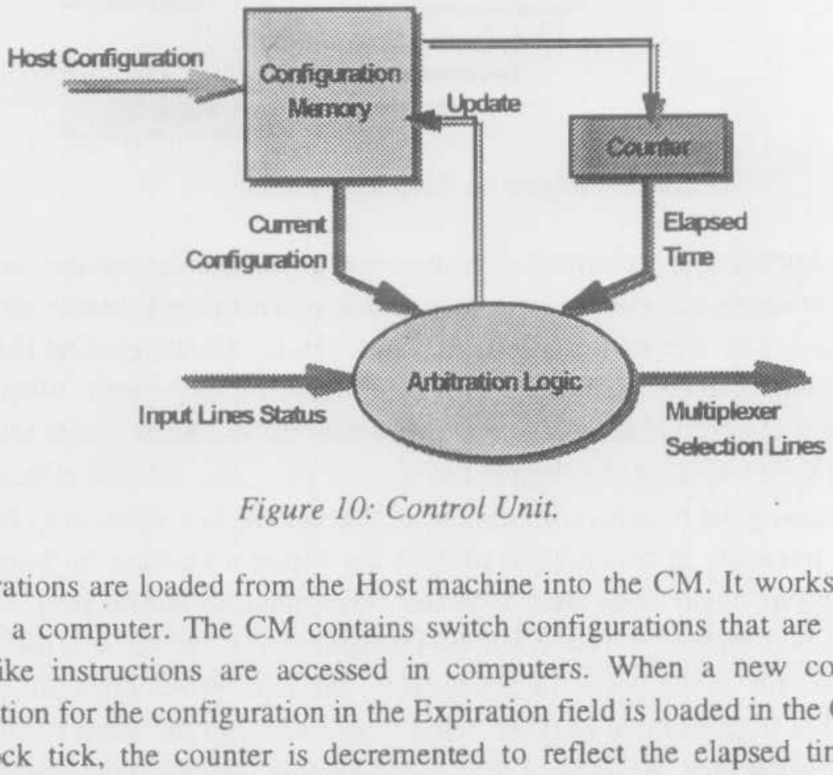


Figure 10: Control Unit.

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 11 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.

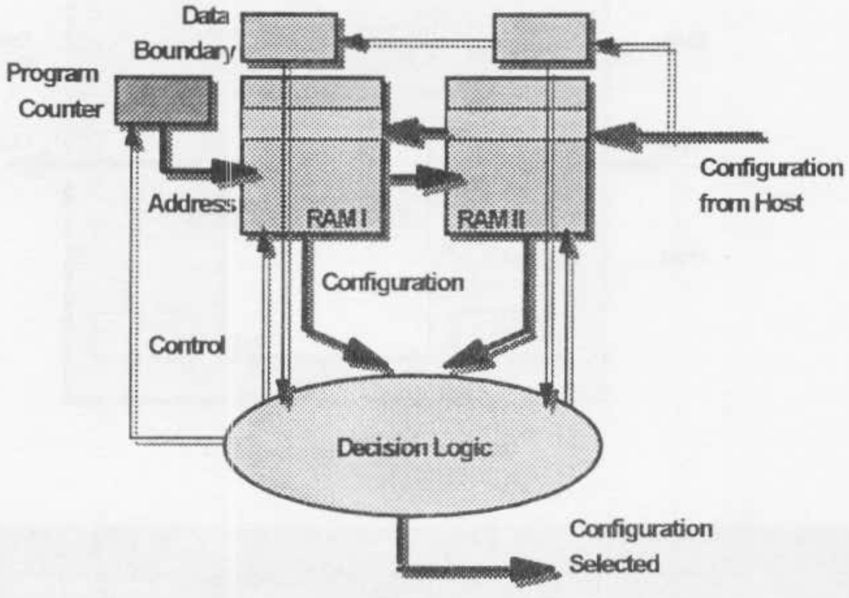


Figure 11: Configuration Memory.

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 12 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 [6].

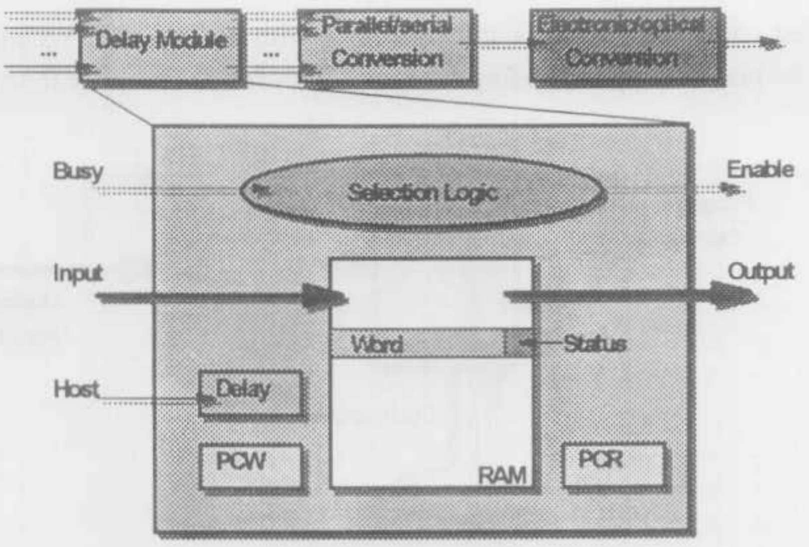


Figure 12: Output Card.

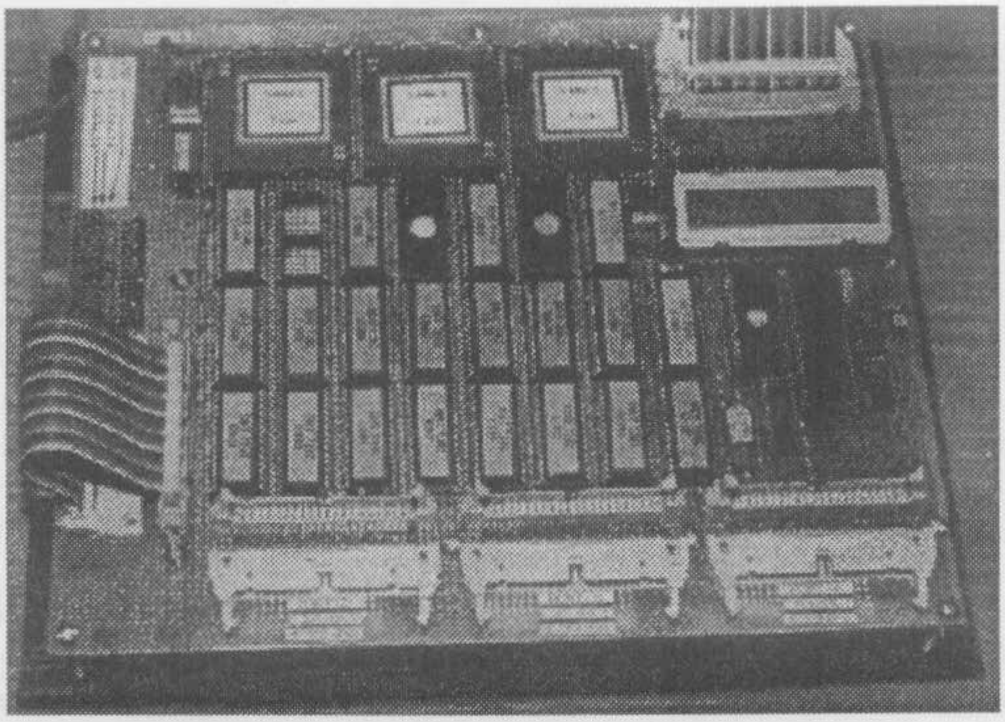


Figure 13: Isoswitch implementation.

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.

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 13. 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 [10] 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.

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 [6, 7].

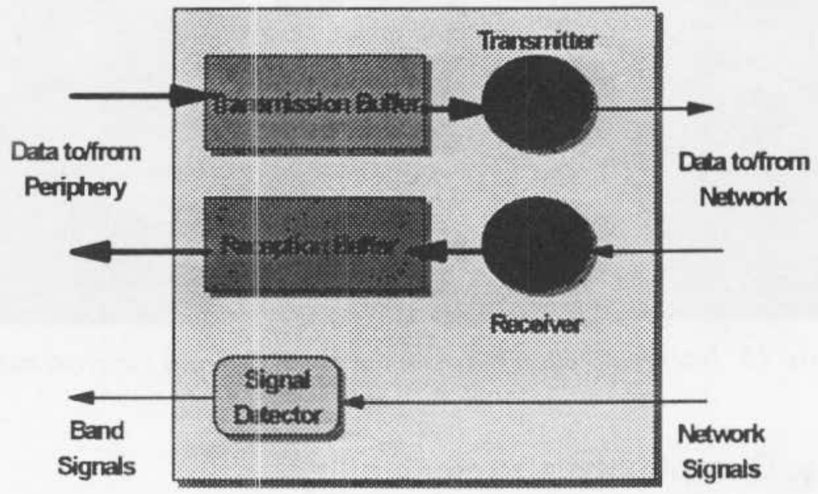


Figure 14: Isochronet interface to an end machine.

Figure 14 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.

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 machines. A picture showing the connection of the switch to the SPARC containing the card is shown in Figure 15. 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.

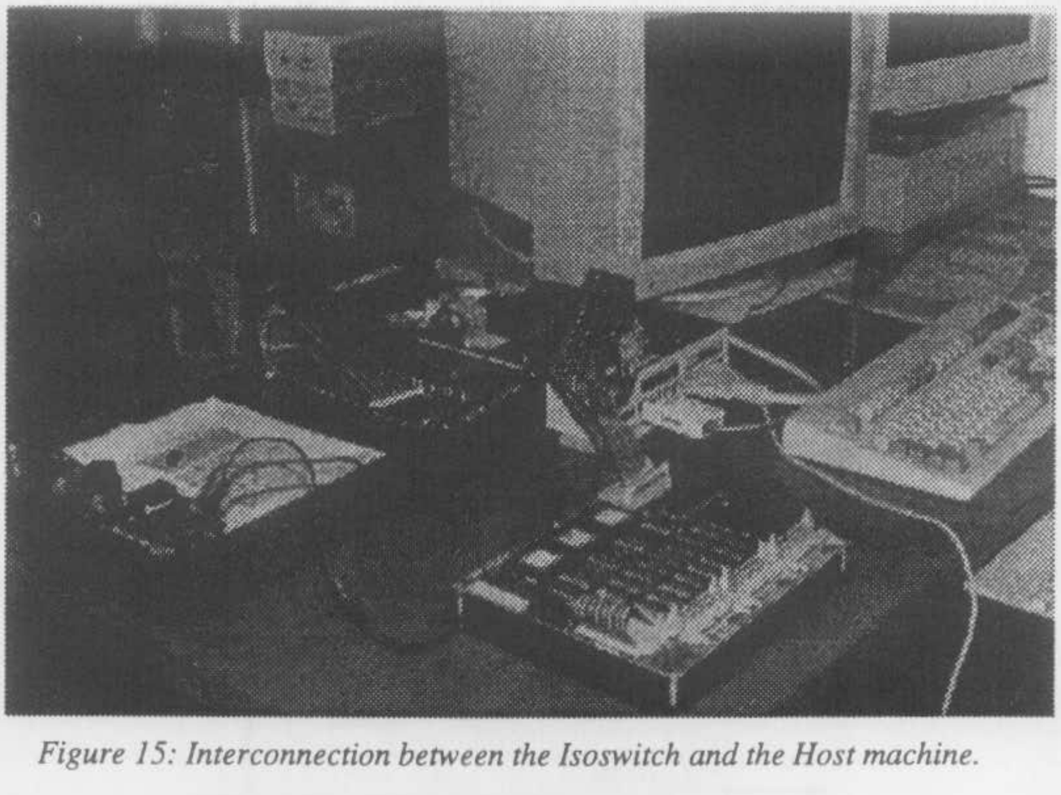


Figure 15: Interconnection between the Isoswitch and the Host machine.

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 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 3.3.

The Switching Fabric implemented contains m tandem multiplexers with n inputs each (see Section 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 3.3, the re-configuration overhead is absent in the Isoswitch due to the use of dual memory modules.

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.

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 [6, 7], 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 4. The main functionality is frame forwarding, reception, and signal forwarding. All these functions can be accomplished with simple circuitry.

8 Conclusions

Isochronets are a new switching architecture that reduces all network-layer functionality to the media access layer. As a result, (1) no frame 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 Isochronet switch (Isoswitch). Because switching is independent of frame contents, both an electronic and an all-optical implementation of the Isoswitch are possible.

This work developed an electronic design and implementation of the Isoswitch. The implemented Isoswitch has four input and output channels operating at 1 Gb/s. The design is modular and scaleable with respect to an increase in the number of channels for both logic complexity and internal switch latency. Inter-operability with other switching techniques is simplified. Finally, it offers novel services: (1) synchronous signaling, (2) no necessity for adaptation, and (3) guaranteed Quality of Service (QoS) provision.

References

[1] Acampora, A.S. and Karol, M.J., "An overview of light-wave packet networks," IEEE Network Magazine, vol. 3, no. 1, pp. 29–41, January 1989.

[2] Bertsekas, D. and Gallager, R., *Data networks*, Second Edition. Prentice Hall, 1992.

[3] Boudec, J.Y.L., "Asynchronous Transfer Mode: a tutorial," *Computer Networks and ISDN Systems*, vol. 24, no. 4, pp. 279–309, May 1992.

[4] Brackett, C.A., "Dense wavelength division multiplexing networks: principles and applications," *IEEE Journal of Selected Areas in Communications*, vol. 8, no. 6, pp. 948-964, August 1990.

[5] Dono, N.R., Green, P.E., Liu, K., Ramaswami, R., and Tong, F., "A wavelength division multi-access network for computer communications," *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 6, pp. 983–994August 1990.

[6] Florissi, D. "Isochronets: a high-speed network switching architecture (Ph.D. Thesis)," Tech. Rep. CUCS-021-95, Computer Science Department, Columbia University, 1995.

[7] Florissi, D. and Yemini, Y., "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.

[8] Mills, D.L., "Internet time synchronization: the Network Time Protocol," *IEEE Transactions on Communications*, vol. 39, no. 10, 1482–1493, October 1991.

[9] Ramaswami, R., "Multiwavelength lightwave networks for computer communication," *IEEE Communications Magazine*, vol. 31, no. 2, pp. 78–88, February 1993.

[10] Roth, C., *Fundamentals of logic design*, Fourth Edition, West Publishing Co., 1992.

[11] Tanenbaum, A.S., *Computer networks*, Second Edition. Prentice Hall, 1988.

[12] Tobagi, F.A., "Fast packet switching architectures for broadband integrated services digital networks," in *Proceedings of the IEEE*, vol. 78, no. 1, 133–167, January 1990.

[13] Turner, J.S., "Design of a broadcast packet switching network," in IEEE Transactions on Communications, vol. 36, no. 6, 734–743, June 1988.

[14] Yemini, Y. and Florissi, D., "Isochronets: a high-speed network switching architecture," in *Proceedings of INFOCOM*, pp. 740–747, San Francisco, California, USA, March 1993.