The Gigabit per Second Isochronet Switch

Danilo Florissi and ***Yechiam Yemini***

Distributed Computing and Communications (DCC) Lab
450 Computer Science Bldg., Columbia University, NYC, NY 10027

Technical Report CUCS-006-95

{Florissi,Yemini}@cs.columbia.edu

Abstract

This paper overviews the electronic and all-optical design and the gigabit per second electronic implementation of a switch based on the Isochronets high-speed network architecture. The absence of frame processing in the Isochronets architecture is core to the switch efficiency and multi-protocol support. The electronic design is low cost and uses simple off-the-shelf components, while the optical design can be realized with current optical devices. The switch interface is simple and provides novel services such as propagation of synchronization signals to upper protocol layers. The switch makes it possible the negotiation of Quality of Service (QoS) while promoting flexible resource sharing. The modular switch designs are scaleable in number of nodes and link speed. Using faster implementation technology, the electronic design can reach scores of gigabits per second, while the all-optical design can potentially operate at terabits per second.

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, Packet Switching (PS) and Circuit Switching (CS) [2, 11], may not appropriately satisfy 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 imperative to build switching technologies that scale efficiently with link speeds.

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, or else the technology may be doomed.

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 are continuously in the grow and switches must be scaled accordingly.

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

The switch described in this work is based on the Isochronets [6, 14] switching architecture for HSNs, designed to flexibility tune multiplexing and QoS 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 distributed bandwidth or time bands among interfering routing trees. When a routing tree is enabled, frames move to the tree root. Switching is solely based on current tree configuration, being independent of traversing frame contents. RDMA can transfer any protocol frame without adaptation in the network. In addition, RDMA is independent of any particular transmission technology and can scale with any transmission speed.

This paper describes an electronic and an optical Isochronet Switch (Isoswitch) design and the electronic 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). These speeds can be increased considerably by using customized circuitry and by employing more parallel data lines inside the switch.

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 on internal delay, scaleable with respect to link speeds, and simple to implement.
- Multiple frame structures can be supported without adaptation because frame structure is only used 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] can be used to implement the fabric, controlled by the slow periodic changes in tree configurations. Furthermore, Isoswitches can be interconnected creating hubs with potentially any number of ports.

Switch control functions can be accomplished through one simple mechanism: bandwidth allocation to routing trees. 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.

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 main advantages of adopting the Isoswitch in HSNs. Section 6 gives a broad all-optical Isoswitch implementation. Finally, Section 7 concludes this paper.

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.

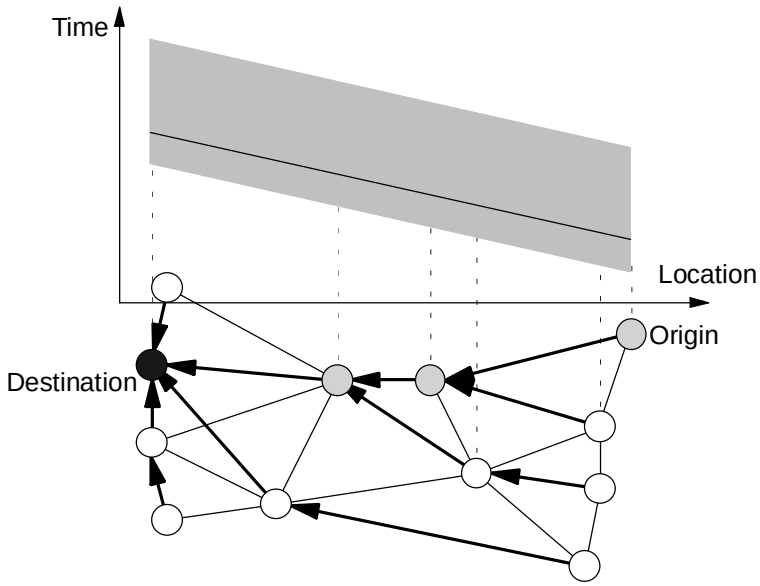


Figure 1. Green-band.

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.

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.

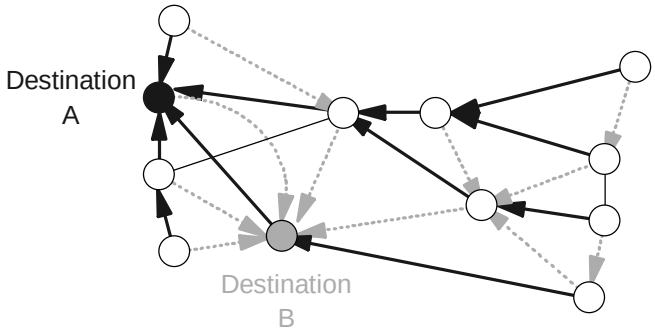


Figure 2. Multiple non-interfering trees.

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.

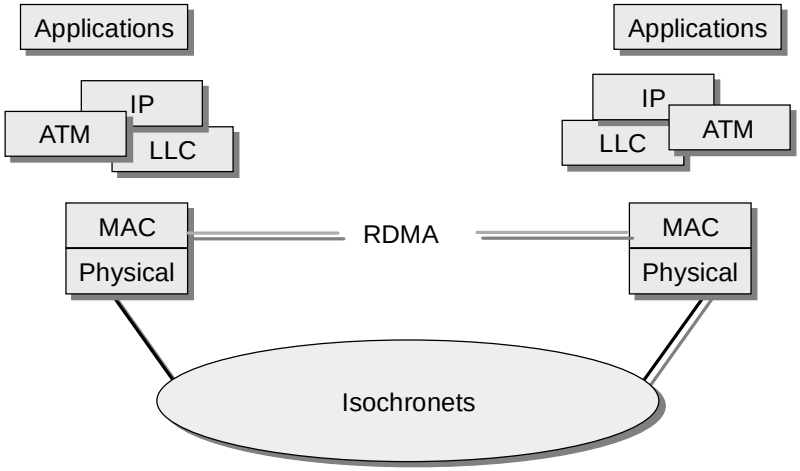


Figure 3. Multiple protocol stacks in Isochronets.

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, such 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.

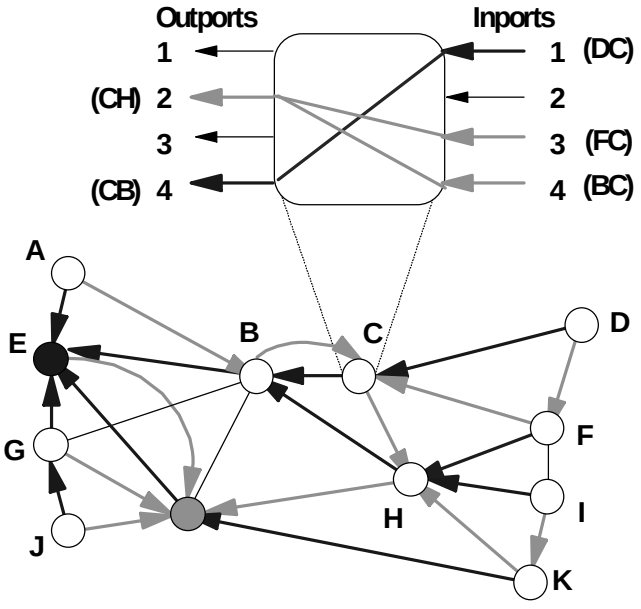


Figure 4: Tree allocation and node configuration.

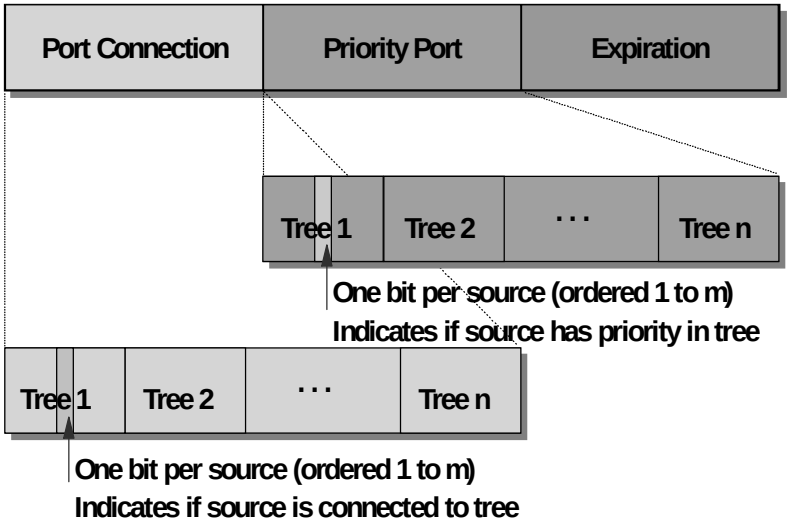


Figure 5: Configuration Table structure.

For the reminder of this section, it is assumed that each node has n input ports (inports) and m

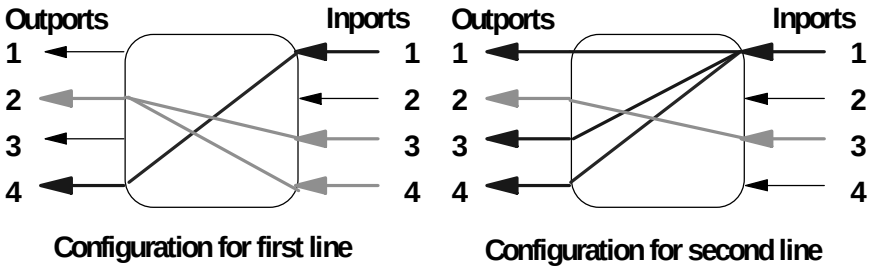
output ports (outports). Each inport is identified by a number in the range [1,n]. Similarly, outport identifiers are in the range [1,m].

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 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 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: Configuration Table in a particular network node.

The CT is implemented as a memory module.

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. The following refers to the definitions in the algorithm. 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 an input that has priority to the output. If so, that input is granted access to the output. In Step 2.2, if not 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/output connectivity.

Definitions. *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. The following 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.*

Algorithm 1: Arbitration Logic in Isoswitch.

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 7.

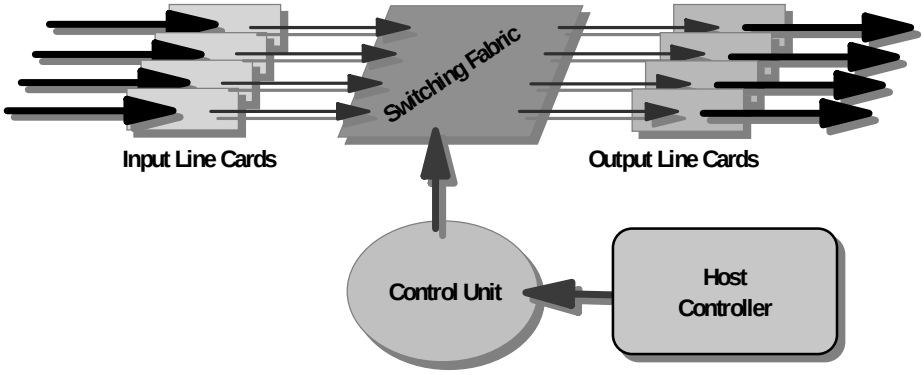


Figure 7: Switch organization.

The host machine computes band allocation information off-line with respect to data transmission and reception. 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 host also synchronize its clock with other nodes using the Network Time Protocol (NTP) [8] and provide synchronization to the control unit.

The input line card receives bit-serial data from the network trunks and convert 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 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.

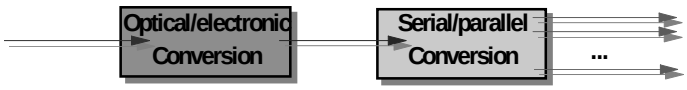


Figure 8: Input Card.

Input queue buffering is performed after the serial to parallel conversion, simplifying queue design. In Isochronets, input or output queueing delivers the same performance. This is not usually the case in normal switching technologies due to the "Head of the Line" (HoL) blocking effect [3]. HoL blocking happens, for example, in packet switching 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.

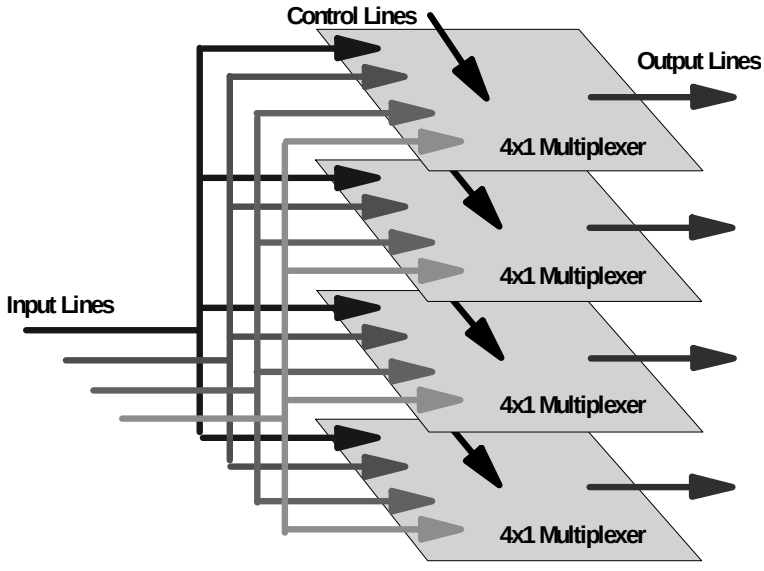


Figure 9: Switching Fabric.

The switching fabric architecture is independent of Isochronets operations, and can be implemented using any interconnection architecture currently being proposed. In particular, architectures that may not switch dependent of the contents of internal frame headers are best suitable for

RDMA. Multi-stage interconnection networks [12, 13] are one such classes of architectures that reduce the internal complexity of the fabric at the cost of extra stages (and thus added end-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. If the internal end-end delay is increased significantly, the batch size can be increased accordingly to amortize the cost. Nevertheless, a complete configuration is used in the prototype 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 below. The AL (discussed in Section 3.2) is implemented as a combinational circuit.

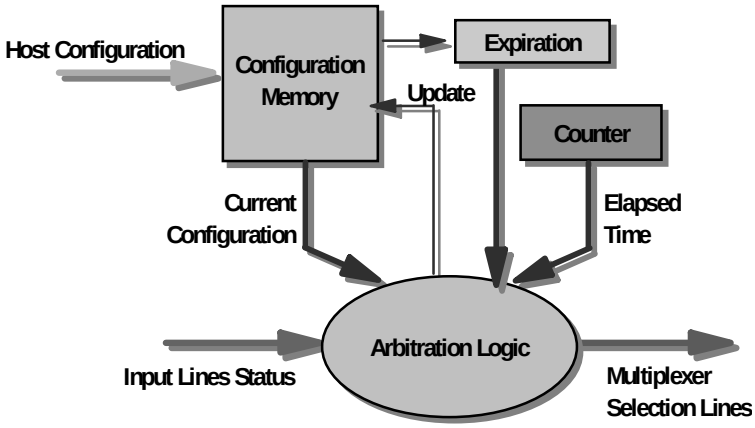


Figure 10: Control Unit.

New configurations are loaded from the host machine into the CM, structured as a CT. The CM works similarly to a program memory, where the instructions are configurations fetched in sequence. These configurations form the Isochronet cycle and their structure was explained in Section 2. 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.

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 which decides the configuration of the multiplexers in the switching fabric. By selecting the proper multiplexer control, the correct input and output connectivity is achieved.

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 loaded operation 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 which 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.

Meanwhile, the host can directly address the other RAM to load 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 it will begin using the new CT once the current Isochronet cycle is finished.

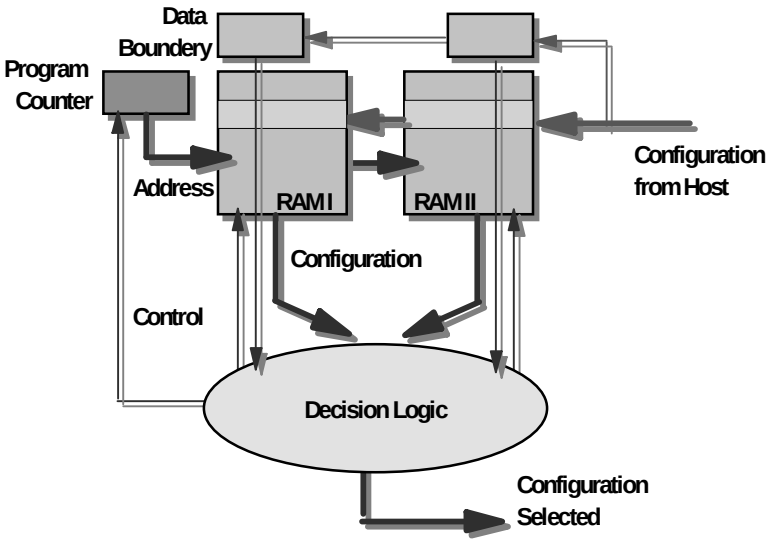


Figure 11: Configuration Memory.

Output Line Cards. Figure 12 depicts the output line card. Besides performing the inverse function of the input line card, it includes a delay element before the parallel to serial bit conversion module. The main function of the Delay Module is to implement the delay that is necessary to make the whole link propagation delay a multiple of the cycle time. 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 are explained in [7].

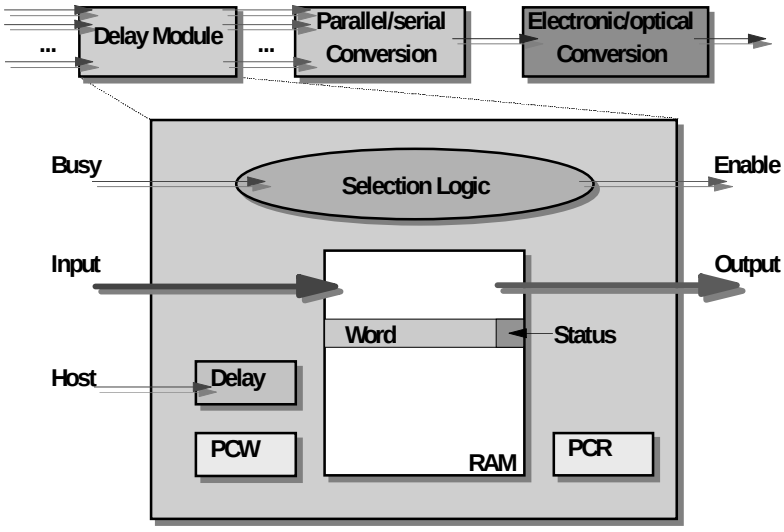


Figure 12: Output Card.

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-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 RAM is a dual port memory module, that is, it 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 equivalent to delay elapse before the first transmission occurs, thus achieving the required delay.

3.4 Operational Characteristics

The Isoswitch was implemented using the 4005H family Xilinx Logic Cell Arrays (LCAs), as shown in Figure 13. This technology is relatively slow, but was chose due to the flexibility provided in implementing prototype systems. To get a feel of how slow the 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 gate is 5 ns. Additionally, the high-level specification is compiled into LCA gates that may be positioned far apart, diminishing even more the overall speed. Even at this slow speed, it was possible to accomplish the goal of implementing a 1 Gb/s per port Isoswitch due to the simplicity in Isochronets operations.

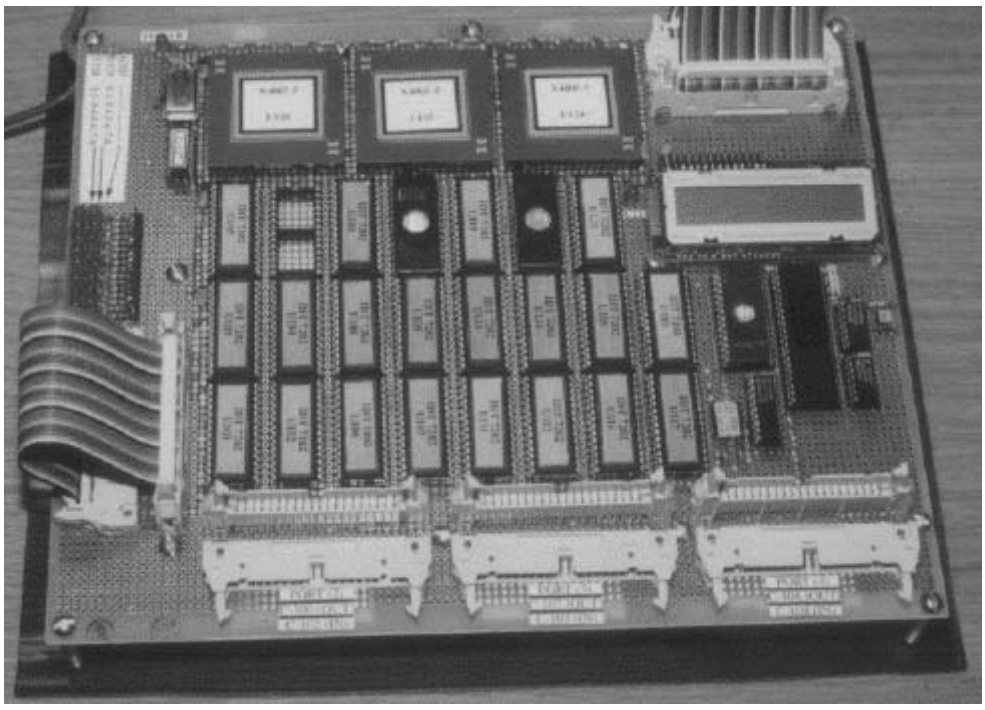


Figure 13: Isoswitch implementation.

The Isoswitch control unit operates at 3.125 MHz, thus reaching 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. Each data word takes 40 ns to cross the switching fabric to the output port. The nominal rate in each input port is thus 3.125×8×40 Mb/s, or 1 Gb/s.

The switch implemented in this phase is to be a building block for an Isochronet hub (Isohub) interconnecting 4 such switches. Since the distances in the hub are small, the output card in Figure 12 was simplified and no delay module is implemented in the Isoswitch.

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 [7] interface.

Figure 14 depicts the interface card to the Isoswitch. Buffers are placed in the interface for data transmission and reception. The objective of the data transmission buffer is to gather data from the slower host machine so that it can be sent at full speed through the switch. Data reception buffer is used to store data received from the high-speed switch until the host machine can access it.

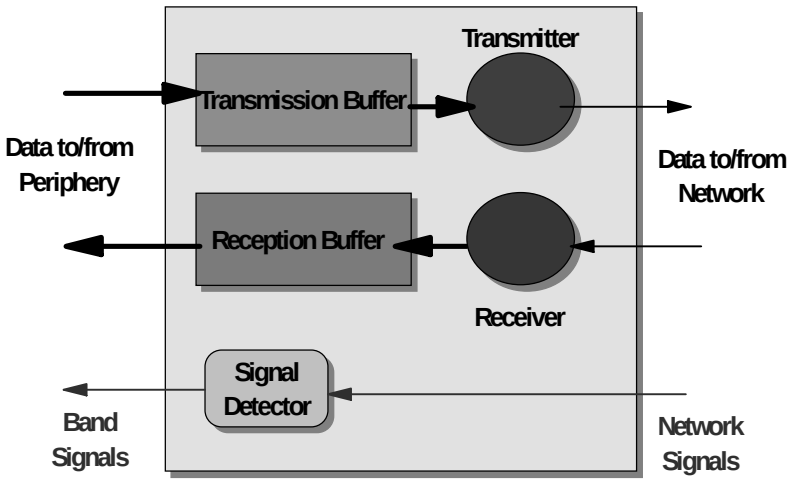


Figure 14: Isochronet interface to an end machine.

The card receives signaling information from the switch when the following events occur: cycle begin, band begin, and data reception. These signals are forwarded to the host machine containing the interface card in the form of interrupts. A status register can be read from 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 spool the status registers to see when events occur. Finally, a control register 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 SPARC machines. The connection of the switch to the SPARC containing the card is shown in Figure 15. Once the transmission buffers are full, the card sends data to the switch at the peak 1 Gb/s rate. 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 implement SBus transfer showed a maximum throughput of 22 Mb/s for plain transfers.

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

5 Isoswitch and HSNs

This section evaluates the suitability of the Isoswitch design for HSNs, according to the goals set forth in Section 1.

5.1 Switch Efficiency, Re-configuration Overhead, Implementation Complexity, and Scalability in Speed and in Number of Ports

This section evaluates the switch efficiency and complexity and how these values scale with the number of ports and the link speeds. The re-configuration overhead is also evaluated. This section evaluates these measures in terms of 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 combinational logic circuits have as input a portion of *Pri* on size n plus *Busy* which is of size n as well (see steps 2.1 and 2.2). Thus, each of the m parallel combinational circuits have complexity $O(n)$. The overall AL complexity is thus $O(nm)$. As for latency, since m parallel combinational circuits are processing n entries, the 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) occupied 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*. It does not depend on other factors normally found in other switching architectures for HSNs such as processing per frame. Processing and link transmission per frame affect 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 be 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. 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, but 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 arbitration rate must be *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 bits frames, the latency must be 424 ns. Notice that such constraint is not a problem, since only combinational arbitration is to be performed (no processing dependent on frame contents). For example, the arbitration latency for the implemented Isoswitch is 320 ns.

The linear dependency of the switch latency on *n*, nevertheless, seem to place a serious constraint on the overall Isoswitch performance when the number of input ports is increased. In reality both, the constraint on link speed and on number of ports, can be overcome by the following techniques applicable due to the RDMA technique.

In the first technique, each periphery source must send always *n* consecutive frames. Since arbitration is now performed for the whole block of *n* frames, its latency may be *nf/b* (that is, n times bigger than originally). Notice that such technique cannot be used in any switching technique that needs to process frames because the switching decision for each of the *n* individual frames may be different. Furthermore, this technique does not affect the frame size (that is, it is different than 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.

In the second technique for link speed increase, The internal data bus in the switch is increased. By increasing the bus size *n* times, the control unit can operate *n* times slower. Notice again that such technique relies heavily on the fact that no frame-dependent processing occurs in the Isoswitch.

Switching Fabric Logic. The interconnection of input and output ports must be preceded by processing in traditional switch design. If a fully connected topology would be chosen for the fabric, the processing capability at each outport would need to be *np*, where *n* is the number of input ports and *p* is the speed at which frames from a port need to be processed. Such processing necessity would make the switch design very complex. To overcome this problem, some HSN switches use blocking fabrics, that is, fabrics that allow only a subset of all possible input to output connections in parallel. Processing is then distributed within the fabric and reduced based on the blocking factor within the fabric. Since no processing occurs in the Isoswitch, the fabric can be implemented as a fully connected network with low combinational logic complexity.

The implementation complexity is reduced in Isochronets since no processing occurs in the fabric, but the complexity in terms of the number of input to output lines can be considerable. To

reduce this complexity, many solutions can be adopted to reduce the number of lines, such as for example multi-stage interconnection networks, as discussed in Section 3.3. It is important to emphasize that Isochronets technology are independent of the interconnection fabric adopted, and multiple solutions with diverse cost-performance ratios can be chosen.

The switching fabric implemented contains *m* tandem multiplexers with *n* inputs each (see Figure 9). Thus, its complexity is *O(nm)* and its latency is *O(n)*.

The switching fabric implementation occupies only 43% of the LCA chip logic, whereas 92% are used.

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. One simple example is to connect two 4×4 Isoswitches by interconnecting one of the output ports of the first to one of the input ports of the second. Such connection would generate a 7×7 Isohub.

Finally, as mentioned in Section 3.3, the re-configuration overhead is absent due to the use of dual memory modules.

5.2 Compatibility with Current Network Devices

The end-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 very powerful 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. Thus, other protocol operations can be supported through the Isoswitch directly, with no changes.

Resource allocation in the Isoswitch is also simplified because allocation occurs off-line with transmissions. That is, the resource to be allocated is a band, which is occurs at a frequency much lower than circuits. As a result, resource management at periphery nodes is simplified as well.

5.3 Switching Services and Interface Complexity

This section analysis the Isoswitch as a black box and identifies the fundamental services that it offers. The emphasis in on services that can be provided in addition to traditional switching services. Additionally, the complexity of the interface is evaluated.

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. In this manner, a 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 in the end nodes to provide synchronous services. The idea is to relay such signals upwards through the protocol stack so that the periphery protocol stack can use to implement synchronous services with strict QoS guarantees. A whole new *synchronous protocol stack* (see reference [7]) can be implemented

using this feature.

These signals are novel when compared to traditional STM synchronization in many respects. Firstly, 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 microseconds. Secondly, no global network-wide synchronization is necessary. Synchronization is achieved through local exchange between a periphery node and directly attached Isoswitch. Thirdly, the signals embody routing and QoS information that may be used by periphery nodes to schedule its activities. For example, these signals can be forwarded to applications that can schedule their activities to the signals. When sending video frames, for instance, applications may generate a frame and sleep waiting for the signal. When the signal arrives, it forwards the frame, generates the new one, and sleeps again on the same event. In this way, frame buffer and processing resources are used only when necessary, thus improving overall resource use.

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, sources can transmit to the respective destinations and be assured that no loss or delay will occur in the switch due to contention.

The interface complexity is minimal, as explained in Section 4. The main functionality is frame forwarding, reception, and signal forwarding. All these functions can be accomplished with simple circuitry and their complexity is independent of the number of nodes, depending only linearly on the link speed. Such dependency can be amortized with simple techniques such as pipeline and dual memory modules in the implementation of the interface card.

6 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) [1, 4, 5] to allocate one wavelength for each band. The architecture for a single tree per band is depicted in Figure 16. Each wavelength is depicted using a different gray scale. Incoming wavelengths are first fed into a selection box (explained below) 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 one of the frames. Contention occurs when two or more of the input links is pouring 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 17. Its function is to detect incoming signals from the links and immediately grant access to one of them, shutting the others.

Each input line is extended inside the selection box. At each entrance, an optical sensor detects incoming light. The idea is to only allow one input per wavelength into the broadcast link. The

selection logic achieves the decision as to which of the inputs should be allowed to proceed and which should be shut. The decision is reached using AL with inputs from the CT and input 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 is big enough to allow for the decision before the optical data reaches the end of the extension.

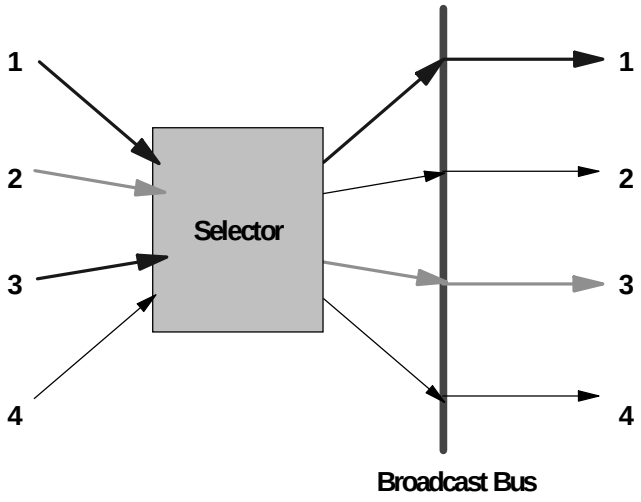


Figure 16: All-optical switch implementation: one tree per band.

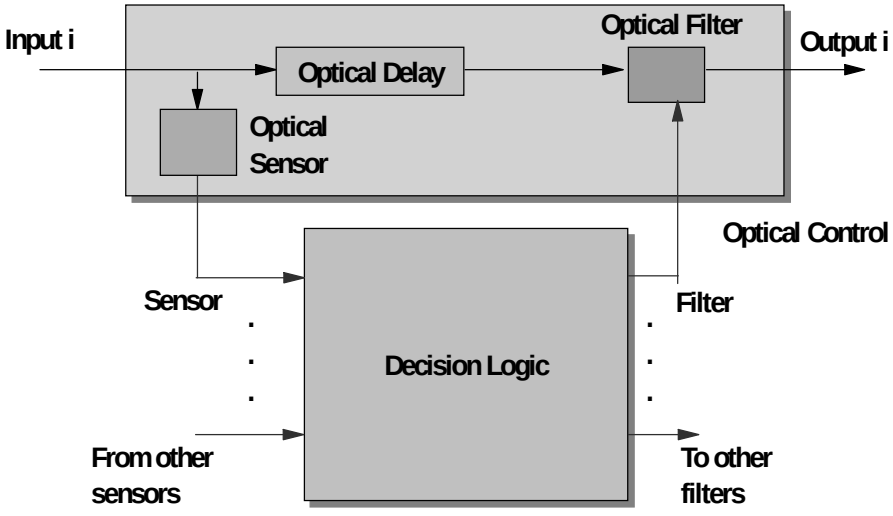


Figure 17: Selection box.

Priority bands are implemented similarly 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. Notice that priority bands are implemented in similarity with the electronic Isochronet implementation, using time division. Priority bands exist over periods of time in a cycle, whereas contention bands are always “opened”.

Multiple trees per band are implemented by extending the architecture in Figure 16 into multiple broadcast links. The idea is to propagate different routing trees in the same broadcast link us-

ing separate wavelengths, but still be able to reuse wavelengths in separate broadcast links. In this manner, the total number of wavelengths necessary is reduced.

The implementation of the multiple broadcast link scheme is the following. Each input link is connected to all the broadcast links, but optical filters are placed at 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 input links broadcast the same wavelength through the same broadcast link. Receivers are placed at each output link. Since only one tree for each wavelength is mapped in each broadcast link, receivers listen to the wavelength of the tree they represent. All the detected wavelengths are multiplexed into the output link and regenerated by the transmitter.

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 [9] for a survey of such schemes) need to provide a special control channel for the reservation of wavelength prior to communication. These schemes suffer the drawbacks of reservation schemes, such as round-trip allocation delay, necessity for rapidly-tunable receivers/transmitters, and dedicated bandwidth. Other schemes (also in [9]) switch frames, with the added delay for media conversion and frame processing. Third, the implementation described is cheaper since it only needs to tune when adjusting the band sizes, which occurs at much slower rates than the speed of incoming frames.

It is important to notice that even though the all-optical implementation uses RDMA-, all bands are opened all the time, avoiding synchronization of bands.

Nevertheless, frame loss may occur in this scheme during contention bands. It is best that an all-optical implementation uses 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 $\frac{\lambda}{n}$ be the input rate (as a percentage of the peak rate 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 $1 - \left(1 - \frac{\lambda}{n}\right)^n$ (that is, if at least one source transmits). As $n \to \infty$, the rate becomes $1 - e^{-\lambda}$. The expected success probability is $\frac{(1 - e^{-\lambda})}{\lambda}$. Finally, the expected loss probability is $1 - \frac{(1 - e^{-\lambda})}{\lambda}$. When $\lambda \to 1$ (loaded system), the expected loss is e^{-1} (less than 37%).

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. If each frame is repeated m times, the loss probability becomes e^{-m}. Thus, m may be computed from the maximum loss rate r that can be tolerated in the system: $r \le e^{-m}$ so that $m \ge -\ln r$. For example, $m = 4$ insures less than 2% loss rate when the system is heavily loaded and the number of input sources is big.

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 frames 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 tol-

erated loss rate.

7 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 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 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.

8 References

[1] Acampora, A.S. and Karol, M.J., "An overview of light-wave packet networks," IEEE Network Magazine, vol. 3, 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, 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, 948-964, August 1991.

[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*, August 1990.

[6] Florissi, D. "Isochronets: a high-speed network switching architecture (thesis proposal)," Tech. Rep. CUCS-020-93, Computer Science Department, Columbia University, 1993.

[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*, 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, August 1991.

[9] Ramaswami, R., "Multiwavelength lightwave networks for computer communication," *IEEE Communications Magazine*, February 1993.

[10] Roth, C., *Fundamentals of logic design*, Fourth Edition, West, 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 1980.

[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*, IEEE, San Francisco, California, USA, April 1993.