Invited Paper

Error and Flow Control Protocols for Terabit Optical Networks

Ted Szymanski

Department of Electrical and Computer Engineering

McMaster University, Hamilton, Ontario, Canada L8S 4K1

ABSTRACT

The design of an optical image guide network for distributed multiprocessing is described. The network supports multiple high bandwidth rings between workstations over distances of 10s of meters. Traditionally, error and flow control functions for multiprocessor networks are implemented in custom high speed electronic *Application Specific Integrated Circuits* (ASICS) which are physically removed from the interconnect's physical layer. In this paper, we consider migrating these functions directly into the optoelectronic physical layer, yielding an "Intelligent Optical Network". Conventional error control protocols are infeasible with dense bit parallel optical systems based on image guides since they require excessive amounts of hardware. The designs of efficient error and flow control protocols for such networks are proposed and analysed. The key blocks of the protocols have been designed, fabricated and demonstrated in 0.8 micron and 0.5 micron CMOS/SEED devices. The protocols require significantly less hardware than alternative schemes, and CMOS/VCSEL devices supporting these protocols are scalable to very high bandwidths, i.e., 10s of Terabits per second.

Keywords: optical networks, image guides, 2D fiber ribbon, error control, flow control, VLSI, multiprocessing

1. INTRODUCTION

Optical interconnects have the potential to provide Terabits of low latency bandwidth for computing and telecommunication systems. These systems can thus alleviate the bandwidth bottlenecks beginning to appear in conventional electronic packaging hierarchy. One attractive optical system is an "Intelligent Optical Network" linking several workstations as shown in Fig. 1. The network consists of a collection of workstations, each containing an embedded optoelectronic smart pixel array, which are interconnected in a ring topology using bit parallel optical image guides. The image guides support multiple bit-parallel optical rings over distances of 10s to potentially 100s of meters, yielding an optical "multi-ring" network. The Smart Pixel Arrays (SPAs) provide an efficient interface between the electronics within the workstation and the multiple optical channels in the network. The SPAs also realize the "intelligence" of the system, typically by providing basic communication protocols directly within the SPA at the physical layer. This optical multi-ring network is architecturally similar to a free-space optical backplane architecture[23,24,25] being developed in Canada by the Canadian Institute for Telecommunications Research[6], with the exception that our optical links use fiber optic image guides or 2 dimensional parallel fiber ribbons rather than free-space links. We envision that such an optical multi-ring network, using state-of-the-art CMOS/VCSEL technology, can be operated at optical clock rates of 1 - 10 GHz, and support Terabits of bandwidth. The communications between workstations would be limited primarily by the speed-of-light latency over the fiber image guides between the workstations. The time-of-flight latency for a 50 meter ring would be approx. 250 nanoseconds, and the latency within the smart pixel arrays will be a small fraction of this at a clock rate of 10 GHz. To achieve this objective, the smart pixel arrays within the workstations must be designed to be exceptionally fast, which is the focus of this paper.

Error control[3,4] is an important function in a real computing system, as a single undetected bit error can contaminate an entire computation. Traditional computing systems often implement error control in custom high speed Application Specific Integrated Circuits (ASICs) at higher protocol layers physically removed from the physical (optical transmission) layer.

In the literature, it is often argued that BERs of 10^{-15} are sufficiently small to ignore error control. We disagree with this view, and argue that system BERs of the range of 10^{-30} may be necessary within a few years, for very high performance computing systems. Furthermore, we argue that error control will grow in importance as optical interconnects are integrated into real systems. The U.S. *Accelerated Strategic Computing Initiative* is aiming to develop machines with 100 TeraFlops per second of computing power by the year 2003, for testing its nuclear arsenal. Given the exponential growth in computing system performance, PetaFlop scale computing systems should be feasible within years. A computation with 1 Petaflops/sec performance would process 10^{17} bits per second (assuming 10^2 bits per Floating Point Operation or *Flop*). At any one time, assume that 10 % of the processed bits are being transmitted over optical data links. The end results of such a computation must be reasonably error-free; suppose the end-result is *acceptable* if the probability of an error in a numeric result is less than 10^{-15}; equivalently, the probability of any bit error occurring during the computation is less than 10^{-15}. Therefore, for a 1 second computation to be *acceptable* the system-wide BER must be restricted to 10^{-31} or less (10^{17-1} bits transmitted per second $\times$ 10^{-31} bit errors per bits transmitted = 10^{-15} bit errors per second). Furthermore, if a computation lasts for 10^4 seconds, the system-wide BER must be restricted to 10^{-35} or less, to achieve the same confidence in the end result. The previous analysis establishes that *system-wide* error control must restrict the BERs to the range of 10^{-35} or less, for 1 PetaFlop machines. In a system with hundreds (10^2) of independent data links, the system wide BER is the cumulative sum of all link BERs, by the statistical independence of the links. Therefore, it follows that the individual link BERs should be in the range of 10^{-35} or less. *In summary, the traditional view that a BER of 10^{-15} is sufficiently small to ignore errors is not valid in high performance computing systems. Our analysis indicates that BERs will have to be far lower than traditionally believed for high performance computing systems.*

Traditional error control schemes often employ "*forwarderrorcorrection*" (FEC) and spectral domain computational methods, e.g., Reed-Solomon codes based on Fast Fourier Transforms[3,14,15]. Popular FEC schemes such as linear block codes, Reed-Solomon codes, and convolution codes[14] offer excellent error correction capability in the presence of random as well as burst errors. However, these techniques are not necessarily well suited for high bandwidth 2D bit-parallel optical systems, since they require a significant amount of hardware to implement. For example, it has been shown in ref. 14 that given an input BER of 10^{-4}, a required output BER of 10^{-12}, a state-of-the-art parallel spectral Reed-Solomon decoder to process a 77 Gb/s data link will require a special purpose ASIC with a chip area of 1cm^2 using a 1 *micron* CMOS process. If follows that a decoder for a 1 Terabit/sec optical data link will require 13 of these ICs, which would be prohibitively expensive. Furthermore, these ASICs only lower the BER to 10^{-12} which is unacceptable for a high performance computing system. Hence, even as CMOS technology improves, traditional FEC error control codes for high bandwidth optical data links will be very expensive and likely intractable.

Error and flow control functions can be implemented in special purpose ASICs or in Field Programmable Gate Arrays (FPGAs). FPGAs are significantly slower than smart pixel arrays, which can implement logic in high speed CMOS. In addition, when error control is performed off the SPA chip, numerous unnecessary delays will be encountered. For example, upon reception of a packet the data must be transferred from the receiving SPA to the external IC where it is processed for error control, requiring an intra-IC data transfer step. If an error is detected the external IC must create and transfer a special protocol packet requesting a packet retransmission back to its local SPA IC, requiring another intra-IC data transfer step. The SPA IC must transmit this protocol packet over the optical network back to the sender SPA, requesting a packet retransmission. The sender SPA IC must transfer the protocol packet to its external IC, which processes the protocol packet and determines a retransmission is required. The external IC sends a copy of the original packet back to its SPA and initiates the retransmission. This process repeats itself until the packet is received without error at the receiver. Each of these intra-IC data transfer steps can incur significant additional delays and cause a significant performance degradation. In addition, each of these additional ICs increases the cost of the system.

Our analysis indicates theoretically that including error and flow control in external ICs or FPGAs can reduce the system bandwidth and increase the delay substantially. Our analysis also indicates theoretically that without effective error and flow control, an optical network for a high performance computing system may not be much faster than a lower bandwidth electrical network which addresses error and flow control. Our analysis is supported by detailed system simulations, written in both the VHDL hardware description language and the C programming language.

Our proposed solution is to migrate the error and flow control protocols directly into the smart pixel arrays for the optical network, yielding an "*Intelligent Optical Network*". Such a system includes *intelligence* at the physical layer, and can *process* vast amounts of optical data as it moves through the physical (optical) layer. This processing of vast amounts of data occurs within the smart pixel array, before the data is transferred "off chip" through electronic I/O pads where it incurs a delay and undergoes a significant bandwidth reduction. This class of "smart interconnects" thus represents a departure from the traditional view of "*All-Optical*" or "*Passive Optical Networks*" networks, where very limited processing is performed on vast amounts of optical data in the all-optical layer. The smart pixel arrays for the proposed intelligent optical network can be constructed using the hybrid CMOS/SEED or CMOS/VCSEL technologies. Alternatively, the smart pixel arrays can be constructed using a high bandwidth multi-chip module linking a CMOS substrate with a VCSEL array and a photodetector array. In the near term the multi-chip module approach may be preferred, due to the availability of the components. However, within a few years we envision the availability of CMOS substrates with dense VCSEL I/O.

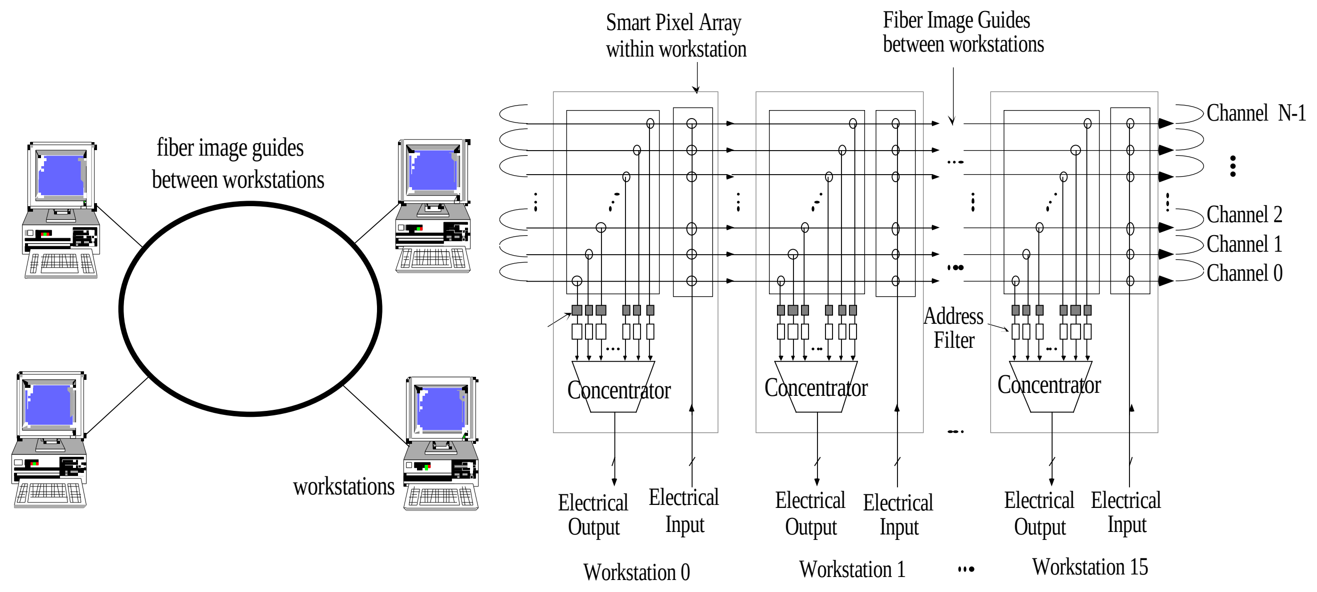


Fig 1. (a) Conceptual view of an optical multi-ring interconnecting several workstations. (b) Broadcast-and-Select architecture with pipelined optical channels.

In this paper, we present the design of an intelligent optical multi-ring network with integrated error and flow control. The system can achieve very low undetected bit error rates as desired by the system designer, i.e., BERs as low as 10^{-30} are achievable after error control. We note that our proposed backplane differs from traditional *All-Optical* and *Passive-Optical-Networks*, since our system includes the error and flow control protocols directly within the SPAs at the "optical layer". *All Optical* networks defer all but the simplest of processing of optical data to external ASICS at a higher level in the protocol hierarchy; such networks typically perform only packet address recognition in the optic domain[8]. While *All-Optical* networks can support very high bandwidths, they cannot perform significant processing of vast amounts of optical data. Similarly, *Passive Optical Networks* are completely passive as their name implies, and they also defer all processing to external ASICs, i.e. see ref. 7. Both of these types of networks must perform error and flow control in external electronic ASICs. However, our error control protocols are still applicable to *All-Optical* and *Passive-Optical-Networks*. With these technologies, all processing must be performed in external ICs. Our proposed error control protocols can still be used in these networks, because they process much more optical bandwidth per unit of silicon area when compared to traditional FEC controls as described in[14].

This paper is organized as follows. Section 2 provides a system overview. Sections 3 and 4 describes the Error and Flow control functions. Section 5 describes the VLSI design of these functions in 0.8 and 0.5 micron CMOS. Section 6 describes projections for the performance of this system over the next decade.

2. SYSTEM OVERVIEW

The structure of a intelligent optical network is shown in fig. 1a; see refs. 23,24,25 for descriptions of architecturally similar systems. The optical layer can implement a *broadcast-and-select* multiple ring system, with multiple optical broadcast channels interconnecting *16* workstations, as shown in Fig. 1b. Each optical channel can be viewed as a slotted ring or as a pipelined bus with wrap-around, so that all packets return to their sender after traversing the ring. This architecture can be called a "multi-ring" or "multi-bus" architecture, where each workstation has access to a lightly loaded broadcast ring or bus. Each channel connects *16 workstations;* a transmission from any workstation will travel down the optical ring and reach all other workstations, implementing the transmission, broadcast or multi-cast function, and return to the sender. We envision that the primary source of delay over the network will be the "*time-of-flight*" of the optical transmissions over the fiber image guides. This delay will be nearly minimal, provided that the smart pixel arrays are carefully designed.

The design of an optical multi-ring using conservative 1999 technology is assumed[25]. We stress that the design described herein is conservative and based on modest 1999 technology, and that much more aggressive designs are technologically feasible with careful design. Assume that each SPA is specified to have 512 optical I/O bits clocked at 300 MHz, and 128 electrical data I/O pins clocked at 150 MHz. This specification results in an optical multi-ring with 153 Gb/s of aggregate optical bandwidth, and with a peak of 9.6 Gb/s of electrical I/O bandwidth per SPA. Each PCB supports one SPA and thus has the ability to inject and extract up to 9.6 Gb/s of bandwidth to and from the multi-ring. A system with 16 PCBs can generate up to $9.6 \times 16 = 153$ Gb/s of traffic. Designs for smart pixel arrays and optical multi-rings which have peak capacities in the 10s of Terabits per second are described in ref. 24.

The error and flow control protocols will determine how much of this peak optical bandwidth is actually "usable". Protocols with large processing delays at either end can dramatically lower the usable bandwidth to a small fraction of the peak bandwidth. The peak usable bandwidth can be increased to essentially equal the peak capacity by a variety of schemes described later in the paper, all of which require more logic or "intelligence" within the SPA.

We are targeting an optical clock rate of 300 MHz for each of 512 optical I/O bits. This clock rate implies a 3.33 nanosecond period between clock edges, which will have a large influence on the SPA logic design. In the future, the optical clock rates are expected to increase to the 1 - 10 GHz range, especially when CMOS/VCSEL technology becomes available. Our design techniques described in section 3 and 4 will readily scale to higher optical clock rates, and larger smart pixel arrays.

Assume the smart pixel array will be designed for a CMOS IC with dimensions 0.8×0.8 cm, using conservative 0.8 micron technology. Electronic bond pads will be placed around the perimeter of the die and will require about 150x150 microns each, yielding 200 I/O pads, 128 for data and 72 for control. Assuming 0.5 mm is reserved around the perimeter for I/O pads, guard rails, and power distribution, then the internal VLSI area for logic and optical I/O is 0.7×0.7 cm. Assume that 10 % of this internal area is reserved for routing of power and ground busses for the optical receivers and transmitters. Assuming a 50×50 micron standard cell for each optical receiver or optical transmitter, then 512 receivers and 512 modulators will require about 6 % of the remaining usable VLSI area. According to MOSIS, IC designers should aim for a peak density of 3,000 NAND gates per mm^2 using 0.8 micron CMOS technology. This figure translates to a density of 243 logic gates per pixel, on average. The IC designer should not exceed this gate density or else the VLSI layout may be unrealizable. In this paper, we propose "lean" error and flow protocols that easily fit within these technology constraints.

Figure 2a illustrates the smart pixel array and its main components. Each SPA has an associated external Message Processor (MP), which acts as the interface between the workstation electronics and the SPA. The SPA may have one electrical "injector" channel (64 bits wide) for injecting electrical data from the MP onto the optical multi-ring, and one electrical "extractor" channel (64 bits wide) for extracting optical data from the multi-ring and forwarding the data in electronic format to the MP. For increased fault tolerance and reconfiguration, this 64 bit wide channel could consist of two 32 bits channels operating in parallel, or in general multiple parallel channels[23]. Workstations are evolving to support multiple processors internally; the SPA can be designed to provide a dedicated optical broadcast bus for each processor within a multi-processor workstation. Each electrical channel also has several handshaking signals for coordinating data exchanges with the Message Processor (MP). The smart pixel can be organized as 8 optical channels each 64 bits wide, for a total of 512 optical I/O bits. Each optical channel has a peak bandwidth of 19.2 Gb/s, and we assume that 2 PCBs share each optical broadcast channel. Referring to Fig. 2a, there are 8 *Channel* modules, one for each optical channel. The SPA also includes a *Transmit Queue* for storing packets to be sent over the optical network, and a *Receive Queue* for storing packets to be sent to the workstation. The *Expander* module forwards packets from the *Transmit Queue* to the appropriate *Channel* module to be transmitted. The *Concentrator* module forwards packets from the optical channels to the *Receive Queue* .

To maximize the effectiveness of the flow control protocols, each SPA should have sufficient "on-chip" buffering for packets. A SPA may buffer up to 8 - 16 packets awaiting transmissions, and up to 8 - 16 packets that have been received from the optical network and are waiting to be transferred to the MP. For maximum efficiency, the packet buffers in the "*Transmit Queues*" and "*Receive Queues*" in figure 2a can be "shared" among the channels in the SPA. These large shared queues will have capacities of several Kbits up to several Mbits of high speed memory, and may occupy up to 50 % of the SPA real estate, depending upon the design. The VLSI resources required for these queues will decrease the resources available to implement error and flow control protocols, increasing the importance of effective protocols that utilize the buffer space.

Fig. 2b illustrates the overview of a single *Channel* module. The *Channel* module consists of a *Channel Control Unit*, a *Local Channel Transmit Buffer*, a *Local Channel Receive Buffer*, four *Pipeline Latches*, and 3 combinational *Logic Blocks* in between the pipeline latches. In general, the optical channels are clocked fast (300 MHz) so that any packet to be transmitted or received must be stored physically near the optical channel, in the local channel buffers rather than the *Transmit Queue*, to avoid the delays associated with long distance VLSI circuitry. A packet to be received is first copied into the *Local Channel Receive Buffer,* and then later it is moved to the shared *Receive Queue* through the *Concentrator* module.

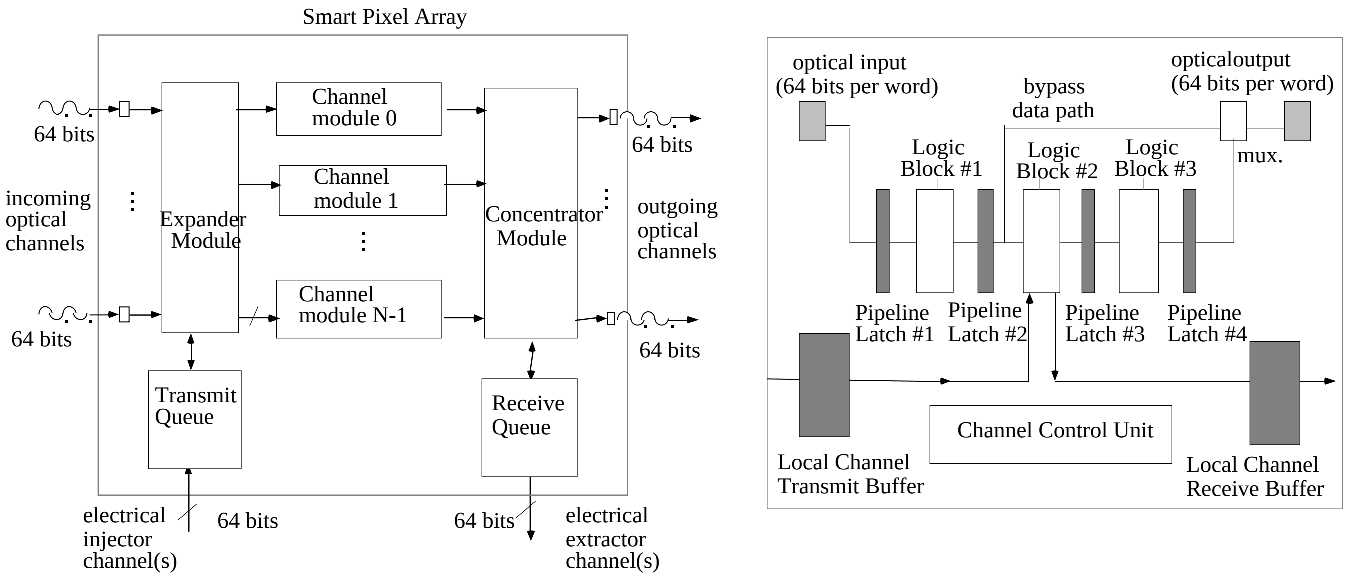


Fig. 2: (a) Overview of SPA with major modules and datapaths shown. (b) Overview of One Channel Module.

The *Concentrator* module in Fig. 2a is a complex digital circuit which can remove multiple packets from multiple optical channels, and move them in parallel into the shared *Receive Queue*. In an optical network, the concentrators must be exceptionally fast, which further complicates their design. The design of very fast and efficient concentrators for optical networks is described in ref. 20. The concentrator may occupy up to 10 - 20 % of the VLSI area of the SPA, depending upon the design.

In general, both packet switched and circuit switched modes are possible in the optical multi-ring network, with both fixed and variable length packets. In this paper we assume a fixed sized packet switched format, with two fixed size packets, short and long packets. All packets are transferred as multiple 64 bit words. The short packet has a fixed length of 32 bytes. Each short packet has 8 bytes for control (header and trailer), 16 bytes for payload, and 8 bytes for error control. This short packet length was chosen to support efficient reads and writes of multiprocessor memory over the network, and can be changed to suit the application.

The short packet header consists of several fields. The *Full-Empty* field (3 bits) identifies the packet as either full or empty. The *Destination* field (16 bits) identifies the packet's destination addresses. The *Acknowledge* field (16 bits) specifies which destination PCBs have acknowledged the packet. The *Source* field (4 bits) identifies the packet's source address. The *Sequence* field (4 bits) specifies the sequence number of the packet. Both source and sequence fields are used to support error and flow protocols which can guarantee that an arbitrarily large message can be partitioned into individual packets, sent over the optical network, and reassembled at the receiver and delivered properly. The *Payload* field (16 bytes) is suitable to carry a shared memory read or write, with a 64 bit shared memory address and a 64 floating point operand. The *Parity* fields (14 bytes) carry all the parity checksums for the packet. The *Error-Detected* field (3 bits) in the packet trailer identifies the packet as either erroneous or error-free. The reason for using 3 bits here will be described in section 4. This packet format is representative, and the system designer can vary the size of the fields and the packet.

The long packet format consists of 720 bytes, with a 16 byte header, a 516 bytes of payload, 8 bytes of trailer and 186 bytes of parity check. This size was chosen to support efficient transmission of 512 byte blocks of data. The long packet format also supports the transmission of variable size messages, up until the maximum length of 512 bytes of payload. To simplify the SPA design all the buffer spaces are designed in increments of the maximum long packet length. The size of 720 bytes is arbitrary and other sizes can be selected.

3. ERROR AND FLOW CONTROL PROTOCOLS

To date there is little consensus of what functions should be included in the smart pixel arrays of optical networks. Redmond and Schenfeld argue that optics should be completely "logic-less" and proposed a logic-less VCSEL-based optical interconnect in [2,18]. In their model all logic processing is done in external ASICs, *off-chip* from the optical transmitter ICs and optical receiver ICs. Proponents of *All-Optical* and *Passive-Optical-Network*s also implicitly argue for error control be to done in external ASICs at higher levels of the protocol hierarchy, away from the optical transmission layer, since these systems cannot perform complex logic operations on vast amounts of optical data. In contrast, we argue that error and flow control functions should be integrated right into the smart pixel array and very close to the optical I/O, rather than deferring error and flow control to external ASICs in higher layers in the protocol hierarchy. This approach will eliminate unnecessary delays incurred if the error and flow control is implemented at an external ASIC, since moving large amounts of data between IC packages over electronic I/O pins is inherently slow when compared to optics.

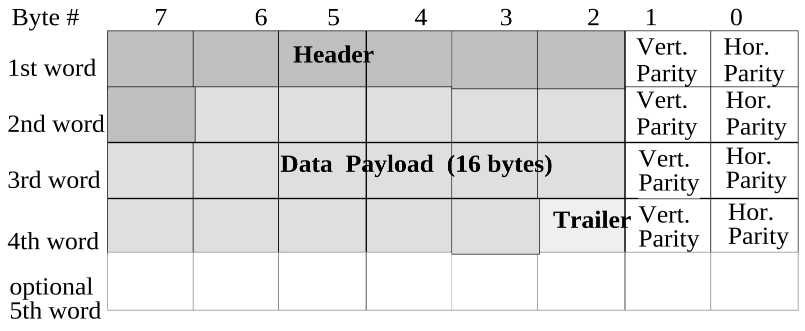


Fig. 3: Typical Short Packet Format - 32 bytes organized as 4 words of 64 bits each.

3.1. Traditional Cyclic Redundancy Check (CRC) Error Control

The traditional CRC error control code [4] is well suited for bit-serial optical datalinks. However, we argue that it is poorly suited for large scale 2D bit-parallel optical datalinks[25]. Using 0.8 micron technology, our VLSI analysis indicates that it is feasible to process at most 2 bits per clock tick, given the 300 MHz optical clock constraint. This bit-parallel CRC scheme still requires 128+16 clock ticks to process each packet, introducing a significant delay. These delays render the traditional CRC error control code infeasible for high speed bit-parallel 2D optical data links.

3.2. Generalized Multidimensional Parity Checks

In this section, we consider a multi-dimensional parity scheme which is relatively easy to implement in hardware. Our optical multi-ring architecture can support a great deal of bandwidth, and it is reasonable to trade some optical bandwidth for error control code simplicity. Simpler codes will require more redundant bits and may have a lower coding efficiency than traditional codes, but will also require much less VLSI processing. Since we have ample bandwidth, the tradeoff of optical bandwidth for hardware simplicity is worthwhile. Hence, we consider a simple hardware-based 2D parity check to detecterrors on the optical multi-ring. While multi-dimensonal parity checks are known, they are not often used in practice, perhaps due to their lower coding efficiency. In our environment, assume the primary cause of bit errors is thermal noise, and a low Signal-to-Noise ratio (SNR) at the receivers. Therefore, bit errors are independent events.

There are several potential sources of bit errors in an optical interconnect, although experimental data is currently unavailable. Poor alignment between fibers and photodetectors can degrade optical signals, although alignment techniques can compensate. Lack of power homogeneity over large 2D arrays of optical bits can cause a low SNR. Optical crosstalk between 2D arrays of optical bits can potentially cause bit errors. We feel that thermal noise will be a dominant cause of bit errors in environments with a low SNR or limited optical power.

In our proposed 2D error control scheme, each 256-bit packet is divided into $b=4$ independent blocks, where each block is arranged in a $x \times y$ array of payload bits, with a conventional 2D parity check. Thus, every row and every column in a block contains an additional parity bit. We will call this a "*block-structured 2D parity scheme*". This scheme yields a packet with $b(x \times y)$ payload bits and $b(x+y)$ parity bits. Referring to the short packet format in Fig. 5, each 64 bit word forms 1 block, and each block can be encoded as an 8×6 array of payload bits, with 8 vertical parity bits and 6 horizontal parity bits. (The remaining 2 bits are used to compute parity of the checksums, which is necessary to detect certain bit error patterns.)

In the 2D parity scheme, all odd number of bits errors are detectable, and all error patterns with fewer than 4 bit errors are detectable. *For an error to be undetected in a block, an even number of 4 or more bit errors must occur, in a specific pattern where all rows and columns with errors only have an even number of errors.* In a 2D parity scheme, the only undetected bit errors occur in specific patterns with 4, 6, 8, ... bit errors. It can be verified that the probability of 4 undetected bit errors occurring in a packet is given by [25]

$$\log_{10}[\Pr(\text{undetected error })] \quad 3.4906 + 4\log_{10} p \tag{1}$$

where p is the "raw" BER, and where x=8, y=6, and b=4. There also exist patterns of 6 undetectable bit errors[25]. It can be verified that the probability that 6 undetected bit errors occur in a block is given by the following (for x=8, y=6) [25];

$$\log_{10}[\Pr(\text{undetected error })] \quad 4.2465 + 6\log_{10} p \tag{2}$$

Observe the decrease in probability for the error patterns with 4 and 6 undetected errors, in equations (1) and (2). It can be verified that higher order patterns with 7, 8, or more bit errors will have lower probabilities and negligible effect when the raw BER is 10^{-3} or lower and can be ignored. Therefore, based on equations (1) and (2), for most analyses only the pattern with 4 bit errors need be considered.

In our application, when a packet with a detectable bit error pattern is discovered, the destination smart pixel array has two choices. It can perform forward error correction (FEC) on the packet, or it can request retransmission of a new packet. Due to the physical and temporal proximity of the sender and receiver in our optical multi-ring, it is easier to simply request retransmission, which we assume. However, FEC will also work very well.

Example # 1: Given a raw BER of p=10^{-4}, with x=8, y=6, b=4, according to Eq. (1) the probability of undetected errors occurring in the short packet shown in Fig. 3 after the error control is 10^{-12} (i.e. 10 exp (3.5 - 16)). This corresponds to an BER after error control of 10^{-14}, i.e. 4 bit errors out of 256 bits occurring with prob. 10^{-12}. (This BER is a first order estimate, since there are statistical dependencies between the bit errors in a packet. Hereafter, we report the "packet error rates" rather than the BERs.) This error rate is low, and may be sufficient for most present day applications. However, for high performance computing systems moving 10 - 100 Terabits of data per second, the BERs will have to be much lower. The next section describes stronger codes.

3.3. 3 Dimensional Error Control

For higher levels of error control, a full 3 dimensional parity check can be used, where the bits are arranged in a 3D $x \times y \times z$ array of payload bits. Each x$\times$y plane has x+y additional parity bits, with a parity bit for every row and column. In addition, one additional x$\times$y plane contains the parity bits computed along dimension z. Note that we no longer use the parameter "*b*" in this 3D scheme.

In the 3D parity scheme, all error patterns with fewer than 8 bit errors are detectable. *For an error to be undetectable, there must be at least 8 bit errors occurring at the 8 corners of a 3D cube.* There are higher order patterns with 12 bit errors which are also undetectable, but these have much lower probabilities and can be ignored. Hence, the probability of an undetected error pattern can be approximated. For short packets with x=8, y=6, and z=4, the following holds [25];

$$\log_{10}[\Pr(\text{undetected error })] \quad 3.8785 + 8\log_{10} p \tag{3}$$

and for long packets with x=8, y=6, and z=90, the following holds [25];

$$\log_{10}[\Pr(\text{undetected error })] \quad 6.49 + 8\log_{10} p \tag{4}$$

Example # 2: The 3D parity check can yield exceptionally low undetected BERs. Given a raw BER p=10^{-4}, with x=8, y=6, z=4, by Eq. (3) the probability of undetected errors occurring in a short packet after error control is approx. 10^{-28} (10 exp (3.88 - 32)). In other words, for most applications the BER will result in no observable errors and there will be no need for additional error control in external ASICs.

Example # 3: Given a raw BER p= 10^{-4}, with x=8, y=6, z=90, according to Eq. (4) the probability of undetected errors occurring in a long packet after error control is approx. 10^{-25} (10 exp (6.5 - 32)). In other words, for most applications the BER will result in no observable errors and there will be no need for additional error control in external ASICs. The BER after error control can be adjusted to any value by the designer, through the selection of the design parameters p, x, y, and z. Also, higher dimensional parity codes can be used when stronger protection is desired.

3.4. Forward Error Correction

The multi-dimensional parity checks also supports moderate Forward-Error-Correction. Single bit errors are the most likely cases in the optical network, and these can be detected and corrected without requiring the sender to retransmit the packet. However, in our current optical network design we request a packet retransmission from the sender, since it is easier to request retransmission than to perform error correction. Nevertheless, we will briefly describe some error correcting capabilities.

In a 2D parity check, all single bit errors are detectable and correctable. A single bit error is detectable by observing parity errors in exactly one row and one column. The bit error must occur at the intersection of row and column with parity errors, and can be easily corrected by inverting this bit. This error correction scheme only requires a single EXOR gate per pixel. Patterns of two detectable but uncorrectable bit errors in a 2D parity check exist; there are 2 bits errors occurring at the opposite corners of a square. The parity check detects this situation by observing parity errors in exactly 2 rows and 2 columns. However, the 2D check cannot correct this situation, since it does not know which two corners of the square have the errors. This bit error pattern can be corrected by moving to a 3D error control scheme. In a 3D parity check, all single and double bit errors are detectable and correctable. All triple bit errors are detectable. Patterns of 4 detectable but uncorrectable errors exist. Improved correction can be obtained by moving to higher dimensions.

4. FLOW CONTROL

Our smart pixel arrays supports two basic flow control protocols, which we call "*persistent send*" and "*one-shot send*". Other more complex schemes exist, although they will require more complex digital logic within the SPAs.

4.1. One Shot Protocol

The *One-Shot* protocol supports the basic sending and receiving of packets with acknowledgment. This scheme can implement a traditional *Stop-and-Wait* protocol, where the sender of a packet waits for an explicit acknowledgment from the receiver for each and every packet transmitted[4]. In the *Stop-and-Wait* protocol, if an acknowledgment does not come within a specified Time-Out period, the packet is re-transmitted automatically by the sending smart pixel array. This stop-and-wait protocol is guaranteed to deliver packets in order and without repetition, and is simple to implement in hardware. The sending and receiving protocols will be summarized.

Sender Protocol: The sending MP loads the packet to be sent into the *Transmit Buffer* Module of the SPA, by writing 64 bits at a time into the electrical "injector" channel using a conventional handshaking protocol. The parity bits for the packet are computed by the *Transmit Queue* module in the SPA. The packet is then sent through the *Expander* Module to the appropriate *Channel* module, where it will be loaded into the *Local Channel Transmit* Buffer. Once the packet is ready in the local buffer, the channel controller waits for the next idle packet slot. Slots go by every few nanoseconds, and hence the logic within the SPA must be extremely fast.

Upon detecting an idle slot, the packet is injected into the slot, the slot is marked full, and the parity bits for the packet will be set appropriately. The channel module then waits for the packet to return (by waiting for the same slot to return), at which point the packet will be removed from the slot and the slot will be marked idle. The sender SPA will notify the MP of the status of the packet, by asserting a Success or an Error signal. The Success signal informs the sending MP that the transmission was successful. The Error signal informs the MP that the transmission was erroneous. In this case, the SPA is designed to automatically retransmit the packet it has stored in its *Local Transmit Buffer.* In this model, the MP is relieved of all error-detection functions.

Receiver Protocol: Channel modules process all packets passing through their ring buffers, looking for packets addressed to themselves. If they detect a packet addressed for them, it is first captured in a *Local Channel Receive Buffer*. The packet is then transmitted through the *Concentrator* module to the *Receive Queue*, and then forwarded to the MP using a conventional handshaking protocol.

Protocols at Intermediate SPAs: All intermediate SPAs can perform error detection on packets as they pass by. Upon detecting an erroneous packet in its Ring buffer, a Channel module will set an Error Detected bit in the packet trailer. This feature can also be disabled, so that only the receiving SPA performs error detection.

The basic stop-and-wait protocol does not support pipelining, where multiple packets can be transmitted by a sender before an acknowledgment is received. Pipelining allows a SPA to have multiple outstanding packet transmissions, and can improve performance significantly. Such scehemes include "Sliding Window" and "Automatic Repeat Request" protocols[4]. Pipelining can be included by replicating the number of local transmit buffers available in each optical channel module. Without pipelining, each SPA can transmit one short packet (32 bytes) every 4 clock ticks, and then must wait for 64 clock ticks for the acknowledgment to be received (16 SPAs times 4 clock ticks latency per SPA). Since 2 PCBs share a single optical channel in our sample system, the peak throughput per optical channel is 64 bytes every 68 clock cycles, or approx. 2.26 Gb/s, which is much less than the peak bandwidth of 19.2 Gb/s. About 50 % of the short packet represents "payload", so the peak usable error-free bandwidth is actually about 1.13 Gb/s per optical channel, without pipelining.

By allowing pipelining with up to 16 outstanding transmissions per SPA, the throughput increases significantly. Each SPA can transmit up to 16 consecutive packets, requiring 64 clock ticks, after which point the first packet has been acknowledged and a new transmission (or retransmission in the case of an error) can proceed without delay. Hence, pipelining increases the usable bandwidth to essentially 100 % of the optical channel capacity of 19.2 Gb/s. In all networks, electrical or optical, each short packet has a certain control overhead. Since approx. 50 % of our short packets represents payload, the peak error-free data rate per optical channel is about 50 % of the peak channel capacity of 19.2 Gb/s. These figures can be adjusted by the system designer, by selecting the packet size, the payload and control sizes, and various other parameters.

For very large messages, i.e., 16 Kbytes in length, the Stop-and-Wait protocol can be inefficient if short packets are used, since a large message must be fragmented into several short packets, each carrying a 16 byte payload, each of which is transmitted one at a time. To transmit large amounts of data efficiently, we propose the use of a long packet format within the optical network, as described earlier in the paper. It is useful to include both types of channels in the smart pixel arrays, i.e., 4 channels can be designed to support short packets only, and 4 channels can be designed to support long packets only. The channels supporting long packets should use a "token ring" media access (MAC) protocol, rather then a slotted ring protocol, to minimize latency. The long packet format supports the transmission of variable size messages, up until the maximum length of 720 bytes. To transmit a 16 Kbyte message, the message can be fragmented into 32 long packets, each with a 512 byte payload, which are transmitted using a simple Stop-and-Wait protocol. This scheme has very good performance. Without pipelining, and with 2 PCBs sharing a single optical broadcast channel, the usable bandwidth is essentially 100 % of the channel capacity of 19.2 Gb/s.

The long packet length can be chosen to transmit a cache line or virtual memory page in one transmission, eliminating the need to fragment the message into smaller packets. Currently, cache lines vary in size from 256 to 512 bits, while virtual memory page sizes vary from 4 kB to 16 kB. The length of a long packet can be chosen to optimize system performance.

4.2. "Persistent Send" Protocol

In the persistent-send protocol, a packet that is not received due to a lack of buffer space at the receiver is left on the ring by the sender, recirculating until it returns with all acknowledge (ACK) bits set. If the packet returns with an "Error detected" bit set, then the sender will re-transmit a clean version of the packet to all destinations. Destinations can detect duplicates by observing the sequence number of the packet. This variation of the Stop-and-wait protocol requires insignificant hardware, and can result in faster transmissions.

4.3. Dedicated "Flow Control Channels"

The smart pixel array can be extended to support a dedicated 64 bit *Control Channel* which broadcasts 4 status bits for each of the 16 workstations on the network, i.e., each SPA could broadcast its own "Buffer Full" status bit permanently over the optical network; If this bit is set then the SPA cannot receive any more packets. All other SPAs can observe the buffer full status bits of any destination workstation (since they are permanently broadcast) and suspend their transmissions to that destination when appropriate. Each workstation may also want to broadcast other status bits[23].

4.4. Livelock, Deadlock and Best 2-out-of-3 Majority Voting

Deadlocks are prevented in each ring since packets are always moving forward. Their forward movement in the ring is never suspended, and therefore we do not require any deadlock detection or resolution logic as required in other systems[17]. Livelock can occur when an empty packet undergoes a bit error and becomes full, and remains circulating on the ring for ever. If all packets were corrupted in this manner, the ring would be permanently full of erroneous packets, thus disabling all workstations from accessing the optical network. To reduce the probability of empty packets being marked full (and vice versa), the Full/Empty and Error Detected fields in each packet can be encoded using 3 bits each. A "1" for these fields is denoted by setting three "1"s in the header, and similarly for logic "0". To determine if a packet is full or empty, a receiver processes these three bits and performs a best 2-out-of-3 majority vote. An incorrect decision will be made only if 2 or 3 of these critical bits all undergo the same error. A single bit error will not change the logic value, due to the majority vote circuit, thereby significantly reducing the probability of erroneous packets circulating forever. To further avoid livelock, each SPA could process packets as they pass by. If it detects a full packet whose source field matches its own source address, the packet can be removed and the slot will be marked empty.

When the source ID bits of a full short packet are corrupted by errors, the sender will not recognize its address in the source ID field and will not clear the packet, which may remain on the ring causing livelock. There are schemes to eliminate livelock even in the presence of bit errors. In our system, the first word of the packet must return to the sender after 64 clock ticks, since every SPA has a fixed latency of 4 clock ticks. Hence, the sending SPA can be designed to count clock ticks and always clear a returning packet. Long packet channels do not have this problem, since they can rely on a token ring access protocol. Alternatively, one SPA can be designed to act as a ring master, which flags every full packet passing by. The flags are reset by the senders as the packets are marked empty. No flagged full packet should ever arrive at the ring master, so any such packet can be reset by the ring master. The presence of bit errors complicates the protocols, but similar schemes are used in bit parallel electrical rings.

5. VLSI DESIGN

We have implemented representative subsets of these smart pixel arrays in 0.8 and 0.5 micron CMOS/SEED optoelectronic technologies[25]. We followed an advanced standard cell *Design Methodology*, using the Synopsis, Cadence, and Spice CAD tools. The SPA logic was first specified in the VHDL hardware description language. The Synopsis[21] CAD tools were then used to perform detailed functional and timing verification of the design. (The VHDL design was annoted with the accurate figures for logic gate delays, optical I/O delays for the timing verification).

Once the VHDL design was validated, the Synopsys synthesis engine was used to transform and optimize the digital logic functions and to generate a VLSI netlist optimized for time, given the 0.8 micron standard cell technology. (Timing constraints are more critical than area constraints, so the goal of the optimization was minimum time). In all cases, the target technology was the CMC K-cell library, a 0.8 μm CMOS standard cell library[5].

Synopsys synthesizes a digital logic design by determining an optimal or near optimal implementation of the logic functions, from the VHDL logic description supplied to it, using the given standard cell library. Synopsys generates an VLSI netlist of the optimized digital circuit, and reports estimates of the VLSI area and VLSI delay of the optimized circuit. Synopsys deals strictly with logic transformations to achieve an optimal or near optimal realization of the specified digital circuit using a standard cell library. The netlist generated by Synopsys was then fed to an automatic VLSI place-and-route tool, the Cadence CAD tool, to complete the detailed VLSI layout. The automatically generated VLSI layout was then fed to the SPICE CAD tool to perform detailed analog signal simulations, which considers voltages, drive currents, loading, and parasitic capacitance and resistance of metal interconnects. The analog simulations from SPICE were compared with the digital simulations from Synopsis, to ensure that the VLSI mask was functioning correctly. Table 2 illustrates typical delays for various logic gates in the 0.8 micron standard cell library. Table 2 also illustrates projections of the logic gate delays for smaller and faster technologies. All delays are "nominal" or expected case, rather than worst case.

The combinational logic designs illustrated in figure 4 represent our logic description before optimization by Synposis. Synopsis transforms the input logic description to an optimized design which cannot be improved. For example, our logic design in figure 7 uses several 2 input EX-OR gates. Synopsis transforms the input design and optimizes it for the maximum clock rate. Synopsis will perform transformations such as De Morgan's theorem, and will consider replacing 2 input EXOR gates with other standard cells such as multi-input logic gates. Synopsis will invert logic signals if they are faster than non-inverted signals in the standard library. In short, the optimized designs produced by Synopsis could not be improved, but they are largely unrecognizable due to the optimization process. Hence, we present our designs before the optimization process.

The proposed smart pixel specifications are ambitious compared to conventional electronic router designs, which deal with much slower clock rates. A 300 MHz optical clock yields a 3.33 nsec period, which leaves very little time for logical processing. From Table 2, the nominal propagation delay through a DFF is 0.7 nsec, i.e., after the clock rising edge 0.7 nsec of the clock period is used before the output of the flip flop is even valid. From Table 2, the nominal setup time for a DFF is 0.46 nsec, i.e., the data must be settled for at least 0.46 nsec at the end of each clock period before it can be latched at the next rising clock edge. Therefore, the time available per clock period for combinational logic is 3.33 nsec - 0.7 nsec - 0.46 nsec = 2.17 nsec. Therefore, all combinational logic in each pipeline stage must complete within 2.17 nanoseconds, a challenging design constraint.

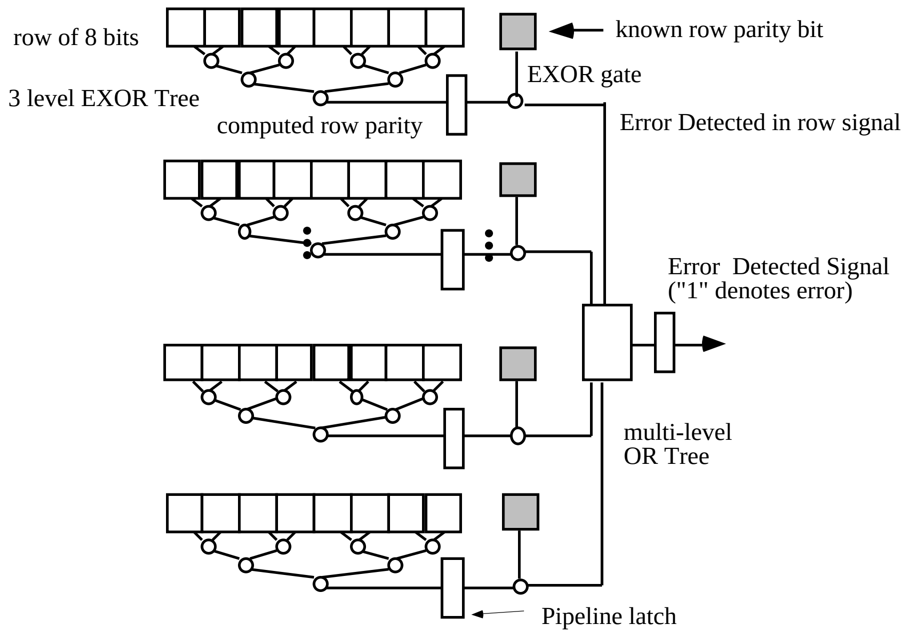


Fig. 4: 2D Parity Computation circuitry.

To meet the 300 MHz optical clock specification, the combinational digital logic was assigned to pipeline states manually by the human designer (since CAD tools are not this advanced yet), such that each stage does not exceed the 2.17 nanosecond critical path. Referring to Table 2, for 0.8 micron technology the 300 MHz optical clock rate limits the combinational logic to roughly 3 EXOR gates, or 6 regular logic gates per stage.

To meet the critical path constraint (2.17 nanoseconds), it was necessary to distribute the error detection logic over several pipeline stages, as shown in Fig. 4. The nominal delay of an EXOR gate is 0.56 nanosec. A parity check on 8 bits requires a 3-level EXOR tree which requires 1.7 nanosec. (A 4-level EXOR tree would not settle within one clock period.) Recall from section 2 that the packet is transmitted as four 64 bit words. In Fig. 4, pipeline stage 1 computes the 8 horizontal and 6 vertical parities of the 6×8 array of bits in a word in parallel. Pipeline stage 2 computes whether each row or column has a parity error, by comparing the computed parity with the a priori known value of the row/column parity with another EXOR gate. Pipeline stage 2 also starts to compute the overall Error/No-Error status of each word of the packet, by ORing together the 8 horizontal parity status signals and 6 vertical parity status signals. This OR tree requires 4 levels of binary OR gates. The first 3 levels of the OR tree can be computed in pipeline stage 2, whereas the last level must be computed in stage 3.

If an error in the block is detected in stage 3, the appropriate Error-Detected bit must be set in the trailer byte of the packet. The value of this Error Detected bit is precomputed in pipeline stage 3. This value is written into the packet trailer by pipeline stage 4, before the packet leaves the SPA. (The *Error Detected* bit is in the packet trailer since the header has already left the smart pixel array by the time an error is detected.)

In this paper, we describe a design where the error detection is performed on every packet as it passes through a SPA. Hence, every packet must pass through the 4 stage pipeline in every SPA. However, it is possible to perform the error detection process after a packet has been received by the SPA and removed from the channel module; packets that pass through a SPA without being received do not need to undergo error detection. This scheme can lower the latency per SPA to 2 pipeline stages rather than 4, and this is indicated in the bypass datapath in Fig. 4. In our current scheme, the error detected bit is set in the packet trailer once an error is detected, and the sending SPA detects this error as soon as the packet returns to the sender.

Address decoding is handled in a similar manner. Pipeline stage 1 compares the packet's destination field with the SPA's own address. To minimize the complexity of address comparison, and yet still maintain a flexible addressing scheme, we assume addresses are encoded using a *1-hot* code. This code requires 16 bits to encode the destinations given 16 PCBs in the backplane. If a workstation *i* is a destination, then there is a 1 in the *i*-th bit of the destination field. This scheme also supports multicasting, since any and all subsets of all 16 PCBs can be specified in the destination field. The one-hot address decoding requires 16 AND gates (to AND the destination field with the unique 16 bit PCB address supplied by the MP), followed by an OR tree which ORs all the results together. The OR tree is 4 gates deep, and the address decoding can be computed in pipeline stage 1. (To conserve bits, more complex codes can be used. For example, a 5-chose-2 code can be used to identify 16 PCBs, although it is slightly more complex to decode, requiring several logic gates).

If the addresses match, the packet should be copied into the *Local Channel Receive Buffer*. The packet is copied from the data from Pipeline Latch #2. (The *Local Channel Receive and Transmit Buffers* need buffer only a few 64 bit words in the case of the long packet format.) Pipeline stage 1 will also determine if the slot is empty and if there is a packet to transmit in the *Local Channel Transmit Buffer*. To determine if a slot is empty we perform a 2-out-of-3 majority vote on the Full/Empty field, as described in section 3, which can be achieved accomplished in pipeline stage 1. If the slot is empty, we can inject the packet into the slot in *Logic Block* #3.

Logic Block #1 will also determine if the slot contains a packet previously sent by this SPA. Such a packet is returning to its sender and should be removed from the slot, so that its header can be examined. The *Channel Control Unit* can find returning packets by counting slots or clock cycles from the time it injected a packet (assuming the ring is formatted to include an integer number of packet slots). If the packet is from this SPA, it can be removed in Pipeline Latch #2.

5.1 Automated VLSI Layout using Standard Cells

Figure 5a illustrates the VLSI layout of individual pixel generated automatically by the Cadence CAD tool, using 0.8 micron CMOS. The Synopsis CAD tool indicated that a clock rate of 300 MHz was achievable through the digital logic. We also used the SPICE CAD tool to verify the timing constraints of the VLSI layout generated by Cadence. SPICE verified that all critical paths meet the design objectives, consistent with the analysis by Synopsis. Hence, the SPAs should meet the 300 MHz optical clock objective.

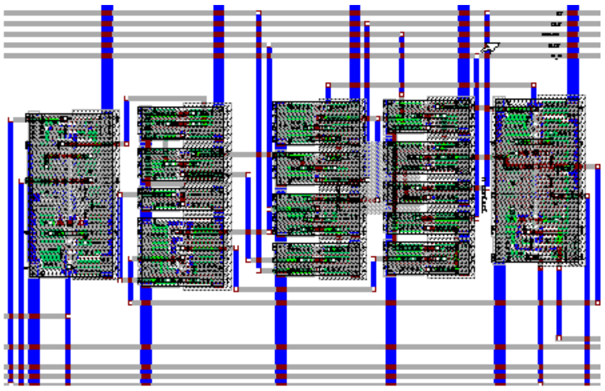
The VLSI area required for the error control logic consists of the horizontal and vertical parity trees in each channel. The error control logic includes the parity generators for 6 rows and 8 columns of each 64 bit block. Each parity generator includes 8 EXOR gates in a tree and 2 DFFs (which includes the logic to compute the parity and compare it to the transmitted parity). Each EXOR gate is equivalent in area to approx. 4 NAND gates, and each DFF is equivalent in area to approx. 6 NAND gates. Hence, the error control logic per channel consumes the equivalent of $14 \times (8 \times 4 + 2 \times 6) = 616$ binary NAND gates. When amortized over 64 pixels, the error control logic represents an overhead of approx. 10 gates per pixel, which is very modest and easily realizable with the 0.8 micron technology, as determined in section 2.

In summary, the proposed smart pixel array meets all the objectives, i.e., it supports an integrated error and flow control protocol and can operate at 300 MHz clock rates using 0.8 micron technology. It also performs error control on 150 Gbps of optical data with very low VLSI area. For comparison, recall that the FEC design in ref. 14 requires 1 cm^2 of silicon to perform error control on a 77 Gbps optical data link.

Representative subsets of these smart pixels were implemented in the 1997 Lucent/ARPA/COOP workshop on CMOS/SEED technology, using 0.5 micron technology. The optoelectronic devices were delivered by Lucent Technologies in the summer of 1998. Figure 5b indicates a CMOS/SEED device which contains our smart pixel arrays. Electronic testing has been completed, and the smart pixels function correctly. Additional testing is currently underway, and a complete description of the logic and testing will be forthcoming.

6. SCALABILITY PROJECTIONS FOR THE NEXT DECADE

In this section, we examine the scalability of a CMOS/VCSEL optical multi-ring network. Table 1 illustrates the Semiconductor Industry Association's (SIA) projections[19] for CMOS technology over the next decade, and projections on the density of optical I/O using the CMOS/SEED technology from ref. 11. To date there are no published projections on the density of optical I/O using CMOS/VCSEL technology. However, if the process for fabricating large arrays of low power VCSELs matures, and the process for flip-chip bonding of VCSELs to CMOS matures, then in the future the CMOS/VCSEL devices may have a similar optical I/O density to the SEED technology, with considerably higher optical clock rates. Hence, in the future SPAs with thousands of high bandwidth VCSELs seem plausible.



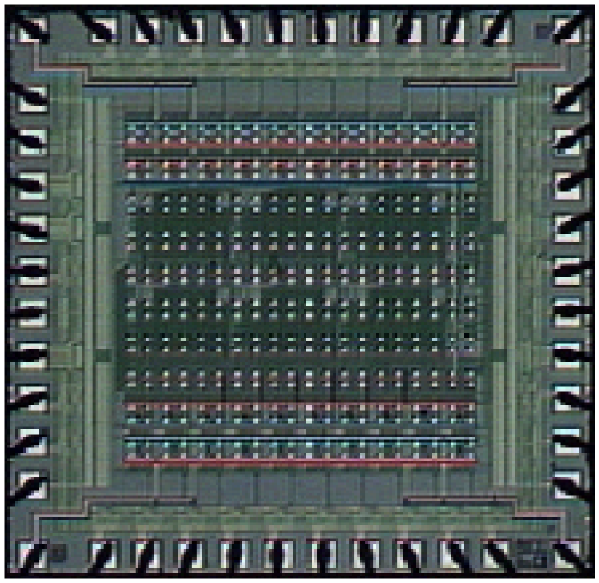


Fig. 5: (a) Automated VLSI Layout of an individual pixel. (b) CMOS/SEED optoelectronic IC implementing smart pixel arrays in 0.5 micron CMOS, fabricated by the 1997 Lucent/ARPA/COOP workshop.

Consider the implications of error and flow control using a CMOS/VCSEL technology in year 2004. Assume 0.12 micron CMOS technology, and a target optical clock rate in the GHz range. Following the methodology of section 4, the data in Table 2 indicates that each pipeline stage will be sufficient to support several logic gate delays, sufficient for 2 or 3 dimensional error control. Hence, we conclude that our pipelined smart pixel design will support GHz optical clock rates using 0.12 micron technology. In summary, the proposed error and flow control schemes scale well. The schemes are simple enough to implement in SPAs potentially supporting thousands of optical I/O and optical clock rates in the GHz range.

Clocking an optical network at GHz rates will require careful design. In existing bit-parallel electrical rings, the clocks on all nodes are synchronized and the clock is usually transmitted along with the data. There is usually a small "programmable delay and buffer" somewhere in the ring which is needed to close the ring. This scheme can be used in bit-parallel optical rings, although the design tolerances will be much lower.

7. CONCLUSIONS

An Intelligent Optical Multi-Ring Network which interconnects multiple workstations through fiber image guides over distances of 10s of meters has been described. Our analysis indicates that without effective error and flow control protocols, the true potential of such an optical interconnect may be difficult to exploit. A class of effective error and flow control protocols for such bit-parallel optical interconnects has been described. Smart pixel arrays supporting these advanced error and flow control functions at optical clock rates of 300 MHz have been fabricated, using 0.5 μm CMOS technology. Our proposed error and flow protocols scale well with improvements in technology. With faster CMOS/VCSEL technologies, optical clock rates in the GHz range should be feasible by exploiting the principle of pipelined smart pixels.

Year	Feature Size (microns) [19]	Gates [19]	Area (Sq. mm) [19]	Max # Optical I/O [11]	Optical Clock [11]	Max. Optical I/O BW
1998	0.25	2 M	600	12,000	350 MHz	2.1 Tb/s
2001	0.18	5 M	800	24,000	500 MHz	6 Tb/s
2004	0.12	10 M	1,000	40,000	700 MHz	14 Tb/s
2007	0.10	20 M	1,250	50,000	1 GHz	25 Tb/s

Table 1: Projections for CMOS/SEED Technology, from ref. 11 and 19.

Feature Size (microns)	DFF setup	DFF prop	OR gate	EXOR gate
0.8	0.462	0.674	0.564	0.723
0.5	0.290	0.421	0.353	0.452
0.12	0.069	0.101	0.084	0.109

Table 2: Projections of Gate Delays over next decade, based on ref. 19.

This paper verifies that multi-dimensional parity checks can yield a powerful error control system and can be simple to implement in 2 dimensional bit-parallel optical systems. This paper also argues that it is possible to achieve an Intelligent Optical Network which is as error-free as conventional main memory, by embedding error and flow control directly in the physical layer of the network, i.e., in the smart pixel array. In this system, all bandwidth passing through the network can be viewed as "error-free", eliminating the need for further error control in software or external hardware integrated circuits. This could represent a significant advantage over other optical networks, such as *All-Optical* and *Passive-Optical* networks, which cannot perform extensive processing such as error control directly in the "optical layer".

Finally, perhaps one of the most important contributions is highlighting the potential importance of error control in optical interconnects for high performance computing systems. Our analysis indicates that BERs will have to be far lower than traditionally believed for high performance computing systems. In our view, traditionally "safe" BERs such as 10^{-15} will likely be inadequate for the high performance (multi TeraFlop) computing systems of the future, where BERs of 10^{-30} may be necessary. Given the ever increasing power of computing systems, the need for increasingly lower BERs and effective error control protocols will likely continue.

8. ACKNOWLEDGMENTS

This research was supported by NSERC Canada Grant OPGO-0121601 and in part by the Canadian Institute for Telecommunications Research (CITR) of the Networks of Centers of Excellence (NCE) Program of the Government of Canada. The use of CAD software and silicon CMOS VLSI foundry services from the Canadian Microelectronics Corporation (CMC) are acknowledged. Thanks to the Lucent Optoelectronic Technologies Group for fabricated the SPA in Fig. 5. Thanks to former graduate students Victor Tyan (now at Lucent Technologies), and Martin Saint Laurent (now at Intel), for their assistance in the design of the SPA. Thanks to Albert Au and Boonchuay Supmonchai of the MACS Laboratory at McGill University for their assistance with testing the SPA. Our collaborators in the CITR major project including Prof. David Plant and Prof. Andrew Kirk in the Photonic Systems Laboratory at McGill University, and Prof. Zvonko Vranesic of the University of Toronto.

9. REFERENCES

1. F. Argenti, G. Benelli, A. Garzelli, “Generalized Stop-and-Wait Protocol”, *Electronic Letters*, Vol. 28, No. 9, pp. 861-863, April 1992.
2. S. Araki, M. Kajita, K. Kasahara, K. Kubota, Kurihara, I. Redmond, E. Schenfeld, T. Suzaki, "Experimental free-space optical network for massively parallel computers", Applied Optics, 35, pp. 1269-1281, 1996.
3. R. E. Blahut, *Theory and Practive of Error Control Codes*, Addison-Wesley, MA., 1983.
4. D. Bertsekas, R. Gallager, *Data Networks*, 2nd Ed., Prentice Hall, 1992.
5. Canadian Microelectronics Corporation (CMC), *BiCMOS Design Kit V2.0 for Synopsys*, CMC, Carruthers Hall, Queen's University, Kingston, Ontario, K7L 3N6, 1996.
6. Canadian Institute for Telecommunications Research (CITR), Photonic Systems Major Project Overview, Annual Reports 1993 - 1998, McGill University (http://www.citr.ee.mcgill.ca).
7. D. Chiarulli, S. Levitan, R. Melhem, J. Teza, and G. Gravenstreter, "Partitioned Optical Passive Star (POPS) Topologies Multiprocessor Interconnection Networks with Distributed Control", *IEEE Journal of Lighwave Technology*, vol. 14, no. 7, pp. 1601-1612, 1996.
8. I. Glesk, K. I. Kang, and P. R. Prucnal, "Demonstration of ultrafast all-optical packet routing", *Electronics Letters*, vol. 33, no. 9, pp. 794-795, Apr. 1997.
9. K. Hamanaka, "Optical Bus Interconnection using Selfoc Lenses", Optics Letters, Vol. 16, No. 16, pp. 1222-1224, 1991.
10. A.G. Kirk, D.F. Brosseau, F.K. Lacroix, E. Bernier, M.H. Ayliffe, B. Robertson, F.A.P. Tooley, D.V. Plant, "Design and Implementation of a two-stage Optical Power Supply spot array generator for a modulator-based free-space interconnect", Optics in Computing OC'98, Brugges, Belgium, pp. 48-51, 1998.

11. A.V. Krishnamoorthy, and D. A. B. Miller, "Scaling Optoelectronic-VLSI Circuits into the 21st Century: A Technology Roadmap", *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 2, no. 1, pp. 55-76, Apr. 1996.
12. F. Lacroix, B. Robertson, M.H. Ayliffe, E. Bernier, F.A.P. Tooley, M. Chateauneuf, D.V. Plant, A.G. Kirk, "Design and Implementation of a Four-Stage Clustered Free-Space Optical Interconnect", Optics in Computing, OC'98, pp. 107-110, 1998.
13. F.B. McCormick, I. Cokgor, et. al., "2 Photon Optical Data Storage", Optics in Computing, OC'98, pp. 574-577, 1998.
14. M.A. Neifeld, and S.K. Sridharan, "Parallel Error Correction using Spectral Reed-Solomon Code," *Journal of Optical Communications*, 17, pp. 525-531, 1997.
15. M.A. Neifeld and R.K. Kostuk, "Error Correction for free-space optical interconnects: space-time resource optimization", Applied Optics, Vol. 37, No. 2, pp. 296-307, Jan. 1998.
16. J.C. Palais, "Fiber Optic Communications", Prentice Hall, 1984.
17. T.M. Pinkston, M. Raksapatcharawong, Y. Choi, "WARPII: An Optoelectronic Fully Adaptive Network Router Chip", Optics in Computing, OC'98, pp. 311-315, 1998.
18. I. Redmond and E. Schenfeld, "A Distributed Reconfigurable Free-Space Optical Interconnection Network for Massively Parallel Processing Architectures", *Proc. Int. Conf. Optical Computing*, Edinburgh, 1994. Institute of Physics Publishing, 215-218, 1995.
19. Semiconductor Industry Association, "The National Technology Roadmap for Semiconductors", SIA, San Jose, 1994.
20. B. Supmonchai, and T. H. Szymanski, "High Speed VLSI Concentrators for Terabit Intelligent Optical Backplanes", *Optics in Computing 1998*, Brugges, Belgium, pp.306-310, 1998. (Ph.D. thesis in preparation.)
21. Synopsys User Documentation version 3.4b, *VHDL Compiler Reference Manual*, Synopsys Inc., 700 East Middlefield Road, Mountain View, CA 94043-4033.
22. T.H. Szymanski, A. Au, M. Lafreniere-Roula, V. Tyan, B. Supmonchai, J. Wong, B. Zerrouk, S.T. Obenaus, “Terabit Optical LANs for Multiprocessing Systems,” *Applied Optics*, Vol. 37, No. 2, pp. 264-275, Jan. 1998.
23. T.H. Szymanski and H.S. Hinton, "Reconfigurable Intelligent Optical Backplane for Parallel Computing and Communications", *Applied Optics*, Vol.35, No. 8, pp. 1253-1268, Mar. 1996.
24. T.H. Szymanski and H.S. Hinton, "Architecture of a Terabit Free-Space Intelligent Optical Backplane", Journal of Parallel and Distributed Computing (JPDC), Nov. 25, 1998.
25. T.H. Szymanski and V. Tyan, "Error and Flow Control in Terabit Intelligent Optical Backplanes", IEEE Journal Selected Topics in Quantum Electronics, Special Issue on Smart Photonic Components, Vol. 5, No. 2, pp. 339-352, March/April 1999.
26. Z. Vranesic, S. Brown, M. Stumm, S. Caranci, A. Grbic, R. Grindley, et al, “The NUMAchine Multiprocessor”, CSRI Technical Report, University of Toronto.