Application of Networking Concepts to Optoelectronic Multiprocessor Architectures

James Rorie[1], Philippe Marchand[2], Jeremy Ekman[1], Fouad Kiamilev[1], Sadik Esener[2]
[1]*Department of Electrical-Computer Engineering, UNC-Charlotte, Charlotte, NC 28223*
[2]*Department of Electrical-Computer Engineering, UC San Diego, La Jolla, CA 92093*

Abstract

This paper describes an approach to creating high-speed multiprocessor architectures involving free-space optical interconnects using knowledge gained from the analysis of traditional network topologies. Development of an architectural specification through identification of the components of a functional description and mapping them to an appropriate network model is presented. By applying these techniques, the designer has the advantage of field tested solutions to the problems that occur in the development of complex, hybrid, hierarchal, optoelectronic multiprocessor architectures and protocols.

1. Introduction

Developing systems involving opto-electronic components can present a number of challenges to the hardware designer. There are a multitude of issues that must be addressed when exchanging data between low speed electrical buses and the higher speed optical environment. These hybrid architectures are further complicated when hierarchal design strategies are applied.

In an attempt to reduce optoelectronic multiprocessor design complexities, concepts from related fields have been applied. Among these fields, network architecture bears the greatest similarity. Many networking concepts have been applied successfully in the area of parallel processing[1] and optoelectronic systems[2][3]. Hence, we have adopted a network architecture model for the design of optoelectronic multiprocessor systems.

1.1 Impetus

The device technologies used in the optical area exist mainly in small production runs; therefore, their price is considerable relative to other technologies. To maximize capital resources during the development cycle, an attempt must be made to reduce the number of prototypes required. This requires extensive simulation of designs that are developed by using sound operational concepts.

In order to accomplish these goals within the time constraints set for a project, rapid prototyping techniques also need to be applied. But these techniques only speed the actual implementation of the design. Other techniques need to be utilized to quickly develop the architectural specifications.

In this paper, we present a technique for using network models as guidelines for developing high-speed, optoelectronic architectures. Previous work in optoelectronic multiprocessor systems has focused on using optical interconnects to implement the interconnection network [10][11][12] and evaluation of proposed optoelectronic multiprocessor architectures[13]. By selecting an example that possesses strong functional relationships, problems that occur in development can be addressed through simple analysis. The result is a structured approach to optoelectronic multiprocessor design.

2. Mapping Functional Specification to Network Topology

The first step to in applying this approach is to find an appropriate real world model for the proposed system and map these concepts to an architectural specification. The critical decision involved in applying the network topological concepts is the selection of an appropriate model. This is accomplished through a careful examination of the functional specification.

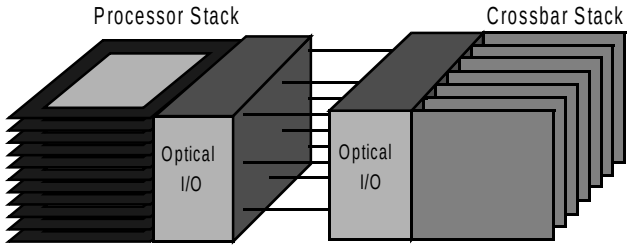
2.1 The 3D-OESP Architecture

The 3D Opto-Electronic Stacked Processor project consists of the development of a custom system architecture based around vertical chip stacking technology. This technology combines individual VLSI die in a three-dimensional fashion to form a cubic device. These stacks are then used with optical interconnects to create a high-speed, high-density computational platform[15].

The architecture needs to support the calculation of two dimensional FFT's and other digital signal processing functions using VCSEL(Vertical Cavity Surface Emitting Laser) based free-space optical interconnects for communication between planes (die) and stacks[16]. Major architectural points are the development of reliable high speed serial link and an inter-processor communication protocol. This protocol must account for the unique requirements of the free-space optical interconnects while maximizing the available bandwidth[5]. One such requirement is the need to incorporate many simultaneous

serial communication links in order to avoid the problems of synchronization and skew across bits in a parallel word.

Figure 1. Block Diagram for the 3D-OESP System

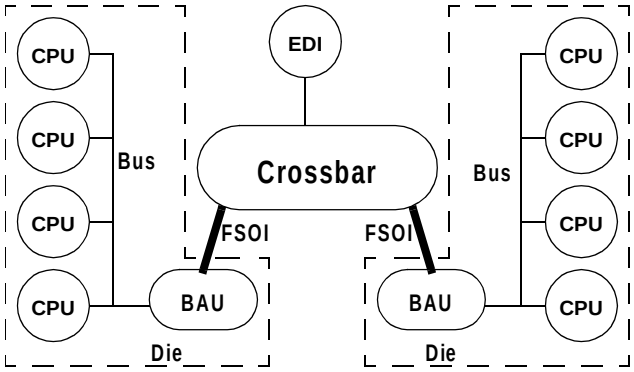


Inter-stack communication will be accomplished through the design of a robust full-duplex optical link centered around two-dimensional VCSEL-detector arrays[14]. This link should be self sufficient, containing all necessary logic required for operation including electrical/optical domain transfer circuitry[4].

Figure 1 shows a block diagram of the proposed system. The three major functional blocks are the processor stack, opto-electronic arrays and the crossbar stack. This particular configuration consists of 16 dies in each of the two stacks. Computation is performed in the processor stack, with each plane having four general purpose CPU's. Communication between dies is accomplished through the opto-electronic arrays. Data is transmitted to the crossbar stack, which is rotated 90 degrees relative to the processor stack. This orientation allows the data to be routed to a different part of the optoelectronic array for retransmission to the processors.

Figure 2 shows a simplified block diagram of the 3D-OESP project implementing a pair of processor and crossbar dies. In the architecture, the CPU's are the source and destinations for all packet traffic. The Bus Arbitration Units(BAU's) provide byte level synchronization and allocate bus access between CPU's. The connection between the CPU's and BAU's is a local tri-state parallel data bus. To provide global communications, a queuing crossbar is included to determine the destination BAU for incoming packets and retransmit to that location. BAU-to-crossbar interconnection is provided by a matrix of free-space optical interconnects(FSOI). Finally, the External Data Interface(EDI) provides an electrical connection for transmission of data to and from the 3D-OESP system.

Figure 2. Simplified 3D-OESP Topology

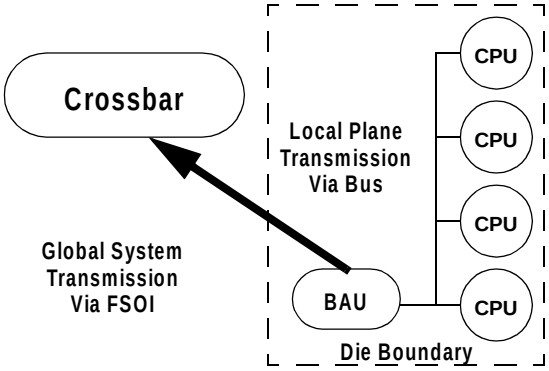


2.2 Functional Inferences

Using the 3D-OESP specifications, a few simple comparisons can be made. The CPU's function as endpoints for data generated on a tri-state bus. Thus they would correspond to network nodes transmitting on a broadcast bus. Data can be transmitted locally to the plane or to CPU's that reside on different planes within the system.

Transmission to a different plane would require that the data be forwarded to a higher level component that has knowledge of the physical relationships between the planes. The forwarding concept is used for transmitting between subnets in a network environment. Since subnets are local in nature, this concept could be applied stating the boundaries for a subnet and transmissions within a plane. Forwarding of packets is provided through the BAU, which is a function traditionally performed by a network router.

Figure 3. Applying Subnet Concepts to the Plane



Global data in the system is transmitted by the free-space optical link to a central crossbar. The optical link has a much higher bandwidth capacity due to a higher clock rate than the local parallel buses and is functionally different in its

operation. It obviously requires some type of data conversion and multiplexing to make maximum use of its capacity. In large networks, high-speed, global fiber connections, termed “backbones” are used to handle the large amount of traffic between subnets. Since we have established that each of the planes roughly corresponds to a subnet, it follows that the global connection between these could be represented by a optical fiber backbone.

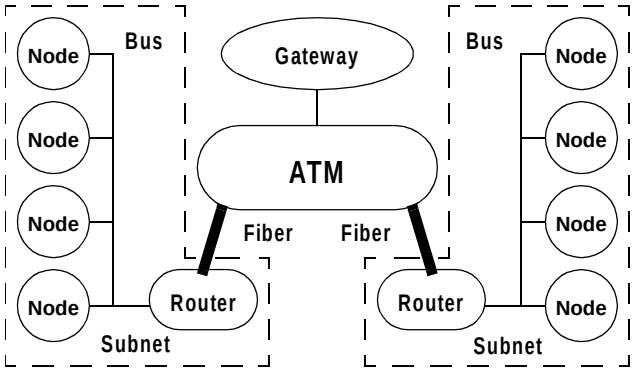
The job of routing global information in the 3D-OESP system is performed by a central crossbar. This device receives information from each of the planes, determines its destination plane and retransmits it on the appropriate optical channel. To perform this task, it must have knowledge of the locations of each of the planes and their associated address ranges. Based on this information, an appropriate model would be an Asynchronous Transfer Mode switch. This device is designed to take network data from multiple input channels, and reroute it to destination ports based on the encapsulated address information.

The final portion of the 3D-OESP architecture to be examined is the External Data Interface (EDI). This is a communications interface that provides a link to systems that are not directly part of the system architecture. Its main tasks are to provide the initialization data for processing and to download the resulting data after computation.

Assuming that we have successfully created a network model for the 3D-OESP architecture as a whole, it would follow that the EDI could be considered a network gateway. This gateway would provide access to the systems external to the basic architecture.

Figure 4 shows an equivalent network topology based on the initial analysis, substituting an appropriate network component for each functional block in the architecture specification. In the model, the CPU’s are represented as traditional network nodes. Each of the nodes is connected on a common bus to form a subnet. Subnets are connected globally through high-speed fiber channels to a central ATM switch. To provide local traffic isolation, each subnet is provided with a router to discriminate between packets based on their destination address. A gateway is provided for external access to the network.

Figure 4. Equivalent Network Model



3. Application of Topological Concepts

Once an appropriated model is selected, the specifications for the architecture are developed. As decisions concerning functional trade-offs arise, the model is examined to see how it implements a solution. By mapping this to the architectural specification, solutions will become apparent for many problems.

3.1 Data Transmission Methodology

The first decision to be examined under the network model is that of a method of transmission throughout the system. Since transmissions on a broadcast network require a destination address, some form of binding is necessary to associate the address information with its data. Networks accomplish this through the use of a structured transmission packet. This packet format specifies the relationship of the address information to the data being transmitted.

Next, a transmission protocol must be determined. This can be provided through the establishment of a virtual circuit or by connectionless packet transmission. Virtual circuits have overhead relating to the creation and destruction of the communication path. They are ideal for extended unicast transmissions by the same source and destination. However, for short bursty traffic, set-up and tear down overhead can be significant.

Connectionless transmissions require the address overhead on each packet and are therefore, inefficient for extended data transmissions. However, they do not incur the initialization overhead of the virtual circuit.

The applications targeted for the 3D-OESP project are of a distributed processing nature. The means that data will be

ordered, yet bursty, similar to ethernet traffic. The model in this case dictates a connectionless, packet oriented approach.

Figure 5. Packet Transmission Format

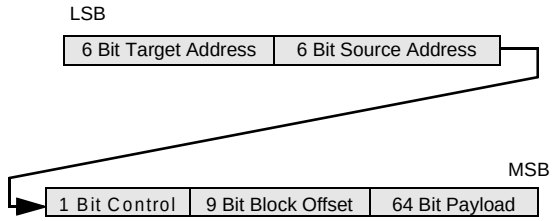


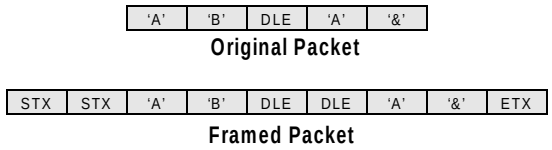
Figure 5 shows a layout of the packet format used in the 3D-OESP specification. The packet transmits a 64-bit payload. With the number of CPU's in the full specification, 8-bits of address are needed to fully identify a destination. Along with the destination, the source address, offset and mode bit are also specified. This information is mainly used by the destination CPU's and could therefore be considered to be data. By specifying it in the system level packet, this information would be available for future expansion at a later time.

3.2 Packet Framing

Once a packet format has been established, the problem of framing becomes paramount. Data that will be transmitted over a high-speed serial channel must provide some method of explicitly specifying the beginning and end of each of the packets. This framing information should be sufficient to be positively identified in the presence of errors, while not providing significant overhead to the packet.

Packet oriented transmission is classified as synchronous in nature and provides for two approaches to providing frame synchronization. Character synchronization specifies data boundaries through the embedding of STX-ETX (Start Transmission, End Transmission) characters in the data stream to identify the boundaries of the data. To provide data transparency, STX-ETX characters that appear the in the data packet are escaped with a DLE (Data Link Escape) character to differentiate them from the packet boundaries.

Figure 6. Character Synchronization



This concept can also be applied at a lower level through the use of bit synchronization. A technique known as bit stuffing uses a consecutive string of 1's or 0's as a frame boundary sequence. The length can be sized for greatest transmission efficiency as the packet is considered a bit stream and character boundaries are ignored. If the data stream contains a sequence of 1's or 0's that is equal in length to the frame sequence, a 0 or 1 is inserted to break the run. In this way data transparency is achieved.

Figure 7. Bit Synchronization

1100101011000111110111111
Original Packet

01111101100101011000111101011110101111110
Framed Packet

Either approach is acceptable in the network model that we have created; therefore, the decision should be based on the factors relevant to the architecture. Since the actual nature of the data will vary from each application, some flexibility will be desired. Because of this, the bit stuffing scheme is preferred since it handles the transmission of binary data or printable characters with similar efficiency.

3.3 Arbitration and Collision Avoidance

Since we have several processors that have need of the local bus, some type of access protocol is necessary to prevent collisions. This protocol must be robust, easy to implement and minimal in resource costs. The three main types of network access protocols are Token Ring, Carrier Sense Multiple Access with Collision Detection (CSMA/CD) and Token Bus.

Using a token ring approach, a token would be transmitted to each CPU. Once received, that CPU would have access to the bus. Data is transmitted sequentially from each CPU in the ring until it is forwarded or reaches its ultimate destination. Token ring provides the best utilization of the bus, but it would require considerable VLSI real estate for the routing of the ring bus.

CSMA/CD would allow each CPU to transmit on the bus as necessary. The CPU would then listen to its transmission to see if another CPU transmitted simultaneously, destroying its data. If a collision was detected, each CPU would wait a random length of time and retransmit. CSMA/CA is easily implemented through the use of an open-collector interface for each of the CPU's. However, it only allows up to 20%-40% of maximum bandwidth before collisions become a problem. Since the parallel bus is the probable bottleneck in the design, this would be inadvisable[7].

Token bus provides the high speed of the token ring with the routing simplicity of a broadcast protocol. It transmits the

token to each of the CPU's and data appears on the central bus. Token bus also provides excellent throughput, achieving 80% bandwidth utilization of a communications channel[8].

For the given architecture, token bus appears to hold the best performance. However, a full implementation is not required because of the close proximity of the CPU's. The concept can be distilled to its basic concepts, a broadcast based protocol with a central arbitration authority. In the modified protocol, the BAU would serve as a bus arbiter. Through control lines, it would signal to each of the processors when they have access to the bus.

3.4 Optical Backbone

In the network model, a traditional broadcast connection is implemented for all local communications within the subnet. Global communications are accomplished over a high-speed fiber backbone. While not capable of transmitting the aggregate of all subnets, it does provide a higher capacity channel. This channel is capable of processing the additional addressing overhead that is required.

A single plane would use a fraction of the FSOI throughput. To take full advantage of the higher bandwidth of the optical channels, multiplexing is used to combine these low speed data paths. Time domain multiplexing over multiple channels is well suited for free-space optical interconnects due to their digital nature, high switching speed and relative immunity to channel crosstalk

Since addressing information is encapsulated into the global packets when they pass through the local router, the CPU's must know something of the topology of the network. This requires a processor that is larger and more application specific. The result is that a general purpose CPU would need to be modified to work in this architecture.

3.5 Routing

To achieve the lowest transmission latency possible, it is desirable to have all network nodes on a common broadcast bus.[9] However, this can drastically reduce performance in hierarchical topologies. In these systems it becomes necessary to filter irrelevant data, preventing it from being needlessly retransmitted.

To achieve this selective retransmission, the components within the system must have some knowledge of the address ranges for each of the transmission channels. In a network environment, routers perform this function, reducing overall traffic by intelligent filtering of packets. Transmissions with local addresses are destined for nodes within the local subnet and are prevented from joining global backbone traffic.

In the network model, the ATM knows of the existence of each of the subnets, but nothing of the addressing scheme within them. Therefore data is routed to the subnet with the appropriate address range. Since each subnet is broadcast based, the data will appear on the bus allowing the appropriate CPU to recognize its respective data.

Applying this to the system architecture, the crossbar would have knowledge of the address ranges for each of the planes. Data that flows through it would be sent to the appropriate optical channel. But some mechanism must exist to prevent local data from being transmitted to the crossbar.

The BAU performs this filtering function locally. If a packet appears on the bus that has an address that does not correspond to the allowed range within that plane, it will be forwarded to the crossbar for redirection. However, when a packet is transmitted that has a local destination, it appears only within the plane.

3.6 Node Address Assignment

One of the key problems in the 3D-OESP architecture was the determination of how CPU's addresses were assigned. Two approaches are used in a network environment, hardwired addressing and software update.

Hardwired addresses have the advantage of simplicity. Network addresses are assigned by manually configuring the adapter. This is a good approach for a static network topology, but in order to implement a dynamic architecture, a more robust approach is required.

Software-updated addresses allow a node to obtain its network address during some initialization phase. The address is received from some central network authority after an initial query. By allowing a system to start in an unknown state, local configuration is not necessary.

Either approach is supported by the model. The software approach is preferable since it allows each processor to be general purpose. However, software addresses are far more complex to create. Ethernet networks use a central authority to assign node addresses during adapter initialization using the BOOTP protocol. Thus the configuration, within certain limitations, doesn't require hardware changes.

Using the centralized approach, an address server of some description must exist on the network. This is accomplished in a number of levels. First, the central authority for the subnets would obviously be the ATM switch. Due to its location, it would be able to provide high order addressing bits, roughly analogous to a subnet mask. The second level of authority must reside within the subnet. This task is performed by the BAU. With its proximity within the subnet, it can assign the final bits of addressing to each of the CPU's.

From this, it follows that the initialization process would be two step. In the first step, the crossbar transmits an information packet to each the of the BAU's, informing them of their respective high order addresses. The second step

would involve the BAU's transmitting the complete address to each of the processors on the local bus. At the completion of this step, each of the processors is initialized relative to its location in the bus and is capable of sending and receiving data.

3.7 Multicasting

One of the functions that would be required during start-up of the 3D-OESP system is the broadcast of initialization data from the EDI to all the CPU's. Data that is transferred will be identical, differing only in address. The simplest approach is to create packet with a destination for each of the CPU's but this uses a large amount of bandwidth, on the order of:

$$E = N*M*P \qquad \text{(eq. 1)}$$

where,

E = Bytes transferred through system

N = Number of planes

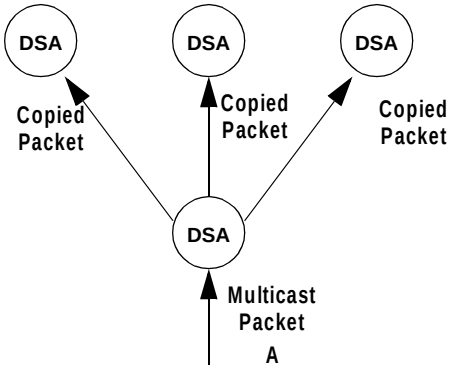
M = Number of CPU's

P = Size of packet in bytes.

Given that this operation could be performed numerous times during a computational cycle, it would be desirable to reduce this overhead.

A technique employed by networks to deals with packets of this nature is known as multicasting. This allows a packet to be transmitted to a group address. The packets are received by Directory Service Agents (DSA's), which attempt to determine the destination for each of the packets. If the local DSA doesn't have enough information to determine an address, it forwards a copy of the packet to all DSA's that could provide more information.

Figure 8. Multicasting Data Flow



Using a multicast packet in the network model, we follow the above example. As the packet is transmitted from the EDI, it first encounters the ATM switch. Since this is a switching point, it would follow that it operates as the first level DSA. The ATM, being unable to completely decode the destination for the packet, would forward a copy of the packet to each subnet. These copies would be received by each router, which is also operating as a DSA. The local router would complete the address and place the data on the local bus for each node to receive.

In the architecture specification, the ATM and BAU would operate as the DSA for the system. Each would copy the packets and forward them to the next level in the broadcast tree. The resulting bandwidth usage for this transmission would be P at the EDI and N*P over the FSOI with no increase in processor complexity.

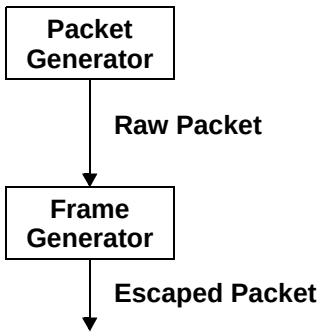
4. Simulation Results

Upon completion of the architectural specification, VHDL and C++ models were created of each of the major functional components within the system. The VHDL models were created under Synopsys, using both behavioral and structural code to describe the components. Simulations on this code were created to explore the feasibility of the 3D-OESP design. The simulations were performed at the Link and Network OSI layers.

4.1 Link Level Simulation

The lowest level simulation performed was created in C++ to examine the efficiency of the FSOI link for different packet payload lengths. Figure 9 shows a block diagram of the simulation model. The packet generator used the C++ rand() library routine to create a string of pseudorandom data in user specified lengths. Randomly generated addresses and control information where added to fully model the packet shown in Figure 5. The data was then passed to a bit stuffing algorithm that created the necessary escape sequences to properly frame the data. At this point, the finalized packet possessed the encoding that would occur in the FSOI serial datastream.

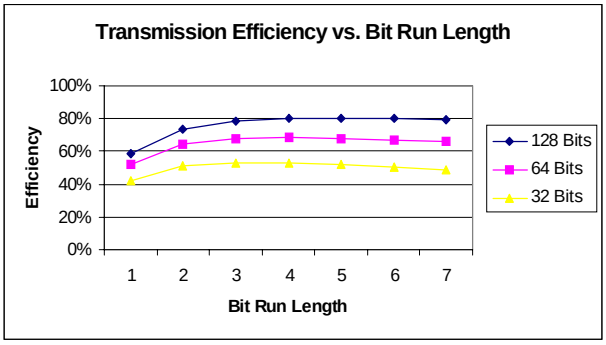
Figure 9. Framing Simulation Block Diagram



On completing the framing process, the encoded packet's size was compared to its original raw form. The result of the comparison was averaged over a large number of simulation cycles. The simulation was repeated with varying packet and bit run lengths.

Figure 10 shows the results of 50MB of data transmitted through the system. In this graph, a theoretical maximum of 100% efficiency would occur with an encoded frame packet equal in length to its original raw size (0% overhead). Simulations showed a trend of increased efficiency for larger packet sizes. Peak efficiency was shown at bit run lengths of 3 to 4, depending on packet size. From this result, it was determined that a 32-bit payload would result in an unacceptable level of throughput. 128 and 64 bit payloads showed far better results with 69%-80% framing efficiency. This translates to an effective channel rate of 690-800 mb/s assuming 1 Gb/s raw throughput and packet sizes of either 64 or 128 bits. The trade-off between 128 bit payloads relative to VLSI real estate has not been determined at this time and must be examined to fully weigh this decision.

Figure 10. Framing Efficiency



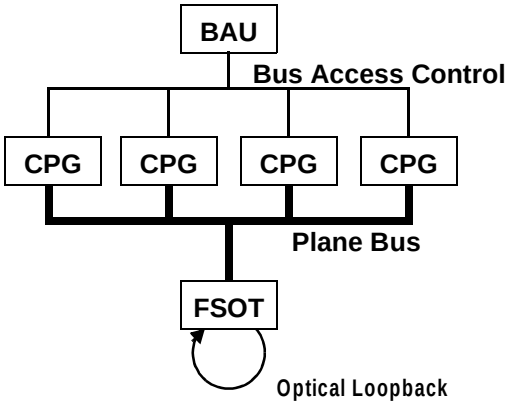
4.2 Network Level Flow

This next level of simulation was performed at the network level under VHDL. Its purpose was to examine the flow of traffic through the channels at the packet level. Of particular interest was the efficiency of the bus arbitration technique. This simulation utilized pseudo-random data created by four CPU packet generators(CPG), a BAU and one Free-Space Optical Transceiver(FSOT).

Packets were transmitted in a pseudorandom fashion through the use of a Linear Feedback Shift Register driving a counter. This created a staggered time interval for each source and allowed the CPG's to generate packets in a manner similar to random traffic. The BAU functioned to allow orderly access to the bus and to direct packets with off chip addresses to the FSOT.

The FSOT was minimal implementation, operating basically as a loopback connection. It received data from the parallel bus, converted it to serial format and transmitted back to itself. To resolve destinations for off chip transmissions, all addresses where stripped of their high order bits (modulo 4). When the packet was received, it would be routed to a CPG that corresponded to the modified address.

Figure 11. Network Simulation Model

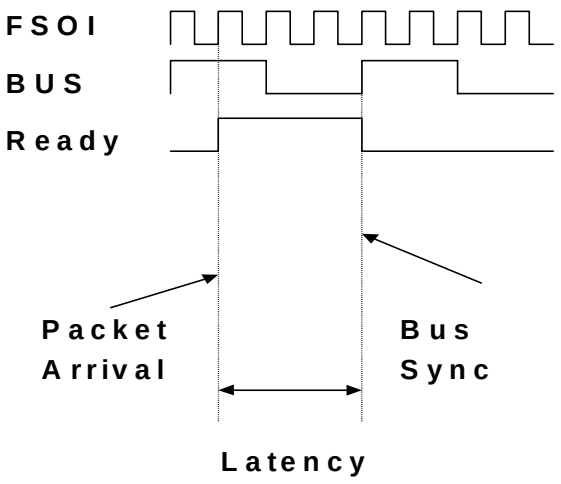


The simulation showed that the system performed with approximately 5% throughput loss due to clock synchronization latency. Delays associated with the bus arbitration protocol were not present due to the pipelined request technique used by the BAU.

Figure 12 shows an example of the clock latency within the system. To take advantage of its higher bandwidth, the free space optical channel operates at a higher clock frequency than the local bus. Data that travels through the optical channel arrives at the local bus relative to the FSOI clock. Since there is a clock differential, the bus clock can take up to

four FSOI clock cycles before it synchronizes and removes the data from the optical buffer.

Figure 12. Bus Synchronization Latency



5. Conclusions

The network topology mapping technique allows a very useful way to generate specifications for complex system architectures. By use of this technique, analysis of the architecture is facilitated through the examination of similar network topologies. The simulation results of the specifications obtained through the use of this technique supports such use.

6. Acknowledgments

This effort is sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory under agreement number F30602-97-2-0122. The US government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon.

7. References

[1] A. Mainwaring, S. Schleimer, “System Area Network Mapping”, 9th Annual ACM Symposium on Parallel Algorithms and Architectures, Newport, RI, June 22-25, 1997

[2] B. Anglis, H. Hinton, “A Dynamically reconfigurable Token-based Optical Backplane”, IEEE/LOES Summer Topicals: Smart Pixels, July 1998.

[3] J. Wu, C. Kuznia, C. Chen, B. Hoanca, A. Sawchuk, “Network with Free Space Optical Data Packet Using Carrier-Sense Multi-Access with Collision Detection (CSMA/CD) Protocol”, IEEE/LOES Summer Topicals: Smart Pixels, July 1998.

[4] J. Rorie, P. Marchand, P. Chandramani, J. Ekman, F. Kiamilev, F. Zane, V. Ozguz, S. Esener, “A System Architecture for use with Free Space Optical Interconnects in a 3D Stacked Processor Environment”, IEEE/LOES Summer Topicals: Smart Pixels, July 1998.

[5] S. Esener, P. Marchand, “3D Opto-electronic Stacked Processors: design and analysis” OC Computing '98, Bruges, Belgium, June 1998.

[6] F. Halsall, “Data Communications, Computer Networks and Open Systems”, Fourth Edition, Addison Wesley, Harlow, England, 1996

[7] J. Hammand, “Performance Analysis of Local Computer Networks”, Addison-Wesely, Reading, MA

[8] B. Stuck, “Calculating the Maximum Throughput Rate in Local Area Networks”, IEEE Computer, May, 1983

[9] S. Leong, B. Dewar, “The Effect of Traffic Locality on Network Architecture Performance”, Proceedings of the IASTED International Conference, Orlando, FL January 8-10, 1996

[10] T. Szymanski, H. S. Hinton, “Reconfigurable intelligent optical backplane for parallel computing and communications”, Applied Optics, March 10, 1996, pg 1253-1267

[11] A. Louri, S. Furlonge, C. Neocleous, “Experimental demonstration of the optical multi-mesh hypercube: scaleable interconnection network for multiprocessors and multicomputers”, Applied Optics, December 10, 1996, pg 6906-6919

[12] A. Louri, S. Furlonge, “Feasibility study of scalable optical interconnection network for massively parallel processing systems”, Applied Optics, March 10, 1996, pg 1296-1307

[13] T. Pinkston, U. Efron, M. Cambell, “Applying Optical Interconnects to the 3-D Computer: A Performance Evaluation”, Journal of Parallel and Distributed Computing, October, 1994

[14] O.Sjölund, D. A. Louderback, E. R. Hegblom, J. Ko, and L. A. Coldren, “Individually optimized bottom-emitting vertical-cavity lasers and bottom-illuminated resonant photodetectors sharing the same epitaxial structure”, Optics in Computing, Bruges, Belgium, June 1998.

[15] Irvine Sensors Corporation, Costa Mesa, CA, 92626

[16] M. Hibbs-Brenner, Y. Liu, R.Morgan, J. Lehman, “VCSEL/MSM Detector Smart Pixel Arrays”, IEEE/LOES Summer Topicals: Smart Pixels, July 1998.