TRANslucent Smart Pixel ARray (TRANSPAR) 
Chips for High Throughput Networks 
and SIMD Signal Processing 

Chih-Hao Chen, Bogdan Hoanca, Charles B. Kuznia, Member, IEEE, Alexander A. Sawchuk, Fellow, IEEE, and Jen-Ming Wu, Member, IEEE 

(Invited Paper)

Abstract—We present a novel architecture for an optical network, TRANslucent Smart Pixel ARray (TRANSPAR), having smart pixel devices which effectively function in an optically translucent manner. The network protocol is similar to carrier-sense multiple-access/collision-detection (CSMA/CD) commonly used in Ethernet, but adapted to a ring configuration using optical parallel packets in free space. The TRANSPAR devices also function as a fine-grain mesh-connected parallel pipeline array for image and video signal processing. We designed, fabricated, and are currently testing the TRANSPAR smart pixel devices and network node hardware. This paper presents the network architecture, tradeoffs, and design decisions, the testing results to date, and ends with considerations on practicality and future scalability.

Index Terms—Cellular logic arrays, CMOS integrated circuits, integrated optoelectronics, optical networks, smart pixels.

I. INTRODUCTION

S MART PIXEL architectures have advanced in recent years from the research stage to a useful complementary technology for optoelectronic interconnections and networks. This has been made possible by tremendous advances in the performance of the optoelectronic devices, and also by concentrated work on developing novel architectures that break away from the all-electronic mold into novel approaches more suitable for optical interconnects. While borrowing from their electronic counterparts, optical smart pixel networks have evolved a personality of their own, from photonic switching fabrics to photonic backplanes. This paper presents the design considerations and the practical implications (for design, testing, and applications) of such an optical network, the TRANSLucent Smart Pixel ARray (TRANSPAR) network. The TRANSPAR network is an optical carrier-sense-multiple-access/collision-detection (CSMA/CD) network, like the ubiquitous Ethernet. Unlike the Ethernet, it is a unidirectional propagation network with a ring topology. Also, unlike the Ethernet, it is built as an optical parallel packet network, using as nodes our smart pixel chip TRANSPAR. TRANSPAR is a smart pixel device having applications in very high throughput networks as well as in single instruction multiple data (SIMD) parallel signal processing, thus allowing the network to operate as a massively parallel pipelined computational system. By connecting TRANSPAR nodes to host computers (Fig. 1), the TRANSPAR network can provide both high-throughput connectivity between the hosts, as well as parallel high-speed processing power.

Our TRANSPAR chip design was fabricated by Lucent through the DARPA/GMU/CO-OP foundry program in 0.5-μm CMOS with flip-chip bonded multiple-quantum-well (MQW) optical modulators and detectors for optical I/O. In this paper, we present the TRANSPAR network architecture and the func-
Fig. 2. TRANSPAR chip layout and detail of one SIMD processor in the \(8 \times 4\) array.

Fig. 3. TRANSPAR ring network configuration.

tionality, the physical layout, and design principles behind the TRANSPAR chip itself, as well as preliminary results on a system demonstrator we are building with the TRANSPAR chips.

The research described here has applications in high-speed networks and their interfaces to advanced electronic computers, parallel pipeline image and signal processing, high-speed communication switching systems such as ATM, multimedia storage, archiving, processing, and distribution.

II. TRANSPAR FUNCTIONALITY

The TRANSPAR architecture and the physical smart pixel chip with the same name are designed to optimize two functions: 1) network interface for three-dimensional (3-D) data packet transfer between computing nodes using a Carrier-Sense-Multiple-Access/Collision-Detection (CSMA/CD) Protocol and 2) high-throughput SIMD-type processing of two-dimensional (2-D) data fields. By combining these two functions, the TRANSPAR chip can be used for parallel pipeline processing (Fig. 1). Multiple nodes on the network operate as SIMD computation engines. Each node performs a small set of operations on each data block and then transfers the data to the next stage in the pipeline for further processing. The nodes operate asynchronously, however packet transfer between two nodes is performed synchronously using a first-in-first-out (FIFO) elastic buffer. This scheme avoids the need for a global clock, which greatly relaxes the design and implementation of the network. Additionally, to increase the throughput, the network operates at a high clock rate, comparable to the on-chip clock, while the electrical interface between the TRANSPAR node and the host computer operates at only a fraction of this clock rate. This allows the electrical interface to be very simple, while allowing the powerful optical network to operate at high data rates.

A. General Considerations

The layout of the TRANSPAR chip is shown in Fig. 2. The chip is \(1.5 \text{ mm} \times 1.5 \text{ mm}\) in size, and was fabricated in the HP14TB 0.5-\(\mu\)m technology through MOSIS. After fabrication, a \(20 \times 10\) array of GaAs MQW diodes were flip-chip bonded to the CMOS logic. Pairs of diodes operate as dual-rail optical I/O, used either as modulators or as detectors, depending on the biasing. Not all diodes are actually being used in our implementation.

The chip operates as an array of parallel optical channels. Each optical channel consists of two pairs of diodes, one pair connected as modulators and the other as detectors. There are a total of 39 such channels: an \(8 \times 4\) array of data channels, three destination address channels, three source address channels, and one clock channel. Behind each data channel there is a general-purpose processing element (PE). PE’s are interconnected electrically using a mesh topology, and operate in SIMD fashion. Each PE contains memory and logic for performing arithmetic and logic operations. Additional circuitry handles the network protocol, the interfacing of the chip with a host computer and the clocking of the chip.

B. CSMA/CD Network Interface and Ring Architecture

The TRANSPAR nodes on a network are physically arranged as a ring (Fig. 3). Host processors are attached to the nodes and data transfers occur asynchronously via optical parallel data packets (OPDP’s) under control of a CSMA/CD Protocol. The TRANSPAR CSMA/CD Protocol is modified from the usual Ethernet to operate over ring networks that pass spatially parallel packets. The latency per node (including both the propagation time through the node and the optical propagation between two adjacent nodes) in a typical physical system is less than 4 ns.

Network nodes operate asynchronously, at on-chip rates of hundreds of megahertz. The interface with the host computer operates also asynchronously, but much slower, allowing for an inexpensive packaging and a simple design for the printed circuit board for the TRANSPAR nodes. For asynchronous operation, each packet includes a clock channel, and the packet is received into an 8-bit-deep elastic storage buffer (FIFO). The elastic buffer is usually employed at the interface between two systems which operate at different clock rates (here, the sender and the receiver chips).
To implement the optical network, the TRANSPAR chips are packaged with precision diffractive optical elements (DOE’s) and other microptic and macroptic components. The packaging of such systems (i.e., the integration of SP devices with optical devices into a rugged assembly) is extremely important for the transfer of this technology into real world applications. The DOE’s include: 1) diffractive microlens arrays, 2) combined diffractive microlenses with Dammann grating functions, 3) various Dammann grating spot array generators, and 4) interconnection DOE’s having more general output patterns.

Each PE contains a network interface, including an optical receiver (Rx), an optical transmitter (Tx), and an 8-bit FIFO buffer. This interface section of the PE uploads or downloads OPDP’s to or from the optical network using the modified CSMA/CD Protocol. The packets consist of eight frames, each frame transmitted on a separate clock cycle. Each frame contains an array of 4 x 8 bits of data, the source and address fields, such as required in image/video processing or packet header recognition and routing.

Unless the PE is uploading an optical data packet to the network, the PE is in transparent mode. In transparent mode, any signal entering the Rx passes directly on to the Tx, incurring only a small delay (~3 ns). When uploading an optical data packet, the PE is in opaque mode. Signals entering the Rx are blocked, while data in a local 32-bit SRAM located at each pixel is transmitted onto the network. Ideally, only a single TRANSPAR node on the network is uploading a packet at any given time. In this case, the transmitted packet travels through all other network nodes almost instantaneously, since they are transparent. Only the node whose address matches the destination address (encoded in the OPDP) downloads the OPDP, using the clock channel included with the packet.

A node with a packet to send must wait until the optical network is idle before uploading an OPDP. However, there is a chance that two or more TRANSPAR nodes will detect an idle network at nearly the same moment and upload OPDP’s that will cause contention (or a collision, in Ethernet terms). The TRANSPAR node contains circuitry to detect such a collision and reset the network.

C. SIMD Computing

For SIMD computing, each TRANSPAR chip contains an array of mesh-connected processing elements (PE’s) implemented by smart pixels. Although the area dedicated to each smart pixel is only 125 x 250 μm², it is sufficient to implement an ALU-based fine-grain SIMD processing element. Nearly all SIMD machines use this PE architecture, including the popular connection machine (CM) [1] and massively parallel processor (MPP) [2], [3]. In the TRANSPAR, these smart pixel PE’s are replicated into a 4 x 8 mesh-connected array and can perform SIMD processing using similar programming methods used in current SIMD machines.

The 1-bit ALU performs data processing on data stored within the PE’s 32-bit SRAM or within a neighboring PE’s SRAM. Although the ALU operates on single bits internally, the TRANSPAR node can be viewed as a computing system with a programmable word length. An on-chip finite-state-machine (FSM) receives macrocode instructions from an off-chip controller (or host computer) and converts them into series of microcodes that are broadcast across the PE array [1]. The host interface is thus insulated from seeing the bit-serial nature of the PE architecture. Table I shows a subset of PE instructions (microcode).

Each PE has three registers used to store intermediate values of the computations. Registers RA and RB provide interface between the ALU and the memory blocks and register RC holds the carry bit for arithmetic operations or the enable bit for PE operations. The enable bit allows the partitioning of the PE array into enabled and disabled PE’s, in which only enabled PE’s perform the instructions which are broadcast to the whole array.

For communication among the PE’s, data fields can either be shifted into the array electrically, via the mesh network (in a one-dimensional (1-D) row parallel format) or optically via the detectors integrated into the PE’s (as a 2-D array). Likewise, data is unloaded from the array through electronic channels in a row parallel format or optically transmitted in 2-D array format.

The pipelined SIMD processing and OPDP transfer can be combined to create a multinode SIMD processor. Each node is a mesh connected SIMD processing array and data transfers between nodes are made using the 8-bit deep, 8 x 4 bit wide data packets, based on the asynchronous CSMA/CD Protocol. Because the network can accommodate up to six nodes, it is equivalent to a 6 x 8 x 4, 3-D array of SIMD processors. This system can perform very fast parallel processing of 2-D data fields, such as required in image/video processing or packet header recognition and routing.

<table>
<thead>
<tr>
<th>Arithmetic</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>AND</td>
</tr>
<tr>
<td>Subtract</td>
<td>NAND</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Register Operations</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set</td>
<td>A &lt; &lt; A</td>
</tr>
<tr>
<td>Clear</td>
<td>A = 0</td>
</tr>
<tr>
<td>Enable</td>
<td>A &lt; &lt; A</td>
</tr>
<tr>
<td>Disable</td>
<td>A = 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Optical I/O</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET_FIFO</td>
<td>A = &lt; FIFO</td>
</tr>
<tr>
<td>XMT_B</td>
<td>Tx = &lt; B</td>
</tr>
</tbody>
</table>
III. DESIGN CONSIDERATIONS ON THE TRANSPAR ARCHITECTURE

In designing the TRANSPAR architecture we often had to make choices, faced with contradictory requirements of optimum performance and limited available resources. We believe that some of these architectural choices are of general interest, so we review some of the choices and explain the reasons behind them, before presenting the actual implementation of the architecture.

A. Network Protocol—Choice for CSMA/CD

Our starting point was to choose the network protocol. We chose to use the well-known CSMA/CD Protocol because it requires little overhead circuitry, it allows random access with no need for global synchronization (for the nonslotted CSMA/CD) and it can be used for optical networks with a minimal number of modifications.

B. Translucency

The next choice we made was to reduce the latency per network node as much as possible. The CSMA/CD Protocol [4] operates at the highest throughput when the ratio of the packet length to the propagation delay in the network is large. For a fixed packet length, this requires a small propagation delay through the network. If the packet size is allowed to vary, a minimum packet length must be enforced, to avoid severe degradation in the performance. To enforce the minimum packet length, empty bits may be stuffed at the end of any short packets. This clearly introduces unnecessary overhead. A much better choice, if possible, is thus to minimize the network delay.

In our previous network designs we had used a synchronous approach [5], [6], with all nodes operating at the same clock rate, synchronized with a global clock. Thus, if a packet had to propagate through multiple nodes between a source and a destination, it was delayed at each intermediate node because it had to be clocked in and clocked out, as in intermediate stages of a pipeline. This introduced a large, unnecessary delay. To reduce the delay, the intermediate network nodes between a source and destination should ideally be transparent, allowing a packet to pass through them with minimum latency. While other researchers have demonstrated such architectures using physically transparent nodes [7]–[9], the technology available through the CO-OP did not allow such an implementation. We then chose to design our nodes to be translucent, i.e., allowing the packet to propagate with minimum delay, through only an optical receiver, a minimum amount of logic and an optical transmitter. The latency is thus minimized, and is due mainly to the optical receiver.

C. SIMD Architecture

One application we envision for the optical network is in interconnecting the nodes of a massively parallel processing computing system. To achieve this, we included a powerful computational engine in each processing node. Due to the distributed, parallel nature of the optical packets, a pixel-based architecture was in order for the network nodes. For its ease of programming, we adopted the SIMD architecture, with PE’s in a node interconnected using an electrical mesh topology. Across nodes, each optical channel connects a PE with its corresponding PE’s in the other nodes on the network, as in a pipelined system. This is equivalent to a massively parallel architecture, with pipelined SIMD processors.

1) The Pros and Cons of SIMD: SIMD architectures are very attractive due to their ease of programming. Because all PE’s are operating in lock step and execute the same instruction, they are always synchronized. In addition, each PE is only connected with its immediate neighbors, eliminating most of the clock skew and clock-data race issues. Finally, SIMD architectures are naturally suited for some parallel problems in distributed computation, for example in image processing, finite-element simulations or database search applications.

For optical parallel networks, SIMD seems to be the ideal choice. Because 2-D arrays of data are transferred in parallel, each PE executes the same type of operation. Both parallel networks and SIMD are ideally suited for image processing applications or for interfacing of page-oriented optical memories. Also, SIMD processing may be useful for parallel processing of optical packets, implementing header recognition, detection, or updating.

2) Bit Serial ALU Architecture: We designed the architecture of the PE under the constraint of the pitch of the optical channel, which, in turn, was dictated by the available technology. The pitch of the optical channels was chosen to allow easy coupling of the free-space channels into an array of optical fibers, on 125-μm centers. For our system design, the pitch of the optical channels imposed a limited area per PE. In utilizing the available area, we had to find an optimal tradeoff between the amount of processing power and the amount of memory in the PE. We chose to use a powerful, yet small-area architecture for the processing logic, to leave more space for memory. We used a bit-serial ALU, in which the processing of a multiple-bit word is done bit by bit, in a serial fashion. This approach requires many clock cycles per operation, compared to a single clock cycle per operation for a full bit-parallel implementation, but the simplicity of the ALU allows a higher clock rate. In a full bit-parallel architecture, processing a long word or a short word take the same amount of time, while in our case the short word can be processed faster. This is because the clock period for a parallel ALU needs to be reduced to allow for the delay through the relatively complex structure, even if only a small number of bits are processed in a given instruction.

3) Microinstructions, Macroinstructions, and the Finite State Machine: To process a multiple-bit word, the bit-serial ALU operation runs a series of identical simple operations, so we were able to use a finite state machine (FSM) to operate on the individual simple operations. To the host computer, the resulting processing system looks like a parallel bit ALU with a programmable word length. The number of bits to operate on is stored in a register and can be configured dynamically, in software. The FSM is presented with macroinstructions that
are interpreted internally and broken into a series of many simple microinstructions, which are executed at high-speed [1]. The FSM is responsible for reading the individual bits from the memory, presenting them to the ALU and writing the results back into the memory. The macroinstructions form a reduced, yet powerful set of instructions, which require only a small number of bits, but which generate internally a large number of microinstructions bits.

4) Ways to Address the I/O Bottleneck: In using SIMD architectures, as well as in parallel optical networks, a major bottleneck is the I/O. Most of the available data sources are serial in nature, and the serial-to-parallel conversion introduces overhead and additional delays. Clearly, using such a data source would not be appropriate for the parallel architectures we describe here. Still, there are a few instances of architectures for which the parallel format is natural (Fig. 4):

• A serial communications channel may transfer data at high speeds, but when processing must be done on the data on the fly, the processing power available at the speed of the serial channel is insufficient. A parallel fan-out is required to be able to provide the required processing power. We have shown a possible architecture in which a pipelined processing system employs a serial-to-parallel converter as a part of the pipeline [5]. Stages along the pipeline are assigned to perform a small number of processing steps, over time periods comparable to the time for the serial-to-parallel conversion. This approach minimizes the effective delay due to the serial-to-parallel conversion by pipelining the conversion with the processing latency of the pipeline.

• Fire-hose systems [10] are processing systems in which a small amount of data at the input generates large amounts of intermediate data. In a pipelined system, the first stages may fan-out the input to generate the fire hose, while stages at the end of the pipeline may collect the results into an inverted fire hose. The central portion of the pipeline may be required to process massively parallel packets of data at high speeds.

• Last but not least, page-oriented memories are becoming interesting storage devices. The challenge here is to be able to preprocess (before recording to the memory) and to postprocess (after reading from the memory) large pages of data. The page (or frame) rates are relatively low, $10^3$–$10^6$ pages per second, but the large number of parallel channels makes the required processing throughput a challenge.

D. Unidirectional Network Propagation

Ethernet is a bidirectional network protocol, but we chose to implement the TRANSPAR as a unidirectional network. This allows us to use a ring topology, as well as a simpler optical system.

1) Ring Topology with CSMA/CD Protocols: The CSMA/CD Protocol is limited to bus or tree topologies. CSMA/CD networks cannot allow loops, where packets could propagate around the network for multiple times. On a bus or tree, terminators are used at the ends to cancel any reflections, ensuring that each packet passes by each node exactly once. In our modified CSMA/CD we use a ring with unidirectional propagation and an opaque source node, which allows us to propagate the packet exactly once along the network and to use the opaque source node as the terminator. This modified CSMA/CD is readily suitable for more robust protocols (hybrid token/random access [11] or the Ethernet piggyback [12]). Such hybrid protocols operate as CSMA/CD at low loads, when collisions between packets are unlikely, and become a token ring at high network loads when the performance of the CSMA/CD is unacceptable.

2) Simpler Optical System: Using unidirectional network propagation simplifies the optical system considerably. Our modulator based smart pixel system requires a fairly complex optical system to bring in the optical power supply [13] (the CW beams for input to the modulators) and to pass the modulated beams to the next stage. Bidirectional network propagation would further complicate the optical system with a fan-out stage and with two sets of optical detectors, to detect incoming optical packets from the two neighbors of each node on the network. A bidirectional propagation would be important if we used a handshaking mechanism between the network nodes, but with our design, the simpler unidirectional propagation is sufficient.

E) Low-Cost Electronic Interfaces

Low cost is the paramount consideration in all engineering design problems. In our case, the smart pixel chip and the OE devices flip-chip bonded onto it are batch produced, which can potentially lower the cost of the resulting TRANSPAR system. On the other hand, the packaging of the chip and the interfacing with the electronic host computer are more labor-intensive, thus more costly. We designed our architecture such that there are no high-speed lines going off-chip. This allows the chip to be packaged in a low cost package, and interfaced with low cost electronics.

To facilitate the testing of the chip, we provided a variety of clocking schemes. For high-speed operation, we designed a ring oscillator on-chip clock, as well as a PLL-based clock multiplier, which allows the on-chip clock to be synchronized with an external clock. The PLL may allow multiple high-speed systems to operate synchronized with a slower external clock [14]. Finally, we have the option of using a slow external clock (without multiplication), to perform low-speed testing on the chip (for example for the device characterization).

As a part of our low-cost design, we incorporated handshaking intelligence into the TRANSPAR chip, allowing the
2. electronic interface to operate asynchronously. Using our design, we are plug-in compatible with a variety of digital I/O boards readily available commercially, one of which we are currently using in our testing setup.

Another consideration in lowering the cost of the optical interface is reducing the number of power supplies. Our current design requires a total of seven power supplies. Four supplies are for the modulator and detector biasing, one is for the digital logic, one for the receiver circuitry (separate power supply, to avoid noise crosstalk from the digital circuitry) and one power supply for the high voltage modulator driver (described in Section III-G). This large number of power supplies is only required because TRANSPAR is a prototype, and requires multiple degrees of freedom for testing. In a later version, we could combine several of the power supplies into a single one.

Due to the limited number of wire-bond pads, some of the pads have multiple functions. This pad multiplexing is just a first step in encoding the macroinstructions in an optimal way. Further optimization could lead to a dramatic reduction in the number of pads, which would further reduce the cost of the electronic interface.

F. High-Speed Optical Network Interface

The optical devices on the TRANSPAR chips have been demonstrated to operate at gigabits-per-second rates [15]. To take full advantage of the available speed, we tried to maximize the clock rate at which the logic for the optical network can operate. For this, we have used a minimum amount of logic on the optical paths, as well as an asynchronous transfer protocol. Packets are sent directly from the memory to the transmitter circuitry, propagate with minimal latency through the intermediate nodes and are received asynchronously in a FIFO on the destination chip.

We also had the choice between a fixed and a variable length optical packet. We opted for the first choice, to simplify the design of the protocol, again in order to attain the maximum transfer speed. A variable packet length would require additional logic to determine the end of a packet, which would slow down the link. The limitation with a fixed packet size may be that some of the smaller packets may need to be padded, but if the processing takes place in word sizes that are multiples of the packet length, no padding is necessary.

G. High Voltage Modulator Driver

Finally, at the lowest design level, we decided to use a high voltage driver for the MQW modulators. We discovered from past experiences that the MQW OE communications link is limited by the transmitter side, namely a combination of the nonuniformity of the spot array generator and the modulator contrast ratio. The receiver sensitivity is extremely good at the sub GHz clock rates we are using. Unfortunately, the spot uniformity of the optical power supply for the modulators is fairly poor.

To understand the importance of the uniformity of the optical power supply, consider a pair of spots incident on a modulator pair for one dual rail OE channel. Ideally, the spots have equal power before they are incident on the modulators. Let us consider that after modulation the left spot is to be of higher power than the right spot for a “ZERO” and the left spot is to be of lower power than the right spot for a “ONE.” If the left spot is of higher power even before the modulators (due to an imperfect optical power supply), the modulator pair will have a difficult job in generating a “ONE,” but it will easily generate a “ZERO.” A high modulator contrast ratio can help balance out the errors due to the nonuniformity of the optical power supply.

As an added problem, the TRANSPAR chip is fabricated in a high-performance technology, and is to be operated at 3.3 V. To ensure sufficient contrast ratio for the modulators, we decided to use a high-voltage modulator driver, which applies a voltage of 5 V (or higher, adjustable independently of the voltage supply for the logic). According to preliminary data from Lucent Technologies, this change from 3.3 to 5 V should yield a contrast ratio improvement by a factor of 1.5...2.

IV. TRANSPAR LAYOUT AND DESIGN CONSIDERATIONS

A. Smart Pixel Technology

The TRANSPAR was fabricated using the Bell Laboratories division of Lucent Technologies foundry for optoelectronic VLSI technology. This foundry is administered under contract with the CO-OP Program (Consortium for Optical and Optoelectronic Technologies in Computing) at George Mason University. This foundry is capable of flip-chip bonding arrays of GaAs multiple-quantum-well (MQW) devices onto standard CMOS silicon integrated circuit.

The silicon CMOS circuitry for TRANSPAR was fabricated in the 0.5 μm (λ = 0.85 μm) MOSIS HP-CMOS14TB process. On the logic, the foundry flip-chip bonded an array of 10 × 20 MQW devices on a 62.5 × 125 μm grid, each device having an 18-μm optical window. MQW devices operate either as p-i-n detectors (with 0.5 A/W responsivity) or as reflective optical modulators with reflectivity between 0.1 and 0.3 (for a 5-V voltage swing) at a wavelength of 850 nm.

The TRANSPAR chip was designed using standard IC CAD software. A high-level description in Viewlogic VHDL was written and then simulated at gate level. The layout was partly synthesized using Cascade’s Epoch design tool, and partly custom designed using Magic. The layout functionality was then verified using the IRSIM tool for digital (logic level) simulation and H-SPICE for analog (transistor level) simulation. The simulations indicate correct operation for on-chip clock speeds exceeding 133 MHz.

B. Mesh Connected Smart Pixels

The TRANSPAR contains a 4 × 8 array of processing element (PE) smart pixels. Fig. 5 shows the physical layout of a single PE. Each pixel is mesh connected to its north, east, west, and south neighbors. To allow row-parallel I/O with the PE array, the PE’s on the east border of the array receive electrical data from wire-bond pads, and the PE’s on the west border send electrical data out to wire-bond pads. All

1 [On-line]. Available HTTP: http://co-op.gmu.edu/
Fig. 5. Processing element (PE) smart pixel is $125 \times 250 \mu m^2$ and contains 550 transistors.

Fig. 6. The smart pixel PE consists of an ALU-based data processing section for SIMD operation (bottom) and an optical network interfacing section (top).

The processing engine of each PE is a bit-serial ALU. The ALU can implement any binary function of two bits. ALU operations use the two registers RA and RB. The register RC is used to store the carry bit for multiple bit arithmetic operations. The bit in register RC can also be used to execute conditional operations with the PE. If the MASK bit is also set, the PE performs the indicated operation only if the bit in RC is zero. This allows only a subset of the PE’s to execute an operation for example in case of arithmetic overflow, only the PE’s in which overflow has occurred will perform a cleanup sequence.

The data storage in each PE includes two memory blocks. One 32-bit SRAM memory is used as general purpose storage for the source and the destination of the ALU operations. A second SRAM block, only 8-bits deep is used as an elastic buffer for the asynchronous packet reception from the optical network. Both memory modules are based on a standard six-transistor SRAM cell design with precharge and sense read-out. Each SRAM block has a single bidirectional port, used for both input and output.

The 32-bit memory is used to store the data for the ALU operations. To minimize the number of input signals, the address locations of memory are not directly visible outside the TRANSPAR node. The only way to access the memory is through the FSM, using a pointer based access. Three registers store two source addresses and one destination address, which are used when performing ALU operations. A fourth register stores the length of the words to be operated on. Thus, an ALU operation deals with an array of bits in the memory starting at the address pointed to by either the source or the destination addresses and of length specified in the word length register (Fig. 7).

The FIFO buffer is used as elastic buffer for the asynchronous reception of the optical packets. The FIFO addresses are not available outside the TRANSPAR node. They are internally controlled by the network interface (for reading from the network) and by the FSM (for writing from the FIFO into the 32-bit PE memory). To simplify the architecture, the FIFO has a single I/O port, and cannot perform simultaneous read and write operations. While an incoming packet is being loaded into the FIFO, the FIFO_FULL bit is 0; the bit is set only after the whole packet has been successfully received. When the host computer requests the FIFO to be read into the 32-bit memory, the FIFO is automatically reset. In case of collision, the node must be explicitly reset, because the FIFO may be in an undefined state.

C. State Register

The 15 bits stored in the state register (SR) determine the behavior of the chip. These bits store the setting for the clocking scheme (internal, external, or internal locked to external using the PLL), the source and address pointers, the word length and additional settings that are used for testing purposes. For precaution, an asynchronous reset pin is used...
According to the clocking is based on the external clock, without the frequency of the chip pins is dedicated for extracting the on-chip clock for measurement purposes as well as for interfacing with the off-chip circuitry. Initially, after an asynchronous reset the for testing, as well as for interfacing with the off-chip logic. The clock distribution (internal or external), the SR value may be changed at any described above. To select the clock speed and the clock source provide a clock range of 125–250 MHz. The external clock can be applied directly as the chip clock or can be frequency multiplied on-chip using a phase-lock loop (PLL) source can be applied directly. After the RESET, if all the circuitry is functioning, the chip can be programmed to operate using the FSM and/or the internal or the multiplied clocks. This ensures that the chip can be tested even if part of the on-chip timing and control blocks are faulty.

D. Internal Clock Generator

The timing of the chip can be obtained either from an internally generated clock (based on a ring oscillator VCO) or from an external source. The internal clock is tunable using an analog input voltage (at pin CK_ADJ). According to the HSPICE simulation, the frequency range is 250–500 MHz for a tuning voltage of 2–5 V. The on-chip frequency divider can provide a clock range of 125–250 MHz. The external clock source can be applied directly as the chip clock or can be frequency multiplied on-chip using a phase-lock loop (PLL) and digital frequency dividers (T flip-flops); the frequency multiplication is by a factor of 16. Additionally, a frequency divider can be used for both the internal and the external clock sources to halve the clock rate, for even more flexibility. This allows the chip to operate under a wide range of clock rates, for testing, as well as for interfacing with the off-chip logic.

The programming of the timing block is done using the SR described above. To select the clock speed and the clock source (internal or external), the SR value may be changed at any time during the operation of the chip. The clock distribution circuitry ensures that the clock rate is changed smoothly, i.e., no clock pulses are clipped during the transition period. One of the chip pins is dedicated for extracting the on-chip clock for measurement purposes as well as for interfacing with the off-chip circuitry. Initially, after an asynchronous reset the clocking is based on the external clock, without the frequency multiplier, to ensure a reliable start of the on-chip logic. For using the other clocking options, the SR must be subsequently updated with the desired values.

E. Finite State Machine (FSM)

The FSM is used as a buffer between the fast on-chip electronic logic and optical networking circuitry on one hand, and the slower off-chip interface to the host computer on the other hand. The 0.5-μm technology used for the TRANSPAR is extremely fast, with the single-gate delay of the order of 150 ps. The optical I/O devices used (MQW modulators and MQW receivers) have been shown to operate at data rates exceeding 1 Gb/s [15]. Unfortunately, the electronic I/O channels at the chip pins are much slower. Connecting the TRANSPAR through an inexpensive printed circuit board and an interface to a general-purpose commercially available computer may limit the per channel speed to a few tens of megahertz. This is a typical example of fire hose architecture [10], where the periphery of the chip is only capable of modest numbers of channels operating at low speeds, but the core circuitry can accommodate many more high-speed channels. To take advantage of the high-communications bandwidth of the optical I/O and of the high computational bandwidth of the on-chip processing, we separated the fast on-chip optoelectronic part and the slower off-chip electronic part, using the FSM as a buffer. This architecture allows the use of an on-chip clock rate of 125–250 MHz with a much slower off chip clock rate.

The FSM operates on-chip, at the on-chip clock rate of 125–250 MHz. It reads macroinstructions from the off-chip interface using a slow asynchronous four-way handshaking protocol. The FSM then converts each macroinstruction into multiple microinstructions that are applied to the on-chip logic and executed at the on-chip clock rate [1]. For this, the FSM directly controls the operation of the pixels. The control lines of the PE’s are not directly accessible from the chip pins (except when running in debugging mode, as described below). Due to the bit-serial pixel architecture and due to the local data access, each 1-bit operation can be performed very fast. The FSM executes long sequences of such fast operations in parallel on all PE’s, achieving a very high computational throughput for the on-chip data.

A macroinstruction for the FSM can be of one of four types: load source/destination address registers, load state register, bypass FSM, and execute microinstructions. The load instructions are used for loading data from the electronic I/O pins of the chip. The load operations are slow, due to the electronic I/O bottleneck, but they only occur infrequently, when a change of context is recorded in the SR. The bypass FSM instructions are used for testing and debugging: the FSM provides only handshaking, all other signals to the PE array are applied via off-chip connections. The execute macroinstructions are the most used, and they are well optimized for achieving high computational on-chip bandwidth. Execute macroinstructions can be of four types: 1) two-operand; 2) single-operand; 3) transmit; and 4) receive.

For network functionality, TRANSPAR is capable of sending OPDP onto the network or receiving packets from the network. The transmit instructions send data from the local
memory to the optical network. To transmit an OPDP, each PE moves onto the PE’s optical transmitter a block of bits starting at the location pointed to by the source address register. If the network is busy, the FSM executes wait states until it can transmit the packet. At that point, the bits are output serially from the SRAM of each pixel, for all pixels in parallel. For each bit transmitted, the FSM generates a clock pulse on the optical clock channel. After transmitting the OPDP, the FSM sends a cleanup sequence to remove the tail of the packet from the optical network. This way, each transmitting node is responsible for removing its packet from the network once the packet has propagated for one roundtrip.

In turn, the receive instruction reads a packet from the FIFO (where it is buffered upon receipt from the network) into the local memory. Each incoming OPDP is accompanied by its own clock, which writes the packet in the FIFO memory of the PE’s, independently of the on-chip clock of the receiving chip. When a TRANSPAR chip has received an OPDP, the FIFO full (FF) flag is set, to notify the host computer. At this point, the host computer can request the packet to be transferred into the SRAM. The FSM then moves the packet from the FIFO memory into the SRAM, starting at the location specified in the destination address register. Because the FIFO can only store a single packet, if a second OPDP is received before the host can read out the FIFO, the second packet is ignored (lost). In this case, the packet loss (LOSS) flag is set, to inform the host computer.

In addition to generating the control signals for the PE’s, the FSM also generates the address decoding for the SRAM. Pixel addresses are encoded as a 4-bit column address and 4-bit row address. In executing a macroinstruction, three counters are used to access the successive bits of each word (the two operands and the result). The three counters are initialized with the contents of the pointer registers and then incremented after each bit-wise operation. Another counter is initialized with the word length and then decremented after each bit-wise operation. When this counter reaches 0, this indicates the completion of the execution of the macroinstruction.

F. Network Interface

The physical layout of TRANSPAR network interface contains: carrier sense module, collision detection module, source address pixel, destination address pixel, optical clock pixel, and FIFO control. The architecture of the carrier sense module and the collision detection module will be described in Section IV-G, when the network functionality of these modules is discussed. We now present the layout of the other network modules in the TRANSPAR node.

1) Source Address Pixel: Three source-address pixels in each frame identify the sender node. These source-address pixels have four states: 1) idle; 2) send; 3) silent; and 4) jam. When idle, they simply remain transparent and transmit the incoming source address from the network. When the node wishes to send a packet, the source-address pixels become opaque and send out the address of the node. When finished sending the packet, the pixels remain opaque during a silent period, when they send out zeros to remove the packet data from the network. Finally, in the case of collision, the source address pixels send out a jam signal to the network, to ensure that all nodes have sensed a collision. As explained in Section IV-G-4, this is needed because only the sender nodes can identify a collision reliably.

2) Destination Address Pixel: Three destination address pixels in each frame identify the destination node. Every node on the network compares the incoming destination address with its own address to see whether there is a match. The match is detected on the destination address bits in parallel, within the first cycle of the clock transmitted with the data packet. This contrasts with serial network methods that require several clock cycles to determine the packet address. When an address match occurs, the node activates its FIFO control and downloads the incoming data packet.

3) Optical Clock Pixel: The TRANSPAR frame contains an optical clock channel that defines the bit rate of the data packet. The optical clock pixel has three states: 1) idle, 2) send, and 3) match. When idle, the pixel is transparent and retransmits the optical clock from the previous node to next node. When in the state of send, it sends out an optical clock defined by the on-chip clock. The clock is straddled with the data, ensuring that the receiver node will sample the data when it is most stable (Fig. 8); this clocking scheme is also more tolerant to clock-data skew. Finally, the clock pixel is in the state of match, when there is an address match. In this case, the optical clock pixel directs the incoming clock to the FIFO control for data download.

4) FIFO Control: Each PE in the TRANSPAR node has an 8-bit deep FIFO buffer for data packet download. This FIFO is controlled by the FIFO control, ticking at the rate of the incoming optical clock. The received packet is written in the FIFO buffer with the incoming packet clock and then read out by the PE’s with the local, on-chip clock. Therefore, no global clock synchronization is required across network nodes.
an optical clock, the source address bits are sampled with
address bits as carrier sense. Because each packet contains
circuitry to simply use the logical OR of the incoming source
is busy. The design is simple, allowing the carrier detection
source address signals an idle network; otherwise the network
by detecting the three bit-wide source address. The “all zero”
on a carrier sense mechanism. TRANSPAR senses the carrier
Protocol. The random access to the shared medium is based
by the TRANSPAR network can be configured
computer nodes, and various intelligent systems within a building
or a campus. The TRANSPAR network can be configured
as a ring connected topology embedding the carrier sense
multiple access with collision detection (CSMA/CD) protocol.

1) Ring Interconnected Network: The configuration of a
LAN is generally limited to either the bus or the ring
topologies as shown in Fig. 9. In the case of the bus, a
node transmits the signal over the network in both directions
simultaneously. Terminations at each end of the bus absorb the
signal, avoiding multiple reflections on the network. For the
case of the ring topology, one-way transmission or two-way
transmission (as in FDDI) can be used. The physical medium
of the network can be coaxial cable, twisted pair wire, optical
fiber, or any medium that can carry signals. In the case of
TRANSPAR, we use CMOS/MQW smart pixels technologies
to generate, modulate, and receive digital signals in free space.

2) Random Access: The random access scheme CSMA/CD,
which is the basis of Ethernet, is one of the most popular
physical layer local-area network (LAN) protocols used today,
because it strikes a good balance between speed, cost and
ease of installation. These strong points combine with wide
acceptance in the computer marketplace to make Ethernet an
ideal networking technology for most computer users today.

For optical networks, we believe that the main advantage
of CSMA/CD is the random access. Optical networks are capable of sending data at gigahertz speeds, when global
synchronization would be impractical. With random access,
networks can use nodes that operate asynchronously, and
possibly with widely different clock rates.

3) Carrier Sense Multiple Access: The CSMA/CD Proto-
col we are using is a slight modification of the Ethernet Protocol. The random access to the shared medium is based
on a carrier sense mechanism. TRANSPAR senses the carrier
by detecting the three bit-wide source address. The “all zero”
source address signals an idle network; otherwise the network
is busy. The design is simple, allowing the carrier detection
circuitry to simply use the logical OR of the incoming source
address bits as carrier sense. Because each packet contains
an optical clock, the source address bits are sampled with
this packet clock at the receiver chips, to allow asynchronous
operation of the source and destination chips.

When a node has data to send, it first listens to the channel
to see if any other node is transmitting at that moment. If
the channel is idle, the node sends the data right away. If the
channel is busy, the node waits and tries to send again until the
channel becomes available, as in the 1-persistent CSMA/CD
[4]. Each network packet contains eight frames. Each frame
contains a 3-bit wide source address header and a 3-bit wide
destination address header. In our particular implementation,
the network can accommodate up to six independent nodes
with addresses from “001” to “110.” Each node has the
same priority to access the network. Each node listens to the
network by sensing the source address bits. The node considers
the channel idle if all zero source address bits are detected;
otherwise, the channel is busy.

4) Collision Detection: Collisions happen when two or
more nodes try to transmit data at the same time. Because of
the nonzero propagation delay between nodes, two or more
nodes could sense the channel as idle and begin sending
data at the same time, causing a collision. When a collision
happens, all colliding packets are garbled. The nodes involved
must step back for a random length of time and retransmit
the collided packets later. The challenge of collision detection is
to have every node detect the collision automatically by
listening to the channel.

Normally, only one node on the network is opaque (sending)
at any given moment, and all other nodes are in transparent
mode. Because of this, the sender expects to receive its own
packet after a round-trip delay. If a collision has occurred,
a second node is opaque and transmitting a packet on the
network, so the first node will receive a packet with a
source address different from its own. This is a very reliable
mechanism for detecting a collision, but it is only accessible
to the sender.

To ensure that all other nodes are aware of the collision, the
sender node broadcasts a jamming signal. The physical reason
of this signal is very different from the Ethernet implementa-
tion of CSMA/CD, where the network signals are analog and
the jamming signal ensures collision detection in the presence
of widely different power levels in the colliding packets.
Because of the unidirectional ring in our implementation,
signals from two colliding packets do not interfere in the
channel medium, so it is not a question of signal dynamic
range. In other words, only one packet can propagate between
two transmitters on the TRANSPAR network. The transparent
nodes in-between have no way of deciding whether this packet
was entirely sent by a single node (a valid transmission) or
made up of frames from more than one transmitting node
(collision). It is the task of the sender nodes to inform all
the nodes on the network about collisions.

Fig. 10 shows an example of collision detection, which
assumes propagation delay of 5 ns per node and packet length
of 40 ns. Node 1 tries to send a packet to node 3. After 10
ns, node 5 senses the channel to be idle and tries to send a
packet to node 2. Node 1 detects the collision at 20 ns when it
receives packet from node 5. Also node 5 detects collision
at time of 20 ns when it receives packet from node 1.
5) Packet Removal: Each node that transmits a packet on the network is responsible for removing the packet after a round-trip propagation, as in the case of FDDI. For this, the node remains opaque for a period after the end of the packet. We call this silent period, because the node sends out an empty packet, with all data and address bits zero. For a node down the line, this empty packet is interpreted as an idle channel. Another node may start transmitting as soon as it senses the silent period. On the other hand, if no other node has data to send, the silent period of the transmitter ensures the removal of the tail of the packet. This silent period is very important. If the transmitter would become transparent immediately after sending the end of the packet and all other nodes would be already transparent, the tail of the packet (which is still propagating on the network), would continue to propagate forever, keeping the carrier sense always high.

V. SIMULATION AND SYSTEM INTEGRATION

The TRANSPAR optical system provides one-to-one imaging of optical transmitters on one TRANSPAR node onto optical detectors on the next node. This provides the unidirectional data transfer required for the ring network. We will implement the TRANSPAR network using two types of optical systems, one using “macrooptic” imaging and another using “microoptic” components. The macrooptic system will use a single objective lens to image the entire field of smart pixel optical I/O ports. The advantage of this optical system is that it can be constructed entirely with off-the-shelf components (with the exception of the diffractive spot array generators for modulator readout). The microoptic system uses custom microlens arrays, with a lens dedicated to each optical channel. This system is more cost-effective when scaling to larger smart pixel arrays, though it is more costly to fabricate, and even more costly to align.

We have simulated all the blocks of the TRANSPAR architecture at logical level (VHDL), at intermediate level (IRSIM) and at device level (H-SPICE). The PE was simulated at 133 MHz, with the clock speed limited by the memory design. The FSM operated at up to 250 MHz. Additionally, we simulated the PLL operation in the frequency multiplier, as well as the switching between the internal and external clock signals. The simulations showed that the transients when switching the clock source did not clip any of the clock pulses. Network simulations showed that the CSMA/CD module operated correctly under various traffic conditions, detecting and handling collisions gracefully.

We recently received the fabricated IC’s with MQW diodes flip-chip bonded on the CMOS logic. The chip was mounted on a printed circuit (PC) board and interfaced with a computer. Due to the intelligence we built into the interface, the PC board required only bypass capacitors for the power supplies and a connector for a high-speed digital I/O board in the computer.

Preliminary tests indicate that the circuitry is operating as designed. The clocking circuitry was tested and shown to operate over the intended range (Fig. 11). Switching among the different clocking schemes occurs smoothly. We developed a computer program that can parse a file of high-level SIMD instructions into TRANSPAR binary macroinstructions and send them to the chip to perform image processing operations. The full test of the TRANSPAR under realistic network and computational conditions is under way and we will soon present a more detailed experimental demonstration.

VI. SCALABILITY

For a practical implementation, the TRANSPAR CSMA/CD network and the SIMD processing core can be scaled in multiple dimensions, to achieve a higher throughput. Some of the dimensions involve simply a linear increase in the resources (hence a linear increase in costs), but some of the other dimensions involve more complex changes, either of architecture or of the fabrication technology itself. In this section, we investigate the maximum aggregate throughput that can be expected from scaling the TRANSPAR architecture.

A first scaling dimension is the number of PE’s used within a node in the network (the “area” of the OPDP). By augmenting the area of the chip to a standard CMOS size...
(2 cm × 2 cm), the number of PE’s could be increased to 50 × 100. At this chip size and at the on-chip clock speeds possible with TRANSPAR (hundreds of MHz), no special circuitry is needed for driving the on-chip lines. Only fan-out buffers will be required for SIMD instruction and clock distribution, given the much larger number of channels. This will slightly increase the area consumption, as compared to linear scaling. The buffers will also increase the latency of the instruction distribution network. Beyond the standard chip size, the number of channels can further be increased using multichip modules or using wafer scale processing. Unfortunately, scaling the optical system beyond thousands of channels may be a much more difficult task. In addition to the limitations due to optical aberrations, the optical power to be distributed to such a large number of devices (for a modulator-based system) will severebly limit the number of channels. In conclusion, the maximum number of data channels is of the order of \( N_{\text{data}} = 10^{4} \).

As a second scaling dimension, the OPDP can be increased in depth, by increasing the amount of pixel memory. This can be accomplished using one or a more of the following three techniques:

- Using dynamic memory (with a single transistor per bit) instead of static memory (which we currently used and which requires six transistors per bit).
- Increasing the pitch of the optical channels, and dedicating the extra area per pixel to memory.
- Using a higher performance CMOS technology, with a smaller feature size (the feature size was 0.5 μm for our demonstrator chip).

These techniques would allow “deeper” OPDP to be transmitted on the TRANSPAR network. Because the overhead of packet transmission is independent of the packet depth, this would allow an increase in the throughput of the optical network. Moreover, an increase in the packet depth (i.e., length in time), would improve the throughput if the network propagation delay remains constant [4]. On the other hand, sending larger packets reduces the granularity and increases the latency. Combining an increase in density (from a 0.5-μm technology to a 0.15-μm technology) and the use of dynamic memory (an area reduction with a factor of approximately 5), the amount of memory per PE could be increased up to 1 kbit.

A third scaling dimension is to increase the number of chips (nodes) on the network. An obvious prerequisite for this is that more address bits be used for source and destination. Most likely, the limitation on the number of nodes on the network will not be due to the address size, but due to architectural considerations. One limitation comes from the nature of the network, a shared ring. As too many nodes are added on the ring, the probability of collision will increase to the point where the network will no longer be usable. Another architectural limitation to increasing the number of nodes results from skew.

As network nodes are physically cascaded, the skew or differential delays among the different channels increases. At each node the OPDP is detected and retransmitted, but not retimed. While this eliminates penalties due to crosstalk, noise and loss on the network, it allows skew to accumulate across the nodes. The differential delays are caused by nonuniformity across array components in the OE link: the spot array generator optical power, the modulator contrast ratio, the detector responsivity, and finally the receiver threshold. Fig. 12 shows the latency (defined as the maximum delay for an incoming optical pulse, from the input of the receiver circuitry to the output of the modulator driver), as a function of the detected photocurrent at the receiver. For this plot, the capacitance of a single-MQW diode is assumed to be 100 fF, as in [16]. With an input power of 100 μW per modulator diode (limited by device saturation), a reflectivity of 0.1 ... 0.3 and a detector responsivity of \( R = 0.3 \text{ A/W} \), the nominal detected photocurrent is about 6 μA, even considering zero coupling losses between the modulator and detector. The latency is thus close to 3 ns per node.

Still, latency is not a major concern on a network, where the propagation delays are not always known in advance. As mentioned above, for a parallel network, skew is the most critical parameter. Fig. 13 shows the channel skew (the
differential delay between a channel and its neighbors) referred to the ideal case, when the diode capacitance is 100 fF and the photocurrent is 6 μA. To estimate skew, we assume that the design parameters of the devices can vary within a ±10% range. The skew due to the variation in the diode capacitance is \( \Delta t_{\text{Cap}} \leq 0.05 \text{ ns} \). Assuming the same 10% variation in \( m \) as proposed above, the skew due to the uniformity of the spot array generator, the modulator reflectivity, and the detector responsivity, the uncertainty in the photocurrent is close to 30%. The skew due to such a large uncertainty in the detected photocurrent is \( \Delta t_{\text{Photo}} \leq 0.4 \text{ ns} \). The total (worst case) skew per TRANSPAR node is \( \Delta t_{\text{Node}} \leq 0.45 \text{ ns} \). In a six-node network, the overall skew may reach \( \Delta t_{\text{Network}} \leq 2.7 \text{ ns} \). To minimize the deleterious effects due to the skew, the clock period must exceed about ten times the amount of skew, \( T_{\text{Clock}} \geq 10 \Delta t_{\text{Network}} \). This will limit the clock rate to 30 MHz!

To address the issue of skew, a look at Fig. 12 shows that by operating at larger photocurrent values the slope of the latency curve (and hence the skew) will be significantly reduced. This can be achieved by using a higher modulator reflectivity, a higher modulator saturation power and a higher detector responsivity. Using a 20-μA photocurrent as the nominal operating point would reduce the skew due to the photocurrent to \( \Delta t_{\text{Photo}} \leq 0.06 \text{ ns} \). Ultimately, this would allow a clock rate of 150 MHz. On the other hand, if the number of nodes on the network is increased to say 100, the clock frequency must again be reduced to about 10 MHz. Clearly, our architecture can provide very high throughput, but due to the delay of retiming it is limited to a small number of nodes in order to operate at high clock rates.

We have just argued that the clock rate is limited by the packet skew. As the device performance improves, we can expect the skew to be reduced. Additionally, skew can be eliminated using circuit design techniques that extract a clock for each channel and resynchronize the received bits using an elastic buffer. This may remove the clock speed limitation due to skew. Beyond that, the network protocol does not limit the clock rate, because the communication is asynchronous (packets are received independently of the on-chip clock). The clock speed is thus limited only by the on-chip propagation delay. This last limit can be pushed further by:

- using a finer feature size technology, 0.15 μm as proposed above;
- using a faster memory design;
- or incorporating a pipelined tree for instructions distribution.

We estimate that the maximum clock speed after these improvements could reach 500 MHz, even for a network of up to 100 nodes, limited by the packet skew.

Summarizing these predictions, a full-scale version of TRANSPAR may have 10,000 PE’s per node, each PE with 1 Kb of memory, for a total of 10 Mb per node. Operating at 500 MHz, an optical transfer would allow a bursty throughput of 5 Tb/s. A full-memory transfer (copying the whole memory space of a source node to a destination node) would take 2 μs. The network could accommodate up to 100 nodes, each operating at OC192 speeds, using a store-and-forward mechanism, whereby nodes receive data from a host at OC192 speed, 10 Gb/s, convert the data from serial to parallel and send it on the network in bursts of 10 Mb.

VII. CONCLUSION

We have presented a novel smart pixel system design which combines SIMD processing and optical networking into a powerful pipelined processing architecture scalable in multiple dimensions. The networking concepts implemented in TRANSPAR are adapted from the Ethernet Protocol, CSMA/CD, to provide high-performance full-access networking using 3-D optical data packets. This architecture illustrates the unique advantages offered by the use of parallel free-space optical data transfer in a smart pixel system. Additionally, we presented tradeoffs and the design decisions that are of general applicability, beyond the particular implementation of TRANSPAR.

REFERENCES

Chih-Hao Chen received the B.S. degree in electrical engineering from the National Taiwan University in 1990, the M.S. degree from the University of Southern California (USC), Los Angeles, CA, in 1993, and is currently a Ph.D. candidate at the University of Southern California. He is a Research Assistant at the Signal and Image Processing Institute in the Department of Electrical Engineering at USC. His research interests include diffractive optical element design, optoelectronic circuit design for smart pixel arrays, and optoelectronic system design for image processing and networking.

Bogdan Hoanca received the electronics engineer degree from the Polytechnic Institute of Bucharest, Romania, in 1992, the M.S.E.E. degree from Syracuse University, Syracuse, NY, in 1994, and the Ph.D. degree in electrical engineering from the University of Southern California (USC), Los Angeles, CA, in 1999.

He is currently with the Integrated Media Systems Center at USC. His research interests include high-throughput all-optical networks (both fiber and free-space) and smart pixel optoelectronic VLSI demonstration systems.

Charles B. Kuznia (M’95) received the B.S. degree in electrical engineering from the University of Minnesota, Minneapolis, in 1987, and the M.S. and Ph.D. degrees from the University of Southern California, Los Angeles, CA, in 1989 and 1994, respectively.

He is a Research Assistant Professor at the Signal and Image Processing Institute in the Department of Electrical Engineering at the University of Southern California. His research interests include optical interconnects for parallel computing and networking circuitry for smart pixel arrays, diffractive optical element design, and fabrication.

Alexander A. Sawchuk (S’65–M’71–SM’79–F’94) received the S.B. degree in electrical engineering from the Massachusetts Institute of Technology, Cambridge, MA, in 1966, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1968 and 1972, respectively.

He was employed by the NASA Goddard Space Flight Center and Communications Satellite Corporation, and joined the University of Southern California (USC), Los Angeles, CA, in 1971. At USC, he is currently a Professor in the Department of Electrical Engineering, Associate Director of the Signal and Image Processing Institute, and Deputy Director of the Integrated Media Systems Center. He was Director of the Signal and Image Processing Institute from 1978 to 1988 and from 1990 to 1991.

His research interests include the use of optoelectronic devices and systems for high information capacity parallel optical computing, interconnecting and network applications. He also works in the areas of digital image processing, and signal processing, and is the author or coauthor of more than 130 technical publications in these fields.

He is a Fellow of the Optical Society of America (OSA) and the International Society for Optical Engineering (SPIE). He received the 1980 Halliburton Award for Exceptional Service, the 1993 Lockheed Senior Research Award, and the 1995 Outstanding Teaching Award from the USC School of Engineering. He is a Member and Delegate, United States National Committee for the International Commission for Optics (USNC-ICO) of the US Liaison Committee of the International Union of Pure and Applied Physics. He has served often in various offices and on technical committees for the IEEE, OSA, and SPIE. He was an Associate Editor for the OSA journal, Optics Letters, from 1985 to 1988, and was Editor of the Information Processing Division of Applied Optics from 1990 to 1996.

Jen-Ming Wu (S’88–M’99) received the B.S. degree from National Taiwan University, Taipei, R.O.C., in 1988, the M.S. degree from Polytechnic University, Brooklyn, NY, in 1991, and the Ph.D. degree from the University of Southern California (USC), Los Angeles, CA, in 1998, all in electrical engineering.

He served as a Graduate Research Assistant for the Department of Electrical Engineering Systems, USC, from 1994 to 1996, and for Integrated Media Systems Center, USC, from 1997 to 1998. At USC, he worked on integration of optoelectronic technologies for chip-to-chip optical interconnections and parallel pipeline processing systems. He has been with the Microelectronics Division of Sun Microsystems Inc., Palo Alto, CA, as a Member of the Technical Staff since September 1998. His research interests include optoelectronics VLSI for parallel computing and networking, microprocessor architecture, and scalable cluster computing. He is a member of the Optical Society of America.