

An FPGA-based Network Interface for WDM Gigabit Networks *

David C. Hoffmeister^{1,2}, Patrick W. Dowd^{2,3}

¹Department of Electrical and Computer Engineering, State University of New York at Buffalo

²Department of Electrical Engineering, University of Maryland at College Park

³National Security Agency, Fort Meade, MD

Abstract

A network interface (NI) for wavelength-division multiplexed (WDM) gigabit networks is presented that is based on field programmable gate array (FPGA) technology. This NI is being used for LIGHTNING, a dynamically reconfigurable WDM network testbed project for supercomputer interconnection. The NI is implemented using FPGAs to allow experimentation involving the media access protocol and reconfiguration control for LIGHTNING. This reprogrammability will also allow the NI to be used as a general testbed platform for WDM gigabit networks.

I. INTRODUCTION

Computer systems have developed rapidly over the past few years. Specifically, processor speeds continue to increase but I/O capability continues to lag. This is not a new trend yet no general solution to this perpetual problem has emerged. This paper describes an approach to support high-performance I/O - both interprocessor communication and remote file system access - with a combination of high speed optical network designed specifically for this situation.

The objective is to develop a scalable technique for clustering: a strategy that is effective (both in performance and price/performance) with both workstation-class processor interconnection and high-performance supercomputer-system level interconnection. The interconnection strategy needs to be flexible to adapt to the severe cost constraints at the low-end and performance requirements at the high-end. This paper defines the general architecture, and then defines more specifically an experimental testbed currently under construction known as "LIGHTNING" that is targeted to supercomputer interconnection. This is a joint project involving the State University of New York at Buffalo (Buffalo, NY), University of Maryland (College Park and Baltimore County, MD), Laboratory for Physical Sciences (College Park, MD), Center for Computational Sciences (Bowie, MD), and David Sarnoff Research Center (Princeton, NJ).

The optical network is hierarchical and based on wavelength-division multiple access (WDMA). WDM creates multiple

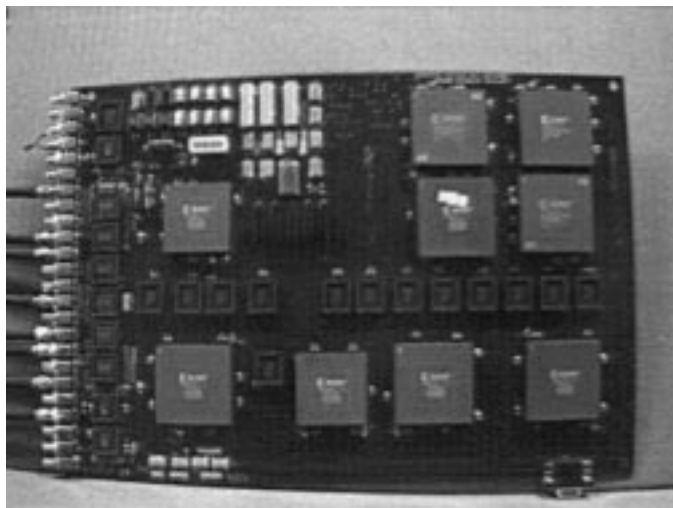


Figure 1: Original prototype card for LIGHTNING.

channels, separated in wavelength, to be concurrently transmitted on a single fiber. The channels can be individually accessed, routed and switched. The architecture can be viewed as a tree, where processors reside at the leaves and the internal nodes have wavelength- and spatial-switching capabilities. This class of architecture for processor interconnection was first described in [1]. Section II. briefly describes the structure.

An advantage of this architecture is that bandwidth can be dynamically re-allocated throughout the system to adapt to shifts in traffic patterns. The bandwidth assigned to different levels of the hierarchical system can be dynamically increased or decreased, based on the reference patterns and file activity of the system. However, in comparison to traditional reconfigurable architectures, the *reconfiguration of the system is automatic and hidden from the user*. The user does not have to logically map an application to a specific topology or inform the operating system at compile- or run-time of its intended communication patterns. LIGHTNING senses the traffic patterns inherent to an application and adapts itself accordingly. This function is hidden from the user and the operating system and avoids placing the burden of understanding the specifics of the system on the programmer. The programmer views the system as a pool of processors and is not involved with process placement. The operating system, during initial process placement and during migration, favors placing a process within the level-1 or level-2

*This work was supported by the U.S Department of Defense, Laboratory for Physical Sciences, College Park, Maryland. The authors can be reached at {dch,dowd}@eng.umd.edu. Preprints and related articles are available via WWW at <http://tebbit.eng.umd.edu/>.

cluster depending on size.

An FPGA-based network interface (NI) has been designed and is being constructed that will be used to construct a gigabit WDM testbed. Figure 1 shows the original prototype card used to verify the media access protocol and optical devices at 250 Mb/s. The goal of the testbed currently being developed is to support a (weakly coherent) distributed shared memory environment between the interconnected processors. This is accomplished through a combination of the network architecture, operating system, and network/memory interface design. The goal is not just to provide a high-bandwidth network, but to deliver that bandwidth to the application rather than wasting it in operating system overhead.

This paper briefly describes the basic architecture and media access protocol of LIGHTNING in Section II. Section III. describes how the media access protocol is supported using the FPGA-based NI. Section IV. describes the components that will be used to build the NI. The paper concludes with a summary of the work presented and a discussion of the reconfigurability of the NI, its application as a general testbed for WDM gigabit networks, and future work planned in this area.

II. LIGHTNING ARCHITECTURE AND MEDIA ACCESS

This section describes the architecture of the LIGHTNING network. First background information is discussed providing a framework for discussion, and then the architecture is defined. The media access protocol is then presented.

The LIGHTNING architecture was influenced by some of the strengths and weaknesses of the Fat-Tree [2]. LIGHTNING retains the advantages of the Fat-Tree structure – a low latency, scalable interconnection – while avoiding some of the limitations through optical interconnects. Optical networks have many advantages over metallic interconnections such as a relaxed bandwidth distance product, large fan out, low power consumption, reduced crosstalk and immunity to electromagnetic noise [3]. Major considerations in determining the best placement for optical connections is the speed mismatch between the optical and electrical components [4] and the performance/cost requirements. Optical fiber has a 30 THz bandwidth [5], much larger than the metallic bandwidth. To simply replace metal connections with optical ones would result in many expensive under-utilized links.

A. Lightning Structure

LIGHTNING has a hierarchical structure in the form of a general m_i -ary tree. The leaves of the tree are processors and the interior nodes are routing elements. The system uses time-, space-, and wavelength-separation to increase the usable bandwidth in the system.

Internal nodes provide wavelength selective routing. The spatial and wavelength optical routing element is denoted as a Lambda Partitioner (Λ_x). The structure of a Lambda Partitioner is shown in Figure 2. The purpose of the Λ_x is to provide a

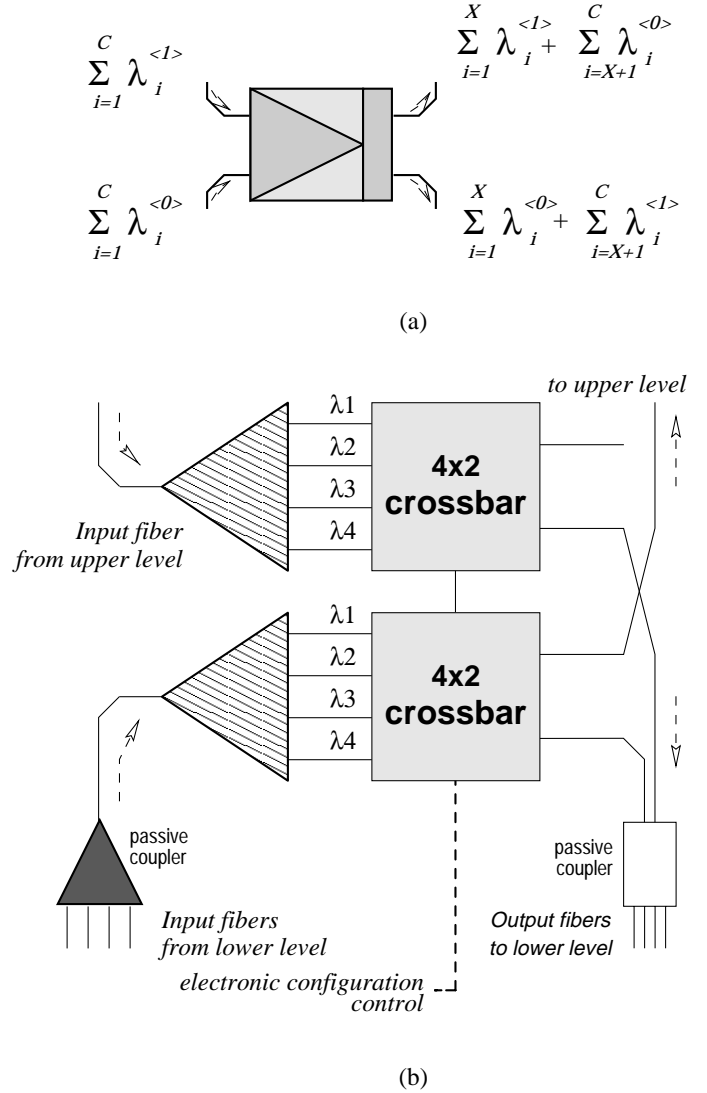


Figure 2: Wavelength partitioning device used to achieve spatial separation of wavelengths. (a) Functional behavior of Lambda Partitioner. Wavelengths above and below the partition point are cross-routed and bar-routed, respectively. Selection of the partition point is electronically controllable by the LIM interface. (b) Structure of lambda partitioner.

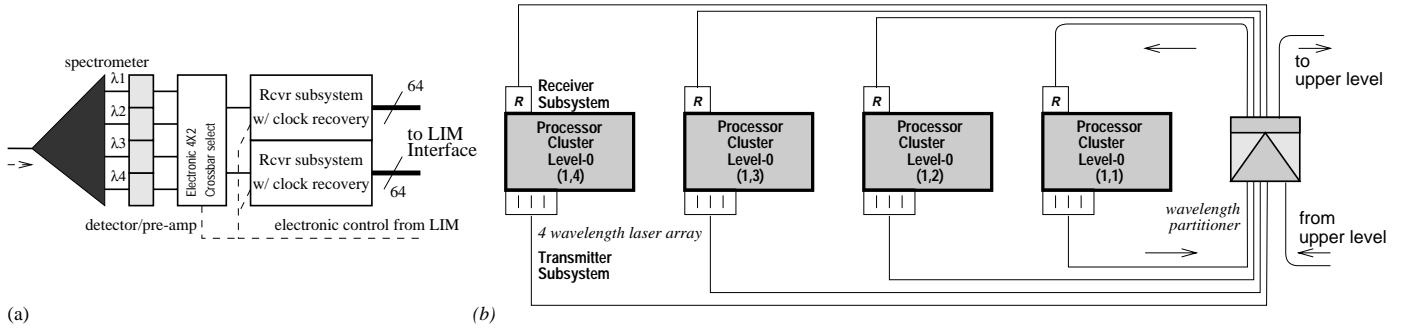


Figure 3: Lightning building blocks. (a) Receiver subsystem where each node can concurrently receive on any two of the wavelengths in this example; (b) A single level example showing that each node contains a receiver described in part (a) and a 4-wavelength laser array and are connected via the wavelength partitioning component. Note that the configuration is illustrated in a star configuration but a bus-type of configuration is possible depending on power budget considerations.

wavelength routing capability where specific wavelengths could be routed to the upper level and other wavelengths could be retained in a lower level. This enables wavelength re-use at peer levels of the hierarchy and reduces the total number of required WDM channels.

A mixed radix system [6] is used to represent the node numbering. Let M (the total number of processors) be a decimal integer represented as a product of r factors:

$$M = \prod_{i=1}^r m_i \quad (1)$$

The processor identifier P , $1 \leq P \leq M$, can be represented as an r -tuple

$$P = (p_r, p_{r-1}, \dots, p_1) \quad (2)$$

where $1 \leq p_i \leq m_i$ for all $1 \leq i \leq r$. The term r denotes the number of hierarchical levels.

An example of a single level network is shown in Figure 3, and a 3-level example is shown in Figure 4. A bus based system is drawn for clarity in Fig. 4 but the actual implementation is a star-coupled based system.

The function of the Λ_x is to serve as a 2×2 switch for each wavelength channel as illustrated in Fig. 2. Denote the number of wavelength channels as C and the ordered set of channels as $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_C\}$. All the wavelengths below the i -level partition point are retained within the i -level cluster while all wavelengths above the partition point are routed to the $i + 1$ level cluster. If the i -level partition point was set at j , then wavelengths $\{\lambda_1, \lambda_2, \dots, \lambda_j\}$ are retained within level i , and $\{\lambda_{j+1}, \lambda_{j+2}, \dots, \lambda_C\}$ are routed to level $i + 1$.

The set of partition points in an r -level system are denoted as $\{X_0, X_1, \dots, X_r\}$. With the partition points staggered at each level such that $X_0 < X_1 \dots X_{r-1} < X_r$, a separate set of wavelength channels are dedicated for communication at each level. Denote C_i as the set of channels allocated for i -level communication, $|C_i| = X_i - X_{i-1}$ where $X_0 = 0$ and $X_r = C$. Denote the collection of processors beneath an i -level node as an i -level cluster. If a processor wants to send a message to another processor in the same level 0 cluster, it uses a channel

in C_0 . If the second processor is not in the same level 0 cluster but is in the same level 1 cluster, a channel in C_1 is used.

The determination of which channel to transmit on is discussed in Section B.. Note that spatial re-use of the wavelength channels in different clusters occurs. A symmetric network would have all the i -level wavelength partitioners in the network set to the same partition point. Alternatively, Λ_x s in spatially separate clusters could partition the wavelengths differently, as long as $X_i < X_{i+1}$. In either case, the partition points can be dynamically changed. This allows the network to adapt to changes in traffic patterns.

The total number of spatial/wavelength channels in the system, denoted as \mathcal{C} , can be calculated to determine the advantage of wavelength re-use. For example, consider a symmetric configuration where $M_j = \prod_{i=j+1}^r m_i$ denotes the number of j -level nodes in a hierarchy, and C_i as the number of channels allocated for i -level communication. Since there are a total of M_j j -level nodes, there is $C_j M_j$ channels providing j -level communication.

Therefore the total number of channels in an r level system is

$$\mathcal{C} = \sum_{j=1}^r C_j \prod_{k=j+1}^r m_k \quad (3)$$

Consider a small system of $M = 4 \times 4$ with $C = 4$. In this example, the total number of effective channels increases to a maximum of $\mathcal{C} = 13$ through wavelength re-use.

B. Media Access Protocol

The wavelength multiplexed environment requires a media access protocol to arbitrate access. With existing upper layer protocols in mind, the media access protocol has been designed to *exploit the bimodal pattern of network traffic*. There have been many studies of *typical* Internet traffic, with TCP/IP, and a common trait is the bimodal distribution of packet size: essentially all of the traffic consists of either big packets or small packets with little traffic in between [7,8]. Batched applications, such as ftp, tend to use large data packets to get high throughput while

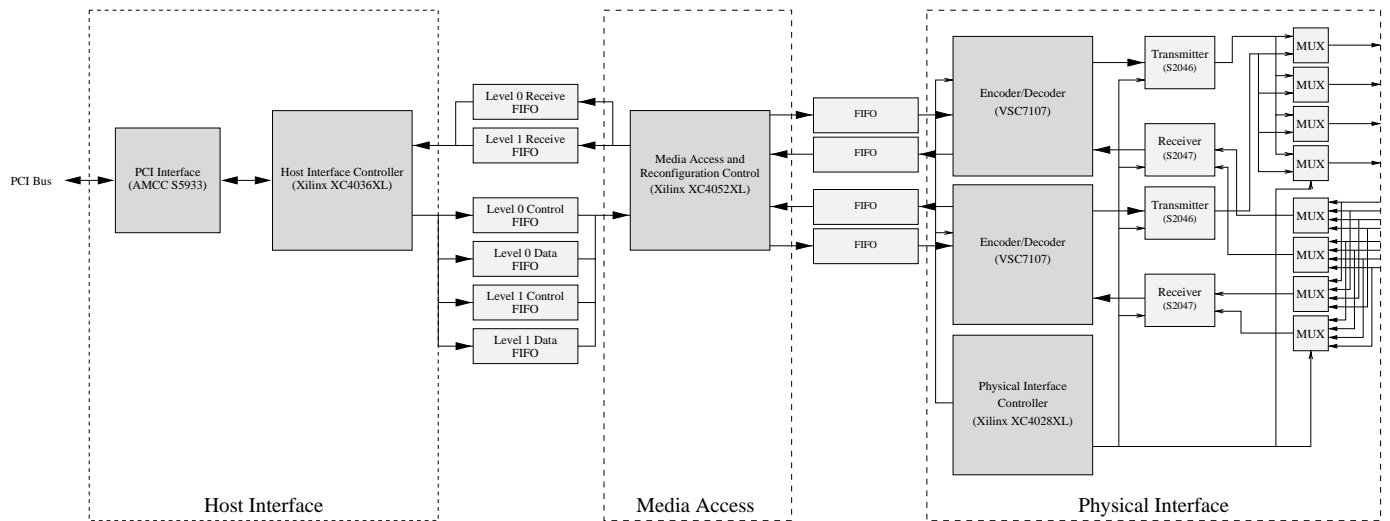


Figure 6: Block diagram of the network interface.

interactive applications, like telnet, tend to use small packets for low latency.

This characteristic is also true with distributed shared memory computer systems. The traffic, generated by a memory coherence protocol needed to achieve DSM, has two major forms: memory *consistency control packets* (such as memory block requests, invalidations, acknowledgments) and *memory block packets* [9]. Memory consistency control packets are very small while the memory blocks can be up to 8 Kbytes. In a non-broadcast scheme there can be multiple memory consistency control packets for every memory block that needs to be transferred.

The media access protocol has been designed to exploit this characteristic. The media access protocol supports two different packet sizes. Small packets called *control packets* are used to reserve access to the network. These packets are only 64 bytes long, but they are also capable of carrying a small payload of 32 bytes. This can be used for transferring small packets from applications as well as the memory consistency control packets necessary to support DSM. Control packets reserve access for sending large packets referred to as *data packets*. These are 8 Kbytes long and can carry large packets from applications as well as memory block packets.

The protocol combines reservation access and pre-allocated receivers. Typical reservation style WDMA protocols use one of the channels as a dedicated control channel [10]. However, the number of available WDM channels usually is small and there are usually many more nodes than number of channels. In such a situation, reserving one channel as the control channel is not very attractive. The media access protocol for LIGHTNING is a hybrid approach using reservation access but without a reserved control channel. Furthermore, it was designed specifically for the situation where each node had a fast-tunable transmitter (based on a laser-array) and a slow tunable receiver.

The protocol reserves access on a channel through control packets. Transmission on each channel consists of two phases:

the reservation control block (RCB) which is followed by the memory block transmission phase. Figure 5 shows the allocation map for a two level media access protocol where each level has two channels. The control packets are broadcast (on the subset of channels assigned to the level of the target node) while the data blocks can be sent on any channel available to that level, regardless of destination. A data packet causes the data cycle to be extended by a slot only when all channels in the previous slot are full or if there is a destination conflict. The only channel conflict that can occur during a data slot is when two or more nodes target the same destination. High utilization is maintained by having the second node transmit in the next data slot. Note that this enables a collisionless environment but significantly reduces the overall cycle time, when compared with TDMA, since data block slots are only assigned when they are needed. This significantly reduces the latency since Figure 5 is not drawn to scale and the data blocks are typically more that two orders of magnitude larger than a control slot.

The laser-array transmitter facilitates broadcast by simultaneously transmitting a control packet on all channels assigned to the target level of a system. This enables LIGHTNING to obtain the advantages of a reservation-based protocol without requiring a WDM channel to be a dedicated control channel. This is especially important in a multi-level architecture since one control channel would have to be assigned per level. A basic assumption and constraint on LIGHTNING is not to require a large number of WDM channels so LIGHTNING has no dedicated control channels and only uses the channels in that mode for a small fraction of the cycle duration.

C. Reconfiguration of the Network

One of the unique features of the LIGHTNING architecture is the ability to change the number of channels assigned to each level of the hierarchy. This provides a utility for increasing the bandwidth to levels with higher traffic loads thereby exploiting tem-

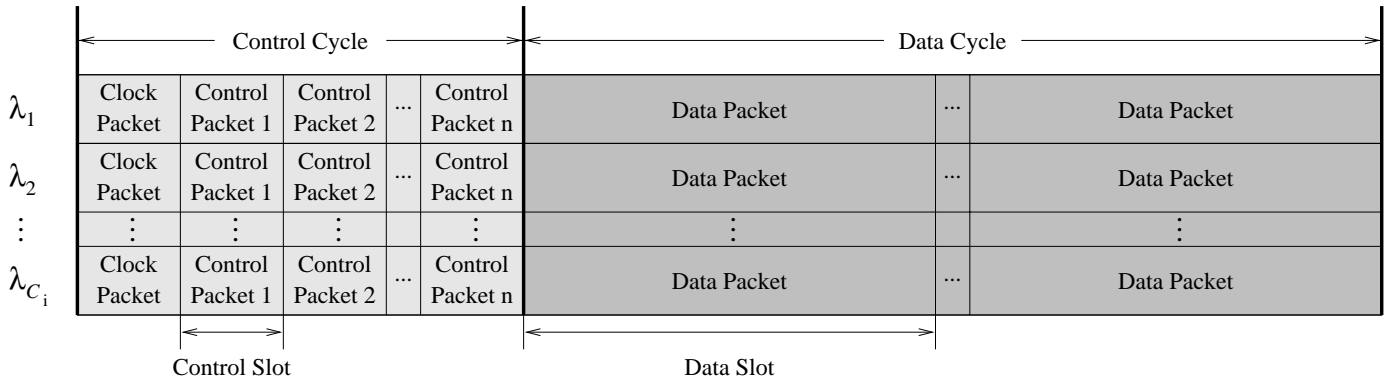


Figure 7: Control and data cycle for one level. The clock and control packets are broadcast on all C_i wavelengths. Data packets are sent on individual wavelengths. (NOTE: Figure is not drawn to scale. Data packets are several orders of magnitude larger than control packets.)

poral locality. Increasing the bandwidth would lead to a reduction in overall system latency.

This is accomplished by monitoring the traffic patterns of the system, in a decentralized fashion, with dynamic reallocation of system bandwidth when an imbalance occurs. Monitoring occurs at the media access level - *operating system intervention is not required*. The monitoring routines measure the traffic intensity of each level and detect imbalances. When an imbalance is detected, the routines decide on the best course of action and initiate the system resource reallocation. Close interaction between the reconfiguration routines and the media access protocol is required to ensure proper operation of the system and to place no additional burden on the system.

Due to space constraints the full details of the reconfiguration algorithm has been omitted from this work. For a more detailed description refer to [11, 12].

III. NI FUNCTIONAL DESCRIPTION

A block diagram of the NI is shown in Figure 6. The NI is made up of three main sections: the host interface, the media access and reconfiguration control (MARC), and the physical interface. The host interface connects the NI with the host computer through the system bus. The MARC section performs the scheduling and synchronization necessary to access the physical medium, as well as monitoring the system traffic and reconfiguring bandwidth when necessary. The physical interface handles the encoding/decoding and parallel-serial-parallel conversions necessary to interface with the optical transceivers.

A. Host Interface

The host interface connects the NI with the host computer through the system bus. For this testbed implementation the host computer will be a PC and the system bus will be PCI. The interface to the PCI bus will be handled by a commercially available PCI chip (see Section A.). The host interface controller arbitrates access to the local side of the PCI chip.

The hierarchical nature of LIGHTNING requires a separation of traffic to each level of the hierarchy. For this testbed a two level system will be implemented. The host interface controller must therefore determine to which level an incoming packet is destined. The host interface controller is also responsible for multiplexing the two incoming traffic streams into one stream that can be sent to the host via the PCI bus.

The media access protocol also requires a separation between small control packets and larger data packets. This distinction is also made by the host interface controller. Once a packet has been identified by both type and level the host interface controller stores it in the appropriate FIFO, as shown in Figure 6, to wait for an available slot in the media access cycle.

B. Media Access and Reconfiguration Control (MARC)

The MARC section of the NI is responsible for synchronizing the NI to the other nodes in the system, processing incoming packets, scheduling the transmission of control and data packets, and monitoring the system traffic to control bandwidth allocation. These functions act in parallel to ensure smooth operation of the network.

1) Synchronization

The NI currently uses a distributed clock mechanism to maintain synchronization between nodes. Since each level operates independently a separate clock is kept for each level. This clock is distributed by a clock node on each level. To increase the fault tolerance of the system any node can assume the responsibility of the clock node if it does not detect one on the level.

At the beginning of every control cycle the clock node begins by sending a clock packet as shown in Figure 7. This packet contains the clock node's current time at the beginning of the control cycle, as well as the propagation delay from the clock node to the Λ_x on that level. All nodes periodically measure the propagation delay through the system on each level and estimate

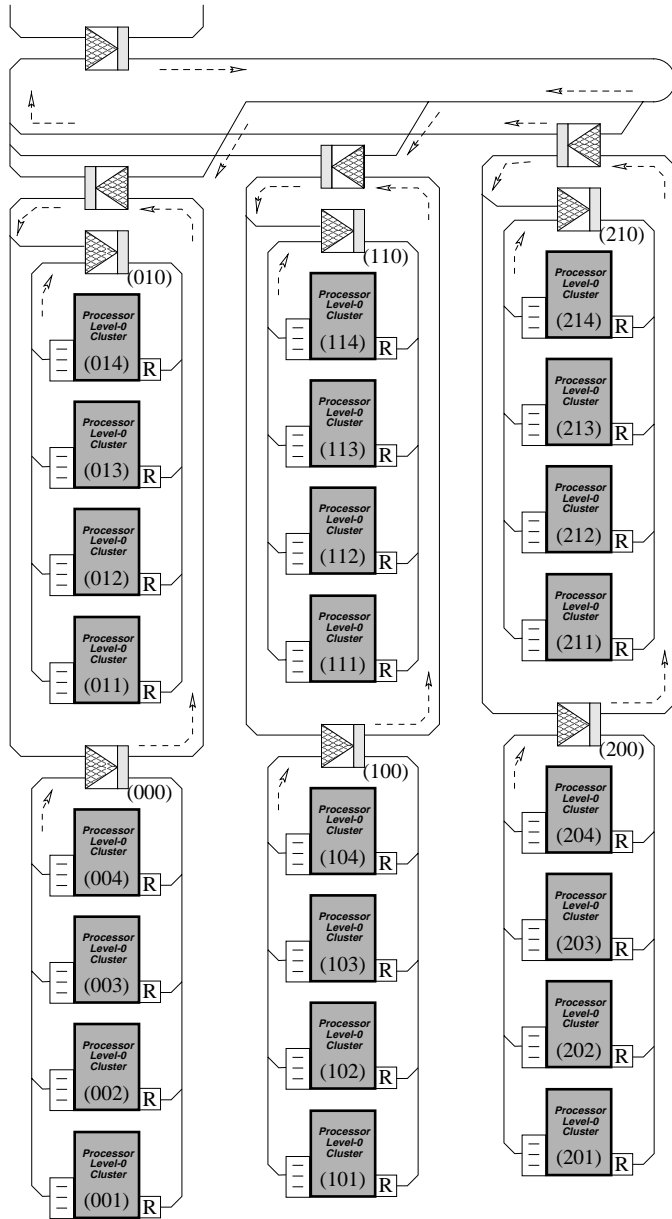


Figure 4: Example of a 3-level Lightning system. A bus based system is shown for clarity but implementation is closer to a star-couple configuration.

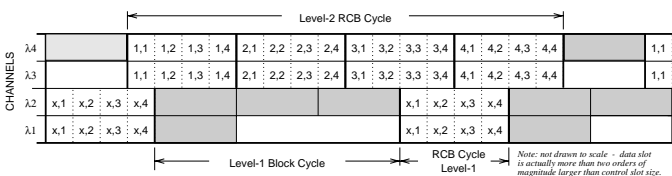


Figure 5: Assignment map for a two level media access protocol for $M = 4 \times 4$ and $C = (2, 2)$

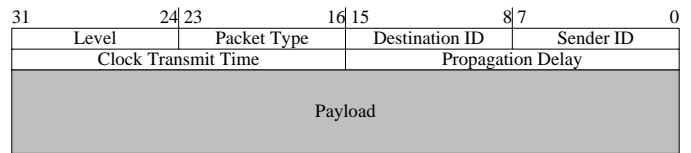


Figure 8: Packet format for clock packets. As indicated in the figure, clock packets can also be used to carry a small payload for network control information.

the delay to the λ_x . The packet format for a clock packet is shown in Figure 8.

Using the information contained in the clock packet the other nodes on the level can determine the value their clock should contain given the time the clock packet is received. By adjusting the local clock to match this value each node can correct any errors in synchronization. This allows all nodes to transmit in their allocated slots with only a small amount of padding on each end of the slot for synchronization errors.

2) Packet Processing

Each node in the system must examine all control packets received on each level in order to keep track of reservations made for the data cycle on that level. A reservation packet will indicate the source node as well as the destination. In this way the MARC can build a map of the data cycle. Once the data cycle is constructed and begins, the data packets received are examined to determine the destination.

A reservation packet is a control packet with a special packet type to indicate that a data packet will be transmitted in the next data cycle. The packet format for control packets is shown in Figure 9. A node builds the data cycle only from reservation packets that are received, including its own. A node does not record the fact that it sent a reservation packet, only that it received one from itself. This provides a uniform way of handling all reservations.

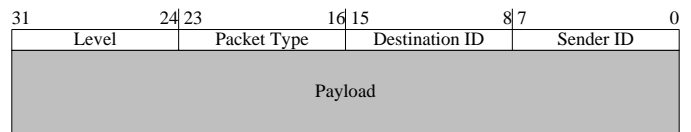


Figure 9: Packet format for control packets. As indicated in the figure, control packets can be used to carry a payload as well as reservation information to support small packet transfers.

When a reservation packet is received the source and destination tags contained in the header are examined. These tags will determine what actions must be taken. There are three possible transactions that can occur when a reservation packet is received: a general data packet is scheduled that does not concern the current node, a data packet is scheduled to be received by the current node, or a data packet is scheduled to be transmitted by the current node. A block diagram of the hardware

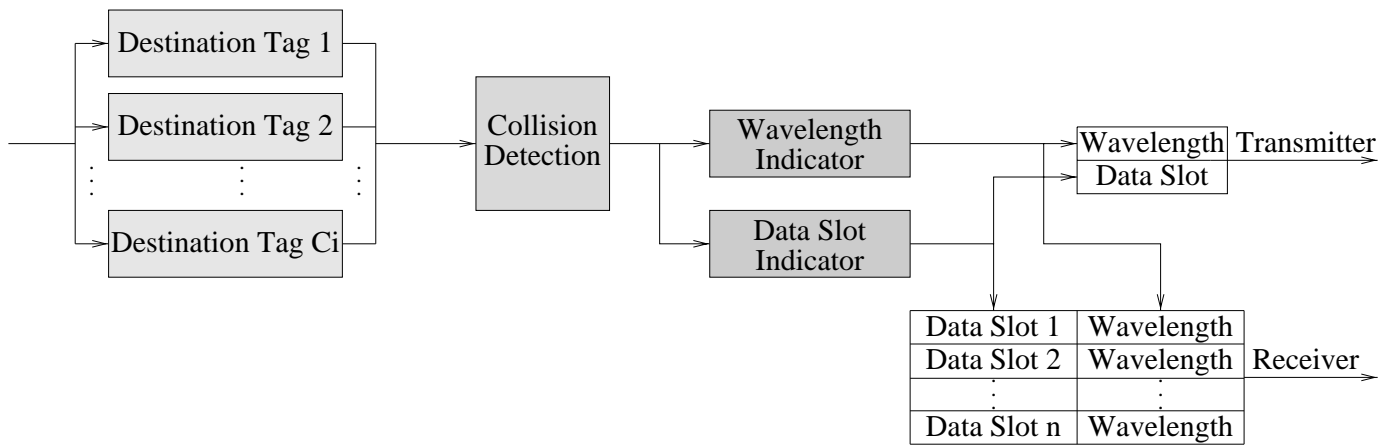


Figure 10: Block diagram of the reservation hardware. Receiver collisions are detected, the receiver’s wavelength is recorded for each data slot, and the data slot and wavelength are recorded when transmitting a data packet.

for handling reservations is shown in Figure 10 as a reference for the following descriptions.

General Data Packets All reservations for data packets, even those that do not originate from and are not destined to the current node, require a certain amount of processing. Each node keeps track of the destinations of the most recent C_i data packets, where C_i represents the number of wavelengths assigned for i -level communication. As reservations are received these destinations are compared to detect receiver collisions. Receiver collisions occur when two data packets destined to the same node are scheduled in the same data slot. Since LIGHTNING uses a collisionless protocol these possible receiver collisions must be detected and avoided.

The current method for avoiding these receiver collisions is very simple. When a possible receiver collision is detected the current data slot is considered filled regardless of how many wavelengths are being used. While this causes a reduction in bandwidth utilization it allows the media access protocol to operate fast enough to keep up with the steady stream of reservation packets without having to buffer reservations, thereby possibly causing a processing delay. Since the MARC is implemented in a FPGA it will be possible to experiment with more efficient scheduling methodologies in the future. The most efficient method would be to sort all of the reservations for a cycle to get the highest possible bandwidth utilization.

If there are no receiver collisions in a data slot the data packets are simply assigned to the C_i wavelengths on a first-come-first-serve basis. When all of the C_i wavelengths are assigned for a data slot the data cycle is extended by one data slot and the process is continued until the end of the control cycle. This process occurs for all data packet reservations and so will not be repeated in the following descriptions.

Receiving Data Packets When the destination tag in a reservation packet is the same as the current node certain information must be recorded so the node can receive the data packet.

The position of the current data slot in the current data cycle is known at all times. When a reservation is received it is assigned to the next available wavelength. These two pieces of information together allow a node to tune its receiver to the correct wavelength in the correct data slot to receive its intended data packet.

Since the level’s receiver must always be tuned to **some** wavelength, the only piece of information that must be recorded is the wavelength that it will be tuned to in each data slot. The default wavelength is the first available wavelength for that level. When a reservation packet is received for a data packet destined to the current node the wavelength for the current data slot is changed to the currently available wavelength. When the data cycle commences the receiver is tuned to the recorded wavelength for each data slot in turn. Since the data packet in that slot may not be destined to the current node it is still examined and the destination tag compared to the current node. Data packets not destined to the current node are simply dropped. This does not put any undue burden on the NI since only one data packet per data slot must be examined at each node.

Transmitting Data Packets When the source tag in a reservation packet is the same as the current node certain information must be recorded so that the current node can transmit its data packet. This is very similar to receiving data packets since the node must record which wavelength will be used and in which data slot the data packet must be placed. The main difference is that each node is allowed to send only one reservation packet per control cycle and, therefore, only one data packet per data cycle. Since other nodes will be transmitting their data packets in the other data slots it is not possible for a node to simply always be transmitting on some wavelength. Thus, both the data slot and the wavelength are recorded and used to transmit the data packet at the appropriate time on the appropriate wavelength. This will ensure that the packet does not collide with another data packet.

3) *Reconfiguration Control*

Each node in the system is responsible for monitoring the traffic on each level of the network. The relative traffic intensities are then compared to identify a level of the network that requires more bandwidth. If such a condition exists the node notifies the rest of the system and reconfigures the allocation of wavelengths to correct the situation. Once this is accomplished the system resumes normal operation. For more details on reconfiguration see [11, 12].

C. *Physical Interface*

The physical interface is responsible for converting the parallel stream of data from the MARC into a coded serial stream for the optical transceiver, and vice versa. This includes 8B/10B encoding/decoding, parallel-serial-parallel conversions, bit synchronization, and frame recovery.

The interface between the MARC and physical interface consists of one FIFO per level per direction. For this implementation there are two transmit FIFOs and two receive FIFOs. Data is taken from the transmit FIFOs and sent through a commercially available 8B/10B encoding chip for use in gigabit Fibre Channel and Gigabit Ethernet systems. These chips accept 32 bits of parallel data and encode it into four 8B/10B encoded words. These are then sent to a commercially available Fibre Channel transmitter that performs the parallel to serial conversion and transmits the data in serial format.

The receiver is also a commercially available Fibre Channel receiver chip that performs both clock recovery and frame synchronization on the incoming data. The data is then sent to the 8B/10B decoder, which outputs 32 bit words of data to the receive FIFOs.

The physical interface controller is implemented in an FPGA. The controller selects the wavelengths for both the transmitter and receiver on each level and controls the flow of data to/from the FIFOs that connect the physical interface to the MARC. Using an FPGA to implement the controller will provide flexibility in interfacing the various components and will not limit the physical interface to one specification.

For our particular application it may be necessary to achieve clock recovery faster than the commercially available chips allow. To facilitate this we have also included a clock input for each data input to the receiver. This allows us to perform clock recovery externally and provide both clock and data to the receiver chip using a special test mode of the chip. This will give us the ability to experiment with novel clock recovery techniques for gigabit packet switched networks.

IV. NI COMPONENT DESCRIPTION

All of the components that will be used in the NI are commercially available chips. These include a PCI interface chip, several FIFOs, three FPGAs, 8B/10B encoder/decoder chips, Fibre Channel transmitters and receivers, as well as assorted line drivers and a programmable clock generator. This section describes the main components and the specific functionality that

will be used in the NI. The specific manufacturer and model number that will be used in the testbed are given. In most cases there are several different products that are suitable.

A. *PCI Interface*

The PCI interface is the S5933 from AMCC. [13] This chip handles 32-bit PCI buses at 33 MHz. The local bus interface is very versatile and allows a pass-thru, FIFO, and mailbox type interface. This allows the greatest freedom in accessing the NI. The S5933 also supports burst reads and writes which will be used to transfer data packets between the card and the main memory of the host. There is also a freely available driver for Windows NT from Blue Water Systems. [14] This will make it easier to debug and test the interface.

B. *Host Interface Controller*

The host interface controller is comprised of a Xilinx XC4036XL-1 FPGA. [15] This chip was chosen for its size and speed capability. Since the PCI bus is limited to 33 MHz this FPGA is able to operate at the same speed as the PCI interface. At a maximum typical gate count of 65,000 gates it is capable of providing all of the functionality needed for this part of the design. [16] It also provides the freedom to specify the card layout and connections without completely designing the functionality of the chip.

C. *Media Access and Reconfiguration Control (MARC)*

The MARC is comprised of a Xilinx XC4052XL-1 FPGA. [15] This chip was also chosen for its size and speed capability. At a maximum typical gate count of 100,000 gates it is capable of providing all of the functionality needed for this part of the design. [16] It also provides the freedom to specify the card layout and connections without completely designing the functionality of the chip.

Much of the functionality of this chip is comprised of state machines. The various state machines operate completely in parallel, unlike the context switching that would be necessary with microprocessors. The program for the FPGA is stored in an EPROM. The part can be completely reprogrammed an almost infinite number of times. Not only can errors be corrected but new features can be added, as well as experimentation with completely new functionality.

D. *Physical Interface Controller*

The physical interface controller consists of a Xilinx XC4028XL-1 [15] with a maximum typical capacity of 50,000 gates. [16] This part, like those mentioned above, allows the physical interface to be flexible and operate at the required speed. It also allows the various levels and directions of traffic to operate completely in parallel, thus avoiding any bottlenecks.

E. 8B/10B Encoding/Decoding Chips

The line encoding and decoding is performed by a pair of Vitesse VSC7107. [17] These are 8B/10B encoder/decoder chips for Fibre Channel serial links. They operate up to a 1.0625 Gb/s serial data rate over copper or optical links. This allows an actual data rate of about 800 Mb/s.

F. Transmitter and Receiver Chips

The transmission and reception of the serial data stream is handled by AMCC S2046 and S2047 chips, respectively. [13] These chips interface easily with the encoder/decoder and provide transmission and reception up to 1.25 Gb/s. However, since the encoder is limited to 1.0625 Gb/s that will be the data rate employed.

G. Programmable Clock Generator

The host interface operates at the PCI bus frequency of 33 MHz. In order to obtain the maximum speed for the Fibre Channel chipsets, 1.0625 Gb/s, a clock frequency of 53.125 MHz is used to clock the Fibre Channel chips, which then provide a clock of 26.5625 MHz to clock the physical interface controller. However, since the MARC section of the NI is interfaced through FIFOs to the host and physical interfaces, the rest of the NI is clocked by a programmable clock generator. This allows the use of any of 16 different frequencies. This was done to provide slow clock frequencies during debug and testing, while still allowing the use of the full speed once testing is complete. During normal operation the MARC section will be able to operate at either 33 MHz to maintain PCI speed throughput to the host interface, or at 26 MHz to match the speed of the physical interface. At either speed it will be necessary for the design to allow for the difference in speed to keep from overloading the slower interface.

V. CONCLUSIONS

This paper described a network interface for use in gigabit WDM networks. The use of FPGAs as the main processing elements of the card allow it to be used as a general testbed for WDM media access protocols and network control. It is currently being used to implement the network interface for project LIGHTNING. The flexibility provided by FPGA technology allows experimentation with the network control algorithms.

The goal of this NI design is to support the LIGHTNING testbed. However, using FPGAs as the main processing elements of the media access controller has resulted in a versatile and reprogrammable NI that could support many different topologies for gigabit WDM networks. Even using only one half of the two level system one could implement any WDM topology that requires a single tunable transmitter and a single tunable receiver. This would include many different types of star-coupled networks, rings, meshes, and other topologies.

The only limit would be the capacity of the FPGA, which in this case is very large.

By utilizing the second half of the two level system many other novel WDM systems could be implemented. Multi-star systems, multiple rings, trees, and other topologies would be possible. This capability provides a great opportunity for exploration of many different topologies that has never been possible before due to the cost and complexity of building all of the necessary hardware. By using FPGAs it is possible to create a whole new network simply by redesigning the contents of the FPGAs themselves. While this is still a challenging task for many architectures, it is much more feasible than building new NIs for each topology. Future work will center on using this flexibility to explore various network reconfiguration algorithms.

REFERENCES

- [1] P. W. Dowd, K. Bogineni, K. A. Aly, and J. Perreault, "Hierarchical scalable photonic architectures for high-performance processor interconnection," *IEEE Transactions on Computers*, vol. 42, pp. 1105–1120, Sept. 1993.
- [2] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Transactions on Computers*, vol. c-34, pp. 892–901, Oct. 1985.
- [3] L. Hutcheson, "High speed optical interconnects for computing applications," in *Proc. SPIE*, vol. 862, pp. 2–10, 1987.
- [4] P. W. Dowd, "Optical bus and star-coupled parallel interconnection," in *Proc. 4th International Parallel Processing Symposium*, (Los Angeles, CA), pp. 824–838, Apr. 1990.
- [5] C. A. Brackett, "Dense wavelength division multiplexing networks: Principles and applications," *IEEE Journal on Selected Areas of Communications*, vol. 8, pp. 948–964, Aug. 1990.
- [6] L. Bhuyan and D. P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Transactions on Computers*, vol. c-33, pp. 323–333, Apr. 1984.
- [7] J. Mogul, "Observing TCP dynamics in real networks," in *Proceedings of ACM SIGCOMM*, pp. 305–317, Oct. 1992.
- [8] K. Claffy, G. Polyzos, and H. Braun, "Traffic characteristics of the T1 NSFNET backbone," in *IEEE INFOCOM'93*, pp. 885–892, Mar. 1993.
- [9] P. W. Dowd and J. Chu, "Photonic architectures for distributed shared memory multiprocessor systems," in *IPPS'94 Workshop on Massively Parallel Processing Through Optical Interconnects*, (Cancun, Mexico), IEEE, 1994.
- [10] I. M. I. Habbab, M. Kavehrad, and C. E. W. Sundberg, "Protocols for very high-speed optical fiber local area networks using a passive star topology," *IEEE Journal on Lightwave Technology*, vol. LT-5, pp. 1782–1793, Dec. 1987.
- [11] P. Dowd, J. Perreault, J. Chu, D. Hoffmeister, R. Minnich, D. Burns, F. Hady, Y.-J. Chen, M. Dagenais, and D. Stone, "Lightning Network and Systems Architecture," *IEEE/OSA Journal of Lightwave Technology*, vol. 14, pp. 1371–1387, June 1996.
- [12] K. Li, Y. Pan, and S. Q. Zheng, eds., *Parallel Computing Using Optical Interconnections*, ch. 1, pp. 3–23. Kluwer Academic Publishing, 1998.

- [13] Applied Micro Circuits Corporation, 6195 Lusk Blvd., San Diego, CA 92121.
- [14] Blue Water Systems, Inc., 190 West Dayton, Suite 202, Edmonds, WA 98020.
- [15] Xilinx, Inc., 2100 Logic Drive, San Jose, CA 95124-3400.
- [16] XILINX, ed., *The Programmable Logic Data Book*. XILINX, 1998.
- [17] Vitesse Semiconductor Corporation, 741 Calle Plano, Camarillo, Ca 93012.