*n*D-RAPID: a multidimensional scalable fault-tolerant optoelectronic interconnection for high-performance computing systems

Chander Kochar, Avinash Kodi,* and Ahmed Louri

Electrical and Computer Engineering Department, University of Arizona, Tucson, Arizona, 85721, USA
**Corresponding author: avinashk@ece.arizona.edu*

Received October 2, 2006; revised December 20, 2006; accepted February 28, 2007; published April 23, 2007 (Doc. ID 75688)

The increasing demand for bandwidth coupled with saturating electrical systems is leading the drive for optics as an interconnect technology. High-performance computing systems (HPCS) consist of a large number of components such as processors, memories, and interconnection links. As the number of components in HPCS increases, the probability of a failure also increases. Therefore, it becomes imperative that the system be fault tolerant to ensure high availability even in the presence of faults. We propose a multidimensional optoelectronic architecture, nD-RAPID (reconfigurable and scalable all-photonic interconnect for distributed and parallel systems), where n can be 1, 2, or 3. nD-RAPID provides high bandwidth, low latency, dynamic reconfiguration, and fault tolerance. While designing the fault-tolerant routing algorithm, we have tried to ensure that it provides optimum performance in the absence of faults, shows minimal degradation in the presence of faults, and can tolerate a reasonable number of faults. In the presence of faults the onboard switching mechanism dynamically reconfigures itself to reroute packets along nonfaulty links. Extensive simulation results are presented that compare nD-RAPID with other popular HPCS topologies. © 2007 Optical Society of America

OCIS codes: 200.0200, 200.4650.

1. Introduction

Electrical interconnects for high-performance computing systems (HPCS) are fast approaching their saturation point, both in terms of allowable maximum bandwidth and latency. It is estimated that future HPCS utilizing off-the-shelf processors will be limited to approximately 20 Gbits/s [1–3]. However, future bandwidth predictions for HPCS are estimated to be in the range of many Tbits/s [4,5]. A technology that can provide high bandwidth, low power, and low latency for HPCS when compared with electrical interconnects is optics. Recently, studies from both the academia and industry have reported the advantages of short-range optical interconnects (including on-chip optical interconnects) in terms of bandwidth, power, and latency [4–7].

In this paper, we propose a novel optoelectronic architecture nD-RAPID (n Dimensional reconfigurable and scalable all-photonic interconnect for distributed and parallel systems), an extended version of a previously proposed architecture called RAPID [8]. RAPID is an all-optical architecture that provides high bandwidth, high connectivity, and low latency. RAPID was able to achieve these features by exploiting several optical technology features that enable: (1) ample communication bandwidth by aggressively utilizing wavelength-division-multiplexing (WDM) and space-division-multiplexing (SDM) techniques and combining them to form a multiple WDM (M-WDM) technique, (2) high connectivity thereby reducing the network diameter resulting in lower queueing-routing delays for packet transmission, and (3) scalable bandwidth and cost that grows linearly with the number of nodes while providing low latency. Although RAPID is an all-optical architecture, nD-RAPID is optoelectronic and is regulated by electrical flow control mechanisms.

As bandwidth demands increase, HPCS have become complex and consist of hundreds and thousands of components such as processors, memories, and interconnection links. For example, the IBM BlueGene/L, which is ranked at the top of top500.org

[9], has doubled in size and now has over 100,000 processors. As the size of HPCS increases, the chance of a failure also increases. For example, in a 1024-node 3D torus network with a bit error rate (BER) of 10^{-15} an error can occur every 16 s [10]. To maintain high availability, it therefore becomes inevitable that HPCS be built such that they degrade gracefully in the presence of all kinds of faults. Among the different types of faults that can occur in a system, faults in the interconnect are of great importance. This is because such faults can isolate a part of the system that might otherwise contain healthy nodes and components. Not only should the interconnect provide high bandwidth and low latency, but it should also be fault tolerant [11].

Some of the common methods used for tolerating faults include component redundancy, reconfiguration, and fault-tolerant routing algorithms. While component redundancy is an effective means to tolerate faults in components such as power supplies and cooling fans [10], it may not be a very efficient means in terms of faults in components such as processors, interconnection links, transmitters, and receivers. This is because of the high cost of the spare components and the possibility that the circuits required to switch to the spare components might fail. The second method involves reconfiguring the routing tables and adapting them to the new topology after the failure [12]. This method can be effective in switch-based networks such as Infiniband [13] or Quadrics [14], where the user defines the topology. Although, when using reconfiguration in such a manner, any number of faults can be tolerated as long as the network remains connected, it has been shown in [15] that such generic algorithms achieve poor performance when applied to regular networks. The third common way to tolerate faults is by designing a fault-tolerant routing algorithm. The design of the router and on-board switching methodology plays a very important role in ensuring an efficient fault-tolerant routing algorithm.

Thus, in the proposed nD-RAPID architecture, we not only encapsulate the key features of RAPID, but also incorporate fault tolerance and dynamic reconfigurability into RAPID to improve its performance. We achieve a fault-tolerant architecture by expanding RAPID along multiple dimensions ($n=2,3$). This ensures that there are now multiple paths to any given board. It also enables us to increase the scalability. In addition to the above features, we have developed a fault-tolerant algorithm for nD-RAPID. The algorithm enables dynamic reconfigurability by rerouting packets when they come across faulty links. The fault tolerant routing algorithm is designed such that it maintains optimum performance in the absence of faults, shows minimal performance loss in the presence of faults, and can tolerate a reasonable number of faults.

Though fault tolerance in nD-RAPID can be achieved from many perspectives such as interconnect, transmitters, receivers, and router, in this paper, we look at fault tolerance from an interconnect point of view. In the absence of faults, we use $x-y-z$ dimension order routing. In the presence of faults, the proposed algorithm reroutes packets along an alternate route in order to avoid faulty links.

Fault tolerance in the electrical domain is well studied, especially for interconnects such as mesh, torus, and hypercube, which are commonly used in HPCS [16–20].

A few fault-tolerant architectures for HPCS have also been introduced in the optical domain. Optical multimesh hypercube (OMMH) [21] and spanning multichannel linked hypercube (SMLH) [22] are examples of two such systems. In references [23,24] are two optoelectronic networks for HPCS. While the above four networks could support reliable data transfer, no fault-tolerant algorithm was provided. In [25], the authors provide a fault-tolerant algorithm for a hypercubelike structure known as hypercube connected ring network (HCRNet). The authors in [26] propose two schemes to tolerate faults in a WDM system using a star coupler and tunable lasers and receivers. Reference [27] gives a fault-tolerant algorithm for Clos networks. The authors in [28] describe two methods to tolerate a single optical switch failure and single optical link failure in an optical torus network.

In this paper, an attempt has been made to rigorously analyze the effects of faults in a switchless optical network for HPCS by simulating the architecture in a faulty scenario. Most other approaches available in literature have either analyzed the affects of faults only through theoretical data [25,26,28] or, if simulation results are available, they are for switched optical networks [27].

2. Architecture Details: *n*D-RAPID

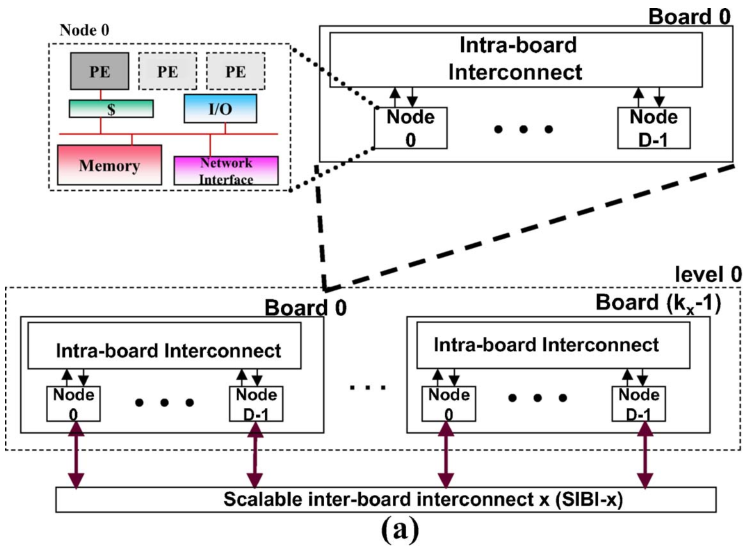
nD-RAPID can be described in a manner analogous to a k-ary n-cube network where n is the dimension and k is the number of nodes per dimension. Given the flexibility of the system, we use a mixed radix approach to describe nD-RAPID. Let D be the total number of nodes on a board. Then a given node in nD-RAPID is given by $R(k_n,d)$, where $n=1,2,3,\ldots$ is the dimension of nD-RAPID and $0\leqslant d\leqslant D-1$. For example, if $n=1$, we have a 1D-RAPID system (k_x-ary n cube) where a node is represented by $R(x,d)$. x gives the board number and d gives the node number. When $n=2$, i.e., for 2D-RAPID (k_y,k_x-ary n cube) a node is represented as $R(y,x,d)$ where (y,x) give the board coordinates and d gives the node number. Similarly for a 3D-RAPID network, $n=3$ (k_z,k_y,k_x-ary n cube) and a node is represented by $R(z,y,x,d)$ where (z,y,x) represent the coordinates of the board. Thus the total number of nodes in the system $N_{\text{total}}=\Pi k_n \times D$.

Figure 1 shows conceptually the 1D-RAPID architecture. Figure 1(a) shows components of a node such as the processor, cache, memory, network interface, and I/O. It then shows how nodes 0 to $D-1$ are connected using an intraboard interconnect to form a board. The intraboard interconnect design is discussed in the next section. It further shows how boards 0 to k_x-1 are connected to form 1D-RAPID. Boards 0 to k_x-1 are connected to each other via scalable interboard interconnect (SIBI-x). Figure 1(b) shows a very simplistic view of 1D-RAPID. Though it seems as if all the boards are connected to one another using a bus, the interconnect is actually point to point.

Figure 2 depicts the conceptual layout of 2D-RAPID. In Fig. 2(a), we see how 2D-RAPID is built from 1D-RAPID. A single 1D-RAPID system is duplicated k_y number of times. They are then connected along the y direction. All boards with the same x coordinate are connected to one another by means of a scalable interboard interconnect (SIBI-y). For example, the board 0 of each level are connected to SIBI-y for $y=0$. Figure 2(b) represents 2D-RAPID in a simplistic manner.

Figure 3 shows the conceptual diagram of nD-RAPID for $n=2$. Although the system looks like a 2D ring structure, it should be noted that it is actually a point-to-point network in the respective dimensions.

Figure 4 shows the conceptual diagram of nD-RAPID for $n=3$. A single 2D-RAPID structure is taken and duplicated k_z number of times. The boards are then connected



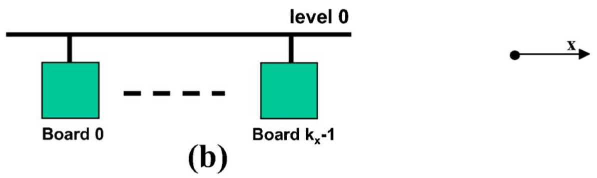


Fig. 1. Schematic of 1D-RAPID architecture along the x direction: (a) shows the internal structure of a node, how nodes are connected to form a board, and how 1D-RAPID is formed from a group of boards; (b) shows the stick diagram.

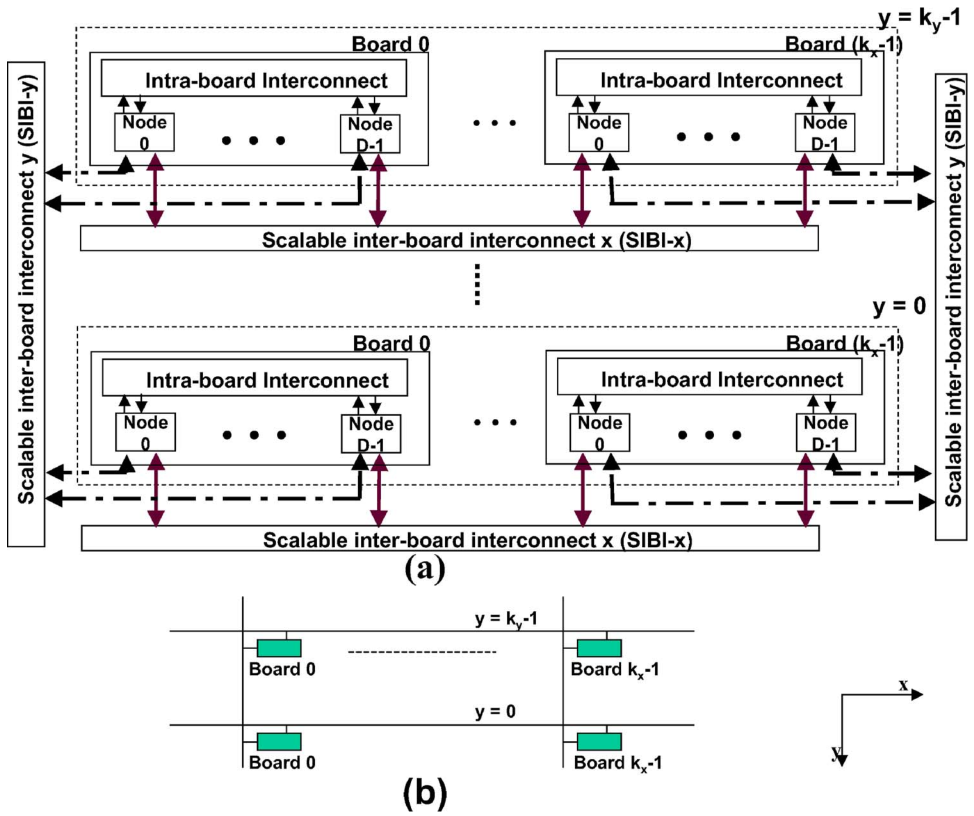


Fig. 2. Schematic of 2D-RAPID architecture: (a) shows how a 1D-RAPID structure is duplicated k_y number of times to form 2D-RAPID; (b) stick diagram representation.

to one another along the z direction using a scalable interboard interconnect (SIBI-z). Two boards are directly connected to one another if exactly two of the three coordinates, i.e., x, y, and z are equal. The point-to-point ring structure has been replaced by straight cylindrical lines. This gives a good understanding of the 3D-RAPID structure without cluttering the figure with too many details. Figure 3(b) depicts a simplistic view of 3D-RAPID.

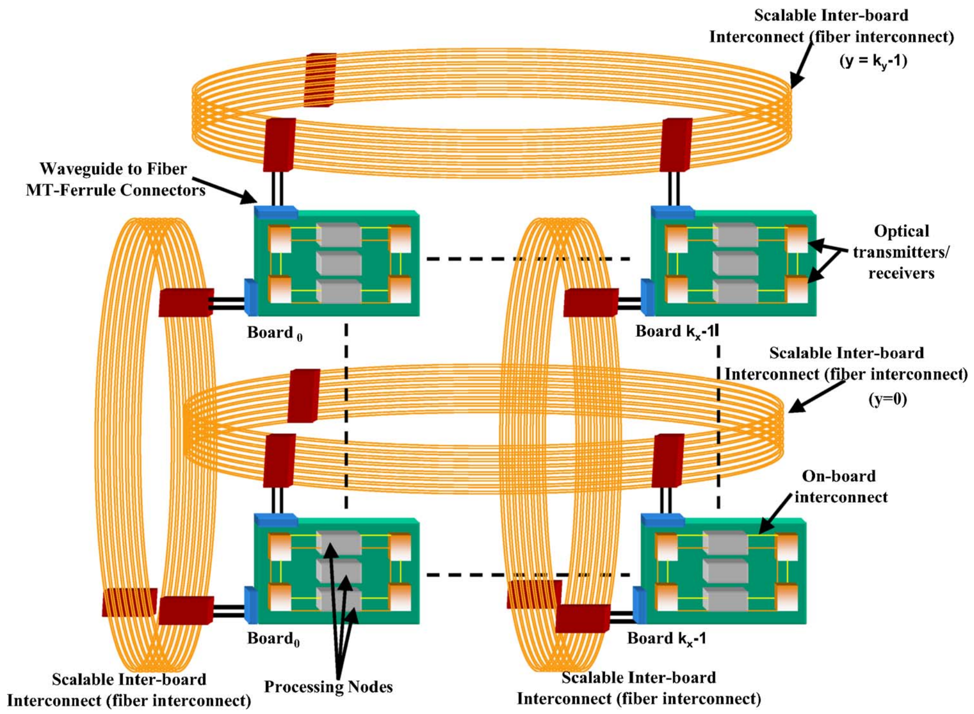

Scalable Inter-board Interconnect (fiber interconnect) (y = k_y-1)
Waveguide to Fiber MT-Ferrule Connectors
Optical transmitters/ receivers
Board $_0$
Board k_x-1
Scalable Inter-board Interconnect (fiber interconnect) (y=0)
On-board interconnect
Board$_0$
Board k_x-1
Scalable Inter-board Interconnect (fiber interconnect)
Processing Nodes
Scalable Inter-board Interconnect (fiber interconnect)

Fig. 3. Conceptual diagram of nD-RAPID for $n=2$.

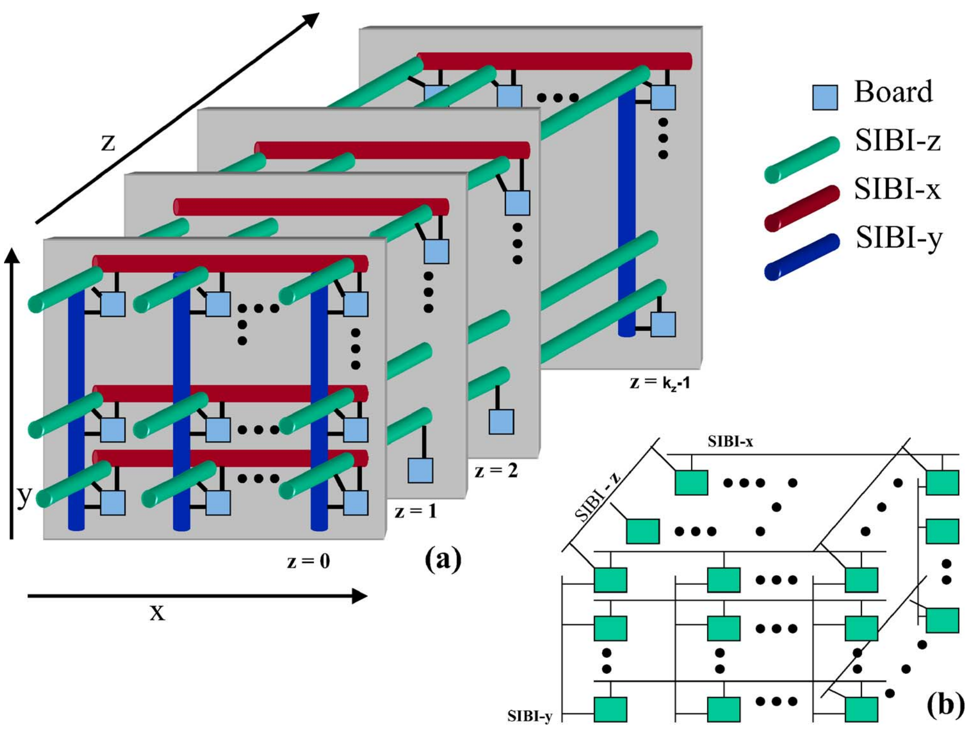


Fig. 4. Conceptual diagram of nD-RAPID for $n=3$. (a) shows how 2D-RAPID is duplicated k_z number of times to form 3D-RAPID. (b) Stick diagram representation.

In nD-RAPID, intraboard communication is implemented in the electronic domain whereas interboard communication is implemented in the optical domain. Apart from increasing fault tolerance, another advantage gained from scaling along different dimensions is that we are able to reuse the same set of wavelengths along each dimension. The intraboard switch design and the interboard communication setup are explained in the following sections.

2.A. *n*D-RAPID: Intraboard Switch Design

The intraboard layout in nD-RAPID is shown in Fig. 5. The network interface at each node consists of send and receive ports, which are then connected to optical transmitter and receiver ports by means of a bidirectional crossbar. A separate set of transmitters and receivers exists for the x, y, and z directions enabling the same set of wavelengths to be used in each direction. Although a crossbar has been used for intraboard interconnect, other interconnection networks may be used. When a node needs to communicate, it uses the send port to transmit over the crossbar to either receive ports for intraboard communication or to the optical transmitters for interboard communication. Similarly an optical receiver after receiving a packet can either send it to a receive port if the packet is at the destination board or send it to an optical transmitter for further interboard transmission.

Each packet that arrives at the input buffers goes through several router pipeline stages before it is delivered to the appropriate output port. As shown in Fig. 5, each input buffer is divided into several virtual channels to prevent deadlock and increase the throughput. A packet that enters the input buffer goes through four stages: route computation, virtual channel allocation, switch allocation, and switch traversal [10].

The router is designed to carry out both baseline routing in the absence of faults and the fault-tolerant routing algorithm in the presence of faults. The routing algorithms will be discussed in detail in later sections.

Table 1 shows how we can scale to large system sizes by reusing the same set of wavelengths in different dimensions as opposed to increasing the number of wavelengths used. We assumed that there are four nodes per board. The diagonal elements show how by reusing the same set of wavelengths along different dimensions we can create a system that can scale from 16 to 256 nodes. Also worth noting is that for a 256 node system, while in 2D-RAPID, we require 14 lasers per board, in 3D-RAPID, we require only nine lasers per board. Thus not only are we able to make a more fault-

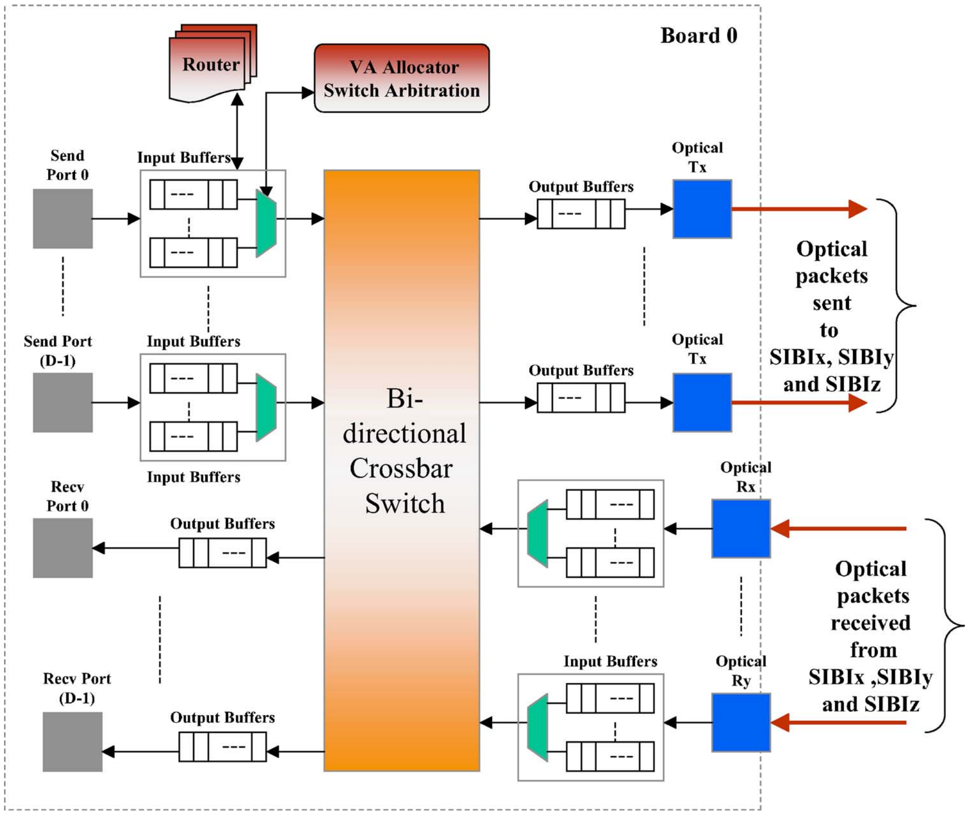


Fig. 5. On board interconnect. VA, virtual channel allocator. Each node consists of a send port and a receive port as shown on the left side of the crossbar. On the right side of the crossbar are optical transmitters and receivers.

tolerant system but also one that reduces costs in terms of lasers and receivers. The drawback, though, is that we reduce the aggregate system bandwidth.

2.B. *n*D-RAPID: Interboard Communication

In this section, we briefly discuss the interboard routing mechanism for RAPID, and how it has been extended for nD-RAPID. A more detailed report on the wavelength and routing allocation can be found in [8]. To explain how interboard communication works, we first take communication in one particular direction, say the x direction. Figure 6 shows a nD-RAPID system with $n=1$ and $k_x=4$. Each board has a fiber associated with it, known as the home channel for that board. Different wavelengths from different boards are merged into it to provide high connectivity.

The wavelength assigned from source board s to destination board d is given by $\lambda^{(s)}_{k_x-(d-s)}$ if $d>s$ and $\lambda^{(s)}_{(s-d)}$ if $s>d$. The superscript indicates the source board and the subscript indicates the wavelength to be transmitted on. For example, if any node on board 1 needs to communicate with any node on board 2, the wavelength to be used is $\lambda^{(1)}_3$, and for reverse communication the wavelength required is $\lambda^{(2)}_1$. A similar approach can be followed when communication is required between boards that need to communicate along the y or z direction, except in each case the wavelength used will be indexed upon the dimension value, i.e., k_y and k_z, respectively.

Table 1. Values Indicate the Number of Lasers Required Per Board to Create a Network of the Corresponding Node Size

	1D-RAPID	2D-RAPID	3D-RAPID
16 node	3	2	x
64 node	15	6	5
256 node	63	14	9

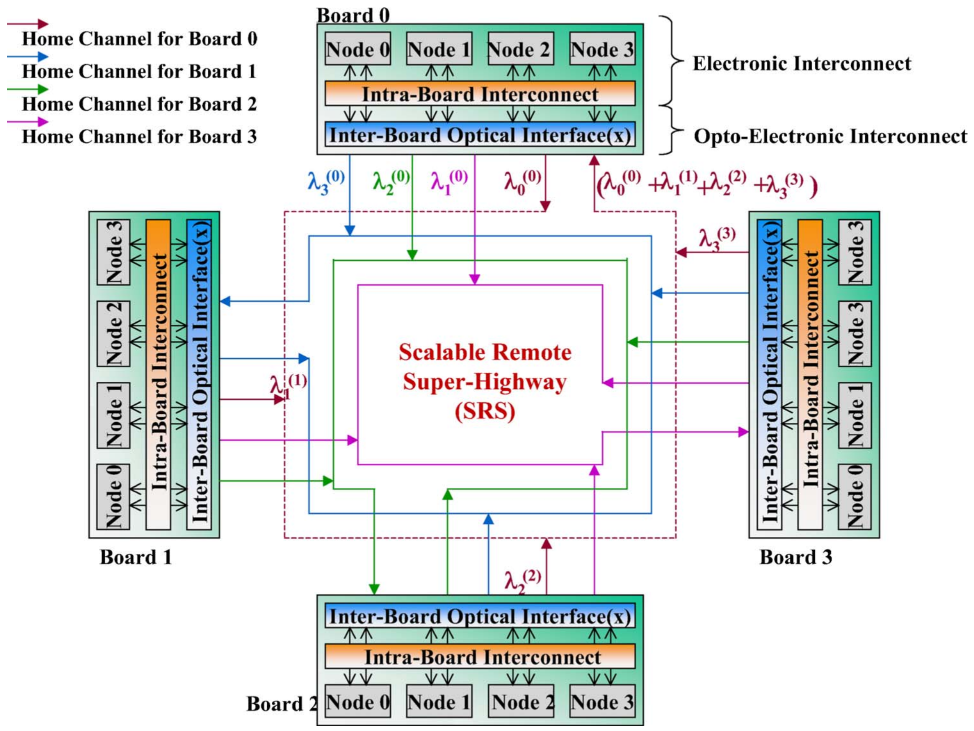


Fig. 6. Routing and wavelength assignment in nD-RAPID along the x dimension. The wavelength assigned from source board s to destination board d is given by $\lambda_{k_x-(d-s)}^{(s)}$ if $d>s$ and $\lambda_{(s-d)}^{(s)}$ if $s>d$.

3. Routing

This section is divided into three parts. The first part talks about the base routing algorithm, i.e., routing in the absence of faults. The second part deals briefly with the fault detection mechanism and the types of faults that the system can tolerate. The third part gives the fault-tolerant routing algorithm. It should be noted that each board can uniquely be identified with three parameters: z coordinate, y coordinate, and x coordinate. Let x_i be the x coordinate, let y_i be the y coordinate, and let z_i be the z coordinate of board i. Figure 7 shows a 2D-RAPID system with $n=2$, $k_x=4$, $k_y=4$. The solid circles represent nodes, whereas the solid rectangles represent transmitter receiver pairs: the horizontal rectangles for the x direction and the vertical rectangles for the y direction. Each board is provided with three sets of transmitter and receiver pairs per dimension because there are three other boards per dimension. The numbers in the parentheses represent the coordinates of the board (z_i,y_i,x_i). Thus a node $P(c,l,b,d)$ can be found on the board with z coordinates$=c$, y coordinate$=l$, x coordinate$=b$. All future references to $P(c,l,b,d)$ will be made using the notation $B(z,y,x)$ node d. This enables us to simplify our routing algorithm. For the sake of clarity, we only show the incoming and outgoing interboard connections along the x and y directions of $B(0,3,0)$. All future examples of the routing algorithm will be explained using Fig. 7.

In the routing algorithm, (z_s,y_s,x_s) indicates the coordinates of the source board, (z_d,y_d,x_d) indicates the coordinates of the destination board, and (z_p,y_p,x_p) indicates the coordinates of the board at which the packet is currently located. During the first iteration of the routing algorithm, the present board is the source board, and in the final iteration the present board is the destination board. x_{fault} denotes a board that cannot receive in the x direction, y_{fault} is used to denote a board that cannot receive in the y direction, and z_{fault} is used to denote a board that cannot receive in the z direction.

3.A. Base Routing Algorithm

The base routing algorithm is a simple four-step process that makes use of $x-y-z$ dimension order routing. In step 1, we check if the present board is the same as the destination board, and if the condition is satisfied then the on-board crossbar is used

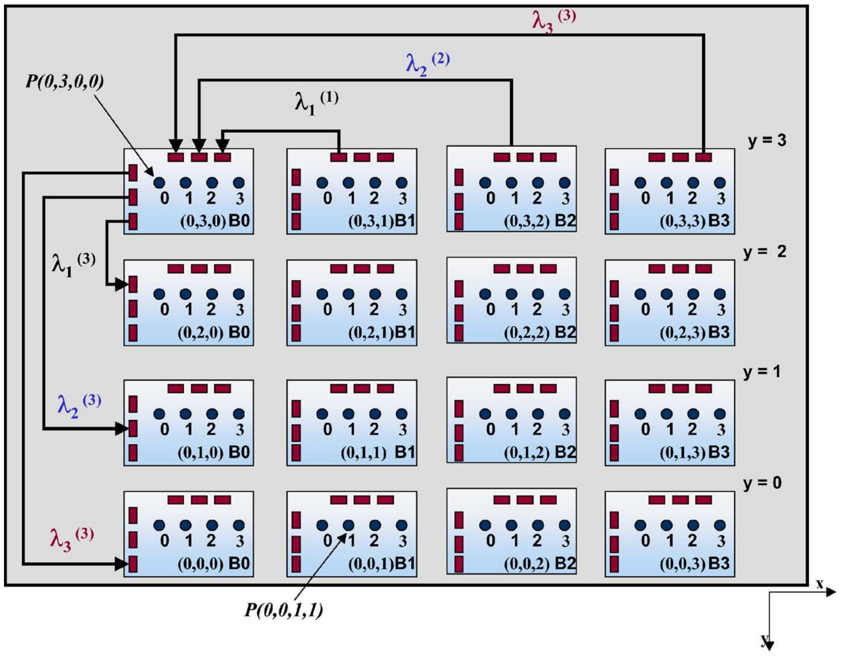


Fig. 7. Setup of nD-RAPID showing board coordinates as (z_i, y_i, x_i) for $n=2$.

for routing and the algorithm is terminated. If step 1 is not satisfied, we go to step 2, which checks if routing in the x direction is required. If step 2 holds, we route in the x direction. In case the boards are aligned in the x direction, we go to step 3, which checks for displacement along the y direction and routes accordingly. Similarly, if none of the above steps have been satisfied, we reach step 4, which checks if routing is required in the z direction and acts accordingly. This is repeated at each board until the destination is reached. The algorithm is summarized below.

Base Routing Algorithm

```
STEP 1: if ($x_p == x_d$) && ($y_p == y_d$) && ($z_p == z_d$)
            Do intraboard routing. Stop.
STEP 2: else if ($x_p != x_d$)
            Route along $x$ direction.
STEP 3: else if ($y_p != y_d$)
            Route along $y$ direction.
STEP 4: else if ($z_p != z_d$)
            Route along $z$ direction.
```

Consider the case when $B(0,3,0)$ node 0 would like to communicate with $B(0,0,1)$ node 1 (both nodes are shown in Fig. 7). At $B(0,3,0)$, we see that the present board and destination boards are not the same, so we move to step 2. Since $x_p != xd$, we route along the x direction to $B(0,3,1)$. At $B(0,3,1)$, we see that conditions of steps 1 and 2 do not hold, but it holds for step 3, i.e., $y_p != y_d$, so we route along the y direction to $B(0,0,1)$. Now, at $B(0,0,1)$ the condition of step 1 holds, thus we do intraboard routing and stop.

3.B. Faults

Although this paper primarily focuses on fault tolerance, in this paragraph, we briefly describe the fault detection scheme that is employed in nD-RAPID. Baseline RAPID uses a λ_0 wavelength for intraboard broadcast. Any message to be broadcast is sent into the home channel of a board using λ_0, and on demultiplexing the signal every node on the board receives it. In nD-RAPID, the optical intraboard interconnect has been replaced by an electrical one. Thus, we can now use the λ_0 transmitter and receiver pair on each board to form a monitor cycle. The λ_0 transmitter can send a low-power continuous signal or a fixed patterned pulse (known as a hello packet) at a given frequency using the home channel of the board. If the receiver for λ_0 does not receive the signal or if it does not receive the correct signal, it knows that a fault has occurred in the home channel of the board. It then communicates this to the transmitters via an error packet. The transmitters then send the error packet to all boards connected to the faulty board. The receivers on receiving the error packet immediately inform the routing unit of the switch and the transmitter that transmits to that board. After this fault detection mechanism, the fault-tolerant routing algorithm takes over if necessary. While the above method is used to detect a fault in the home channel and fiber, it is also possible that other channels in the fiber may not be working properly. To detect these faults, we make use of the one-is-one relationship between a transmitter and a receiver, i.e., a transmitter transmits to only one receiver and a receiver receives transmission from only one transmitter. Therefore, a fault in either the transmitter or the receiver implies that the channel needs to be shut down. The addition of a cyclic redundancy check (CRC) trailer during packet transmission and calculation of the checksum at the receiver can be used to detect errors in the packet transmission and reception. A problem in detecting a fault can occur when transmission restarts after a certain amount of idle period. If during the idle period some fault occurs, the receiver might keep waiting indefinitely for a packet to arrive, not knowing that a fault has occurred. To overcome this scenario, we suggest the use of hello packets. After a given idle period, the transmitter must transmit a hello packet. If the receiver does not receive the hello packet, it knows something is wrong and follows the same procedure as described above to notify all concerned.

As stated earlier, there is one fiber per board per dimension. Thus, if the fiber connected to a board in a given dimension breaks down, that particular board gets isolated from external traffic in the respective dimension. For example, in Fig. 6, if the fiber that brings traffic into board 0 from the x direction breaks down, then board 0 is isolated from other boards with respect to incoming traffic along the x direction. In other words, nodes on board 0 will be able to communicate with nodes on other boards, but the reverse is not possible. To reach a node on board 0, nodes on board $B(0,0,m)$, $m=1,2,\dots,k_x-1$ will have to redirect their traffic in a direction orthogonal to the fault; in this case it could be either the y or the z direction. A similar fault can occur when a fiber that connects different boards along the y or z dimension breaks down. In this case, to bypass the fault, nodes will have to first transmit in the x/z and x/y directions. We use $xyz-yzx-zxy$ routing. That is if you arrive from the x direction and further routing is needed, then you will route in the y direction. In case routing in the y direction is not needed, the requirement for z direction routing is checked (if needed) and the appropriate step is taken. Similar steps are followed if the packet arrives from the y or z directions.

The routing algorithm guarantees success when there is at least one path available from the source to the destination. When all links to a board break down, the board becomes isolated and as a result will not be able to receive any packets. Also, in this paper, we have assumed that the transmitters and receivers are faultless and have the ability to detect faults in the links through which they send and receive packets. The above-mentioned faults are subjects of another paper.

3.C. Fault-Tolerant Routing Algorithm

The fault-tolerant routing algorithm is briefly explained below. Step 1 remains the same. Steps 2, 3, and 4 incorporate $xyz-yzx-zxy$ routing, i.e., if the packet was previously routed along the x, y, z direction, it will now be sent along the y, z, x direction, respectively. In step 5, we check to see if routing in the x direction is required and possible. In case routing in the x direction is not possible due to some fault, we test for routing in the y direction. Routing in the y direction will occur if it is required and if there is no fault at the destination board in the y direction. Otherwise we see if routing is required in the z direction. In case routing in the z direction is not required or it is not possible due to a fault, we go to the x fault. Step 6 checks for the routing requirement in the y direction. If routing in the y direction is required and not possible due to a fault, the algorithm tries to see if routing is possible in the z direction; else it goes to the y fault. Step 7 checks for the routing requirement in the z direction and goes to the z fault in case it comes across a link failure. x, y, and z faults basically help bypass the fault by first sending the packet in a direction orthogonal to the fault and then along a route parallel to the fault till the fault has been bypassed. The algorithm is summarized below.

Fault-Tolerant Routing Algorithm

```
STEP 1: if ($x_p$==$x_d$) && ($y_p$==$y_d$) && ($z_p$==$z_d$)
  Do intraboard routing. Stop.
STEP 2: else if coming from $x$ direction go to $y$ route
STEP 3: else if coming from $y$ direction go to $z$ route
STEP 4: else if coming from $z$ direction go to $x$ route
STEP 5: $x$ route:
  if ($x_p$!==$x_d$) && ($x_d$!=$x_{fault}$)
      Route along $x$ direction.
      else if ($y_p$!=$y_d$) && ($y_d$!=$y_{fault}$)
        Route along $y$ direction
      else if ($z_p$!=$z_d$) && ($z_d$!=$z_{fault}$)
        Route along $z$ direction
        else go to $x$ fault
STEP 6: $y$ route:
  if ($y_p$!==$y_d$) && ($y_d$!=$y_{fault}$)
      Route along $y$ direction.
      else if ($z_p$!=$z_d$) && ($z_d$!=$z_{fault}$)
        Route along $z$ direction
        else go to $y$ fault
STEP 7: $z$ route:
  if ($z_p$!==$z_d$) && ($z_d$!=$z_{fault}$)
      Route along $z$ direction.
    else if ($x_p$!=$x_d$) && ($x_d$!=$x_{fault}$)
      Route along $x$ direction.
        else go to $z$ fault
STEP 8: $x$ fault:
      go to board $x_p$, $(y_p-1) \bmod L$, $z_p$
STEP 9: $y$ fault:
      go to board $x_p$, $y_p$, $(z_p+1) \bmod C$
STEP 10: $z$ fault:
      go to board $(x_p+1) \bmod B$, $y_p$, $z_p$
```

Again we consider the case when $B(0,3,0)$ node 0 would like to communicate with $B(0,0,1)$ node 1 in Fig. 7. This time let us assume that the fiber to $B(0,3,1)$ is broken, i.e., $x_{\text{fault}}=1$ for level 3.

We see that the condition of step 1 is not true, and since we are at source board at present, steps 2, 3, and 4 are ignored. At step 5, we see that conditions $x_p!=x_d$ holds, but $x_d!=x_{\text{fault}}$ is not true. Hence we are not able to route along the x direction. The next condition of step 5 is satisfied, i.e., $y_p!=y_d$ and $y_d!=y_{\text{fault}}$. Thus we route along to y direction to reach $B(0,0,0)$. At $B(0,0,0)$, the conditions of steps 1 and 2 are not true, but that of step 3 is satisfied and we therefore go to z route. From z route, we route to $B(0,0,1)$. At $B(0,0,1)$ now the condition of step 1 is satisfied, and therefore we do intraboard routing and stop.

It must be pointed out that the location of the failure can affect the total number of hops required to reach the destination and thus the average latency a packet will incur in the system. For example, if we consider a packet that needs to travel along all three dimensions: x, y, and z, if there is a fault in the x direction, the packet can first be routed in the y or z direction and then along the x direction. This results in the packet still traversing three hops in the optical domain. On the other hand, if it comes across a fault in the z direction, since it has already traversed along the x and y directions this results in the effective path resulting in more than three hops in the optical domain.

4. Performance Evaluation

We used YACSIM and NETSIM discrete event simulators to evaluate the performance of nD-RAPID. Its performance is compared with various electrical networks for uniform and permutation traffic traces. The electrical networks chosen for comparison were 2D torus, 3D torus, hypercube, and fat tree. The 2D torus is used in the Alpha 21364 network [29], the hypercube is used in the SGI SPIDER chip used for SGI origin machines [30], and the fat-tree topology is the basis of most Mellanox switches used in Infiniband [31], Elan 2, and QsNet.

4.A. Simulation Methodology

YACSIM and NETSIM can be combined to construct a wide range of direct and indirect electrical networks. NETSIM was modified to implement virtual channels at the input ports. To implement an optoelectronic network such as nD-RAPID the NETSIM component library had to be augmented, since it was originally built only for electrical networks. Optical components such as couplers, fibers, waveguides, demultiplexers, and splitters were added. The functional modeling of the components at the system level was implemented to determine three parameters of interest: (a) length, to determine the propagation latency; (b) attenuation, to determine the signal loss due to the components; and (c) wavelength, to determine routing within a component (waveguide).

Cycle accurate simulations were used to evaluate the performance of nD-RAPID and other electrical interconnects. The load was varied from 0.1 to 0.9 of the network capacity. Network capacity was determined from the expression N_c (packets–node–cycle), which is defined as the maximum sustainable throughput when a network is loaded with uniform traffic [10]. The simulator was warmed up under load without taking any measurements until steady state was reached. Then a sample of injected packets was labeled during a measurement interval. The simulation was allowed to run until the packets reached their destination.

The electrical network router model parameters reflect the design from the SGI SPIDER routing chip. The SPIDER routing chip [30] has a channel width of 16 bits, and its speed is 400 MHz, resulting in a unidirectional bandwidth of 6.4 Gbits/s and a per port bidirectional bandwidth of 12.8 Gbits/s. For the optical network, we assume a channel speed of 10 GHz and a transmission rate of 10 Gbits/s. The packet size was kept constant at 64 bytes, with the flit size as 8 bytes.

Traffic patterns commonly found in scientific applications were used to measure the performance. We used uniform traffic, butterfly, complement, and perfect shuffle traffic patterns [32]. The performance of the networks was measured in terms of throughput and latency [10,33–35]. Throughput is defined as the sustained data delivery rate given some offered bandwidth at the network input. The network latency is the average delay spent by a packet in the network, from the insertion of the head flit into the input buffer till the reception of the tail flit at the destination. It includes the source queuing delay to ensure that the traffic pattern being applied at the measurement points in the intended pattern.

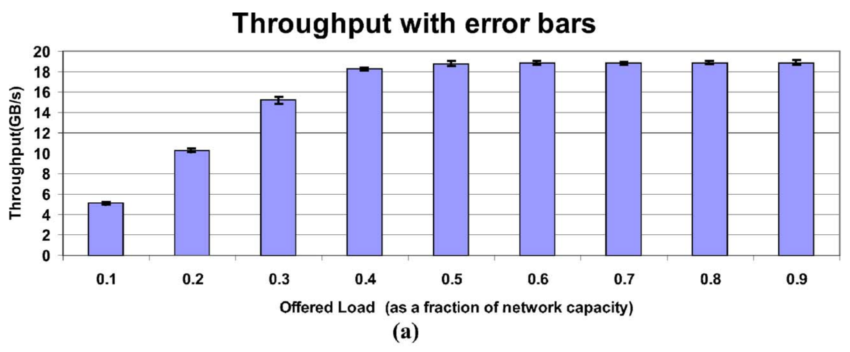
4.B. Simulation Validation

To validate our simulation methodology, we took a 64 node 3D-RAPID system and ran it 16 times, changing the seed of the random function generator each time. We then calculated the mean throughput and latency at each load (from 0.1 to 0.9). The confidence interval was calculated according to standard mathematical formulas.

Figure 8(a) shows the throughput variation, with the bar representing the mean value at each load. The error bars represent the maximum and minimum throughput measured at each load. Figure 8(b) shows the latency variation with the bar representing the mean value and the error bars showing the maximum and minimum values measured. From our measurements, we found that the values lie within the 99% confidence interval.

4.C. Simulation Results

In this section, we present our simulation results. In Fig. 9 we compare the throughput of a 64 node 1D-RAPID, 2D-RAPID, fat-tree, hypercube, and torus. In Figs. 10 and 11, we compare the average latency for the above networks with different traffic patterns, namely, uniform, perfect shuffle, butterfly, and complement. In Fig. 12, we show the results of bidirectional traffic that we simulated for 2D-RAPID. Figure 13 shows the throughput and latency results for a 256 node 2D-RAPID, 3D-RAPID, fat-tree, hypercube, and 3D torus. In Fig. 14, we compare throughput and latency for a 512 node 3D-RAPID, 3D torus, and hypercube. Figures 15 and 16 look into the performance of the fault-tolerant routing algorithm for 2D-RAPID and 3D-RAPID, respectively. Figure 9 compares the throughput of 64 node 1D-RAPID and 2D-RAPID with



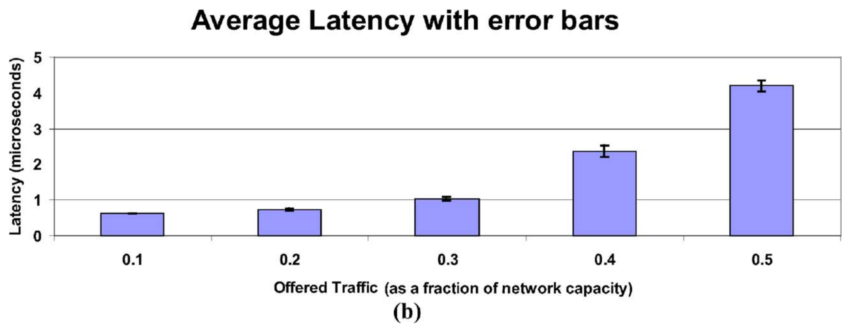


Fig. 8. (a) Throughput for 64 node 3D-RAPID with uniform traffic. The bars represent the average values, whereas the error bars represent the maximum and minimum values measured. (b) Latency for 64 node 3D-RAPID with uniform traffic. The bars represent the average value, whereas the error bars represent maximum and minimum values measured.

hypercube, torus, and fat-tree for uniform, perfect shuffle, butterfly, and complement traffic patterns. Hypercube shows the best performance among the electrical systems. For uniform traffic, we see that 2D-RAPID outperforms hypercube by ~22.1% in throughput. 2D-RAPID outperforms hypercube in perfect shuffle and butterfly traffic patterns as well. For complement traffic, both hypercube and 2D torus show better performance than 2D-RAPID. This is because of the basic design of nD-RAPID, where all nodes on a board use the same wavelength for a given destination board. This results in high contention for the same wavelength, thereby resulting in low throughput and high average latency. Due to the fact that in 2D-RAPID the nodes are distributed among two dimensions, the contention is less as compared with 1D-RAPID. This is why 2D-RAPID shows better performance than 1D-RAPID.

Figure 10(a) compares the average latency of the above networks for uniform traffic. As can be seen, 1D-RAPID and 2D-RAPID saturate at much higher loads than the electrical systems. Figure 10(b) compares the average latency for perfect shuffle. We

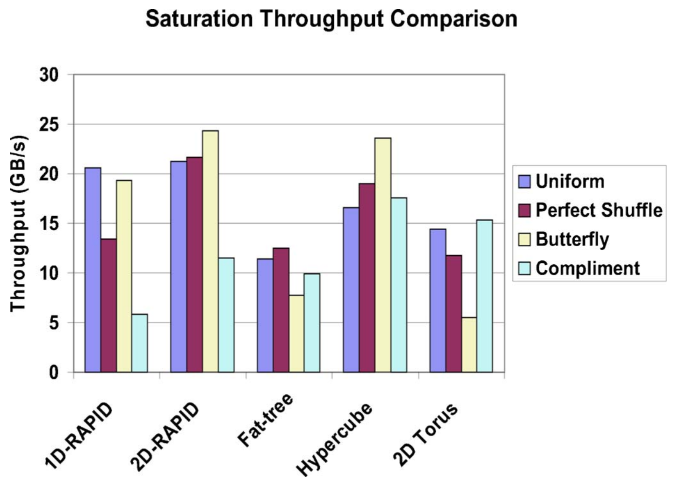
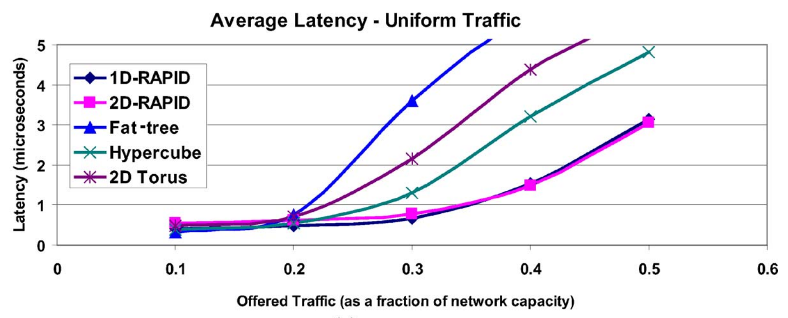
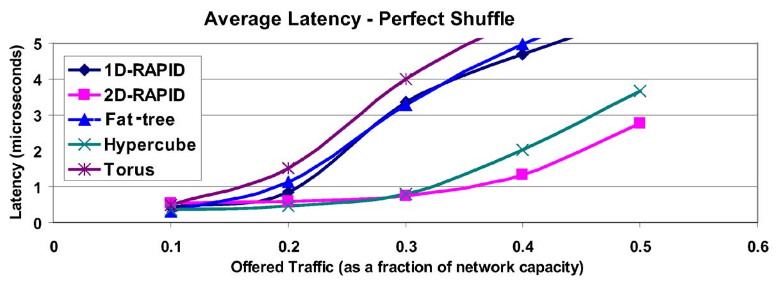


Fig. 9. Throughput comparison of 64 node 1D-RAPID, 2D-RAPID, fat-tree, hypercube, and 2D torus.



(a)

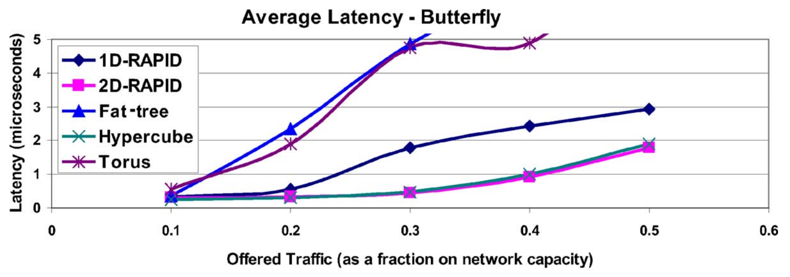


(b)

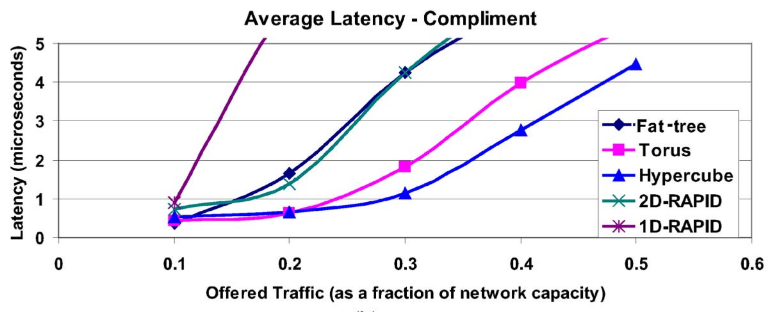
Fig. 10. Latency comparison 64 node 1D-RAPID, 2D-RAPID, fat-tree, hypercube, and 2D torus for (a) uniform traffic and (b) perfect shuffle traffic.

see that while 2D-RAPID saturates at much higher loads, 1D-RAPID saturates faster. This is because in 2D-RAPID we are able to expand in two dimensions, thereby reducing the probability that the destination nodes for a given source board will lie on the same board. This, in turn, reduces the contention for a single wavelength. Figures 11(a) and 11(b) compare the latency values for butterfly and complement traffic patterns.

We simulated bidirectional traffic, and the results are shown below in Fig. 12. We used a 64 node 2D-RAPID system in which nodes 0 and 63 communicate with each other in a bidirectional manner, and the remaining nodes communicate among one another in a uniformly random fashion. With low load in the network, the bidirec-



(a)



(b)

Fig. 11. Latency comparison of 64 node 1D-RAPID, 2D-RAPID, fat-tree, hypercube, and 2D torus for (a) butterfly traffic, and (b) compliment traffic.

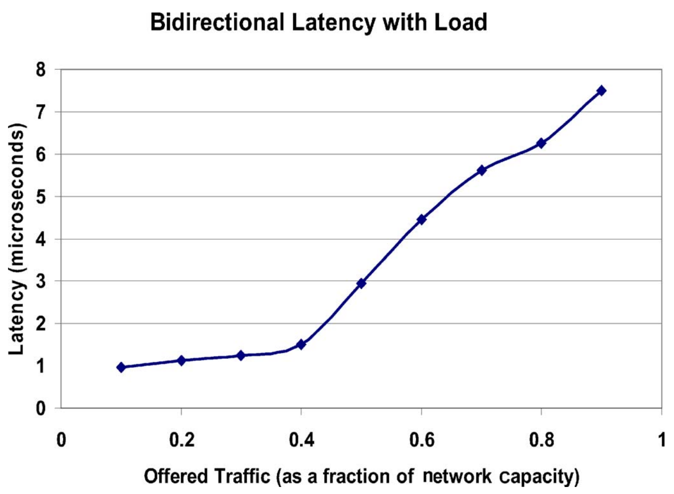


Fig. 12. Bidirectional latency in a 64 node 2D-RAPID system with nodes 0 and 63 carrying out bidirectional ping and the remaining nodes following uniform traffic.

tional latency increases slowly, in fact, at a load on 0.1, the bidirectional latency is very close to the bidirectional latency at no load as expected. Once the network reaches saturation, the bidirectional latency also shoots up due to the queuing delays it encounters in the network.

Figure 13 compares 2D-RAPID and 3D-RAPID with fat-tree, hypercube, and 3D torus for a 256 node system with uniform traffic. As was predicted earlier, 2D-RAPID shows higher throughput at saturation than 3D-RAPID. This is because the aggregate optical bandwidth per board for 2D-RAPID would be $14\times10=140$ Gbits/s whereas for 3D-RAPID, it is $9\times10=90$ Gbits/s. This is shown in Fig. 13(a). Figure 13(b) shows the latency comparison at saturation between the systems.

In Fig. 14, we compare a 512 node 3D-RAPID with hypercube and 3D torus networks for uniform traffic. As Figs. 14(a) and 14(b) show, 3D-RAPID shows 45% higher throughput and saturates at a higher load than hypercube and 3D torus. This shows that as the network size increases, the advantage of using optics improves.

Figure 15 shows the effects of faults in 64 node 2D-RAPID system for uniform traffic. $X1$ means there is one fault in the x direction, $Y1$ means there is one fault in the y direction, and $X1Y1$ means there is a fault present in both the x and the y directions. Even with two faults in the system, we see only an 8% decrease in throughput.

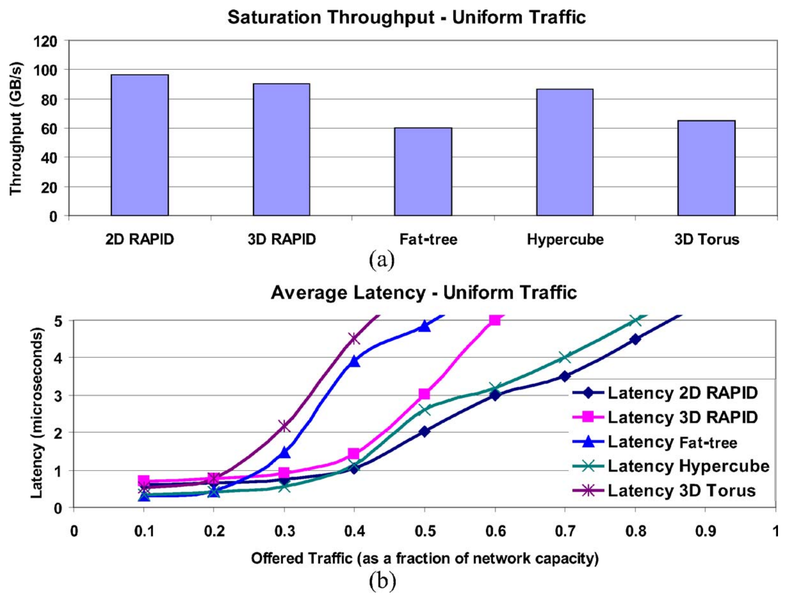


Fig. 13. Comparison of 256 node 2D-RAPID, 3D-RAPID, fat-tree, hypercube, and 3D torus for (a) throughput comparison and (b) latency comparison.

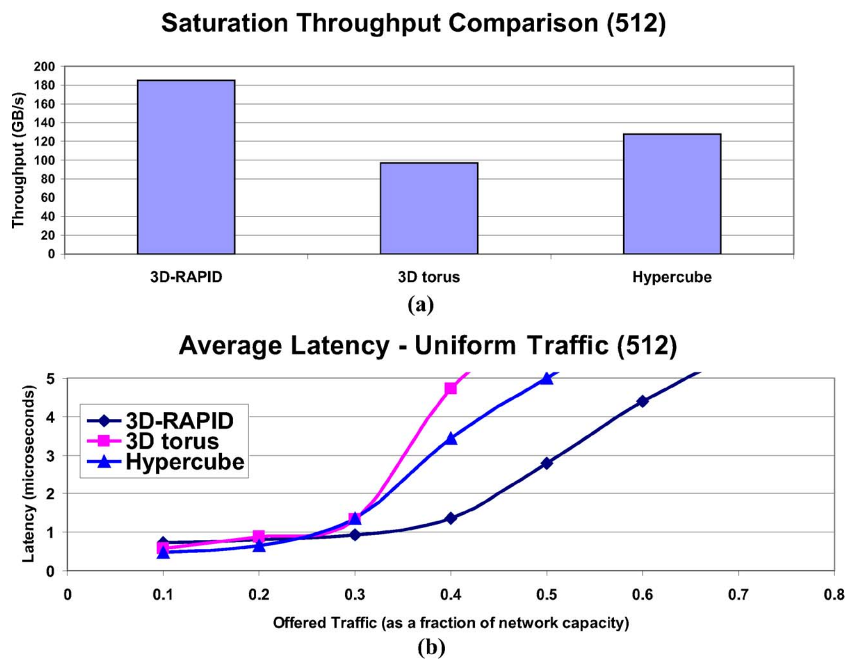


Fig. 14. Comparison of 512 node 3D-RAPID, 3D torus, and hypercube. (a) Throughput comparison. (b) Latency comparison.

Figure 16 shows the effects of faults in 64 node 3D-RAPID system for uniform traffic. We use the same notation as above, with the only new symbol $Z1$ representing a fault in the z direction. We used four boards in the x direction, and two boards each in the y and z directions to create a 64 node 3D-RAPID. Due to the asymmetry in the and system, the load in the x direction will be twice the load in the y and z directions [10]. Thus we see that a fault in the x direction has maximum impact on throughput and latency. With three faults in the system, we see a decrement of only 9.3% in throughput.

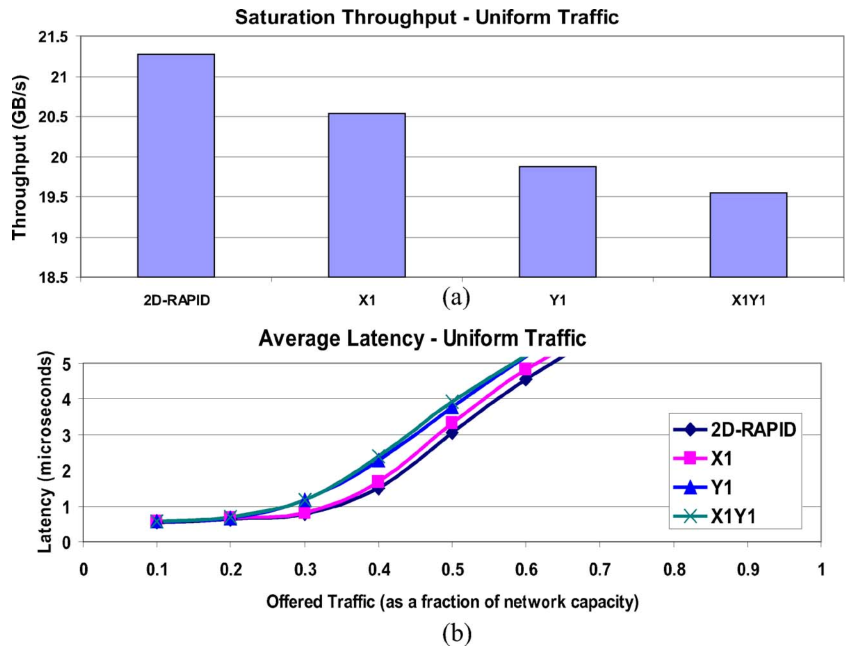
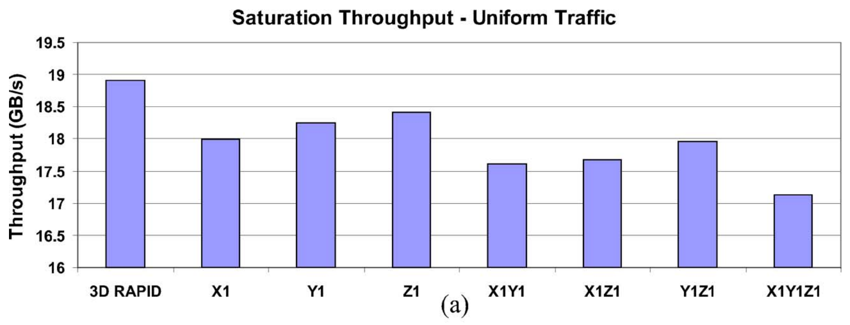


Fig. 15. Comparison of 2D-RAPID with faults. $X1$ denotes a fault in the x direction, $Y1$ denotes a fault in the y direction, and $X1Y1$ denotes a fault each in the x and y directions. (a) Throughput comparison. (b) Latency comparison.



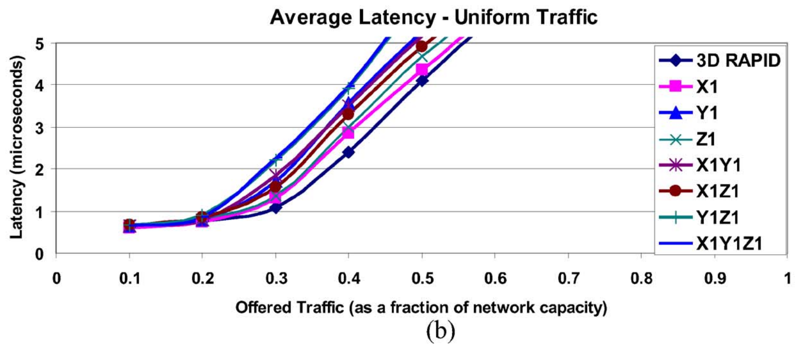


Fig. 16. Comparison of 3D-RAPID with faults. $X1$ and $Y1$ have the same notations as before, and $Z1$ indicates a fault in the z direction. (a) Throughput comparison. (b) Latency comparison.

5. Conclusion

In this paper, we propose a multidimensional optoelectronic architecture, nD-RAPID for HPCS, that provides fault tolerance and dynamic reconfigurability. While taking maximum advantage of the high bandwidth optics has to offer, we reduce the costs by building an architecture where the optical active components are present only on the board. The architecture is made fault tolerant by employing a n-dimensional ($n=2,3$) structure, which ensures that there is more than one path to every board. The fault-tolerant algorithm was designed with the idea that it should not affect performance in the absence of faults and show very little performance degradation in the presence of faults. Dynamic reconfigurability is achieved by rerouting packets when they come across a faulty link. nD-RAPID was simulated using a cycle accurate simulator to verify the results, and it was seen that in the worst-case faulty performance, throughput fell by 9.3% for a 64 node system. When nD-RAPID was compared with other popular networks for HPCS, it was seen that it consistently outperformed them in most of the traffic patterns tested.

Acknowledgments

This research is supported by National Science Foundation grants CCR-0309537 and CCF-0538945 and a grant from Intel Corporation.

References and Links

1. B. E. Lemoff, M. E. Ali, G. Panotopoulos, G. M. Flower, B. Madhavan, A. F. J. Levi, and D. W. Dolfi, “MAUI: enabling fiber-to-the processor with parallel multiwavelength optical interconnects,” J. Lightwave Technol. **22**, 2043–2054 (2004).
2. D. E. Culler, J. P. Singh, and A. Gupta, *Parallel Computer Architecture: A Hardware/Software Approach* (Morgan Kaufmann, 1999).
3. D. Huang, T. Sze, A. Landin, R. Lytel, and H. L. Davidson, “Optical interconnects: out of the box forever?,” IEEE J. Sel. Top. Quantum Electron. **9**, 614–623 (2003).
4. E. Mohammed, A. Alduino, T. Thomas, H. Braunisch, D. Lu, J. Heck, A. Liu, I. Young, B. Barnett, G. Vandenton, and R. Mooney, “Optical interconnect system integration for ultra-short-reach applications,” Intel Technol. J. **8**, 114–127 (2004).
5. A. F. Benner, M. Ignatowski, J. A. Kash, D. M. Kuchta, and M. B. Ritter, “Exploitation of optical interconnects in future server architectures,” IBM J. Res. Dev. **49**, 755–775 (2005).
6. D. A. B. Miller, “Rationale and challenges for optical interconnects to electronic chips,” Proc. IEEE **88**, 728–749 (2000).

7. J. H. Collet, D. Litaize, J. V. Campenhout, C. Jesshope, M. Desmulliez, H. Thienpont, J. Goodman, and A. Louri, "Architectural approaches to the role of optics in monoprocessor and multiprocessor machines," Appl. Opt. **39**, 671–682 (2000).
8. A. K. Kodi and A. Louri, "Rapid for high-performance computing systems: architecture and performance evaluation," Appl. Opt. **45**, 6326–6334 (2006).
9. http://www.top500.org/.
10. W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks* (Morgan Kaufmann, 2004).
11. A. A. Chen and J. H. Kim, "Planar-adaptive routing: low cost adaptive networks for multiprocessors," J. ACM **42**, 91–123 (1995).
12. T. M. Pinkston, R. Pang, and J. Duato, "Deadlock-free dynamic reconfiguration schemes for increased network dependability," IEEE Trans. Parallel Distrib. Syst. **14**, 780–794 (2003).
13. Infiniband Trade Association, http://www.infinibandta.com.
14. F. Petrini, W. Feng, A. Hoisie, S. Coll, and E. Frachtenberg, "The Quadrics Network (QsNet): high performance clustering technology," in *Proceedings of the 9th IEEE Hot Interconnects* (IEEE, 2001), pp. 125–130.
15. J. C. Sancho, A. Robles, and J. Duato, "A flexible routing scheme for networks of workstations," *Proceedings of the International Conference on High Performance Computing* (Springer, 2000), pp. 260–267.
16. M. S. Chen and K. G. Shin, "Adaptive fault tolerant routing in hypercube multicomputers," IEEE Trans. Comput. **39**, 1406–1416 (1990).
17. M. S. Chen and K. G. Shin, "Depth-first search approach for fault-tolerant routing in hypercube multicomputers," IEEE Trans. Parallel Distrib. Syst. **1**, 152–159 (1990).
18. Y. M. Boura and C. R. Das, "Fault-tolerant routing in mesh networks," in *Proceedings of the 1995 International Conference on Parallel Processing*, Vol. 1, pp. 106–109, 1995.
19. M. E. Gomez, N. A. Nordbotten, J. Flich, P. Lopez, A. Robles, J. Duato, T. Skeie, and O. Lysne, "A routing methodology for achieving fault tolerance in direct networks," IEEE Trans. Comput. **4**, 400–415 (2006).
20. R. V. Boppana and S. Chalsani, "Faul tolerant wormhole routing algorithms for mesh networks," IEEE Trans. Comput. **44**, 848–864 (1995).
21. A. Louri and H. Sung, "An optical multi-mesh hypercube: a scalable optical interconnection network for massively parallel computing," J. Lightwave Technol. **12**, 704–716 (1994).
22. A. Louri and B. Weech, "A spanning multichannel linked hypercube: a gradually scalable optical interconnection network for massively parallel computing," IEEE Trans. Parallel Distrib. Syst. **9**, 497–511 (1998).
23. R. D. Chamberlain, M. A. Franklin, and C. S. Baw, "Gemini: an optical interconnection network for parallel processing," IEEE Trans. Parallel Distrib. Syst. **13**, 1038–1055 (2002).
24. K. Barker, A. Benner, R. Hoare, A. Hoisie, A. K. Jones, D. J. Kerbyson, D. Li, R. Melham, R. Rajamony, E. Schenfeld, S. Shao, C. Stunkel, and P. Walker, "On the feasibility of optical circuit switching for high performance computing systems," *Super Computing Conference SC'05*, November 2005.
25. S. Banerjee and D. Sarkar, "Hypercube connected rings: a scalable fault tolerant logical topology for optical networks," Tech report, Dept. of ECE, (University of Miami, Florida, 1994).
26. P. Lalwaney and I. Koren, "Fault-tolerant schemes for WDM-based multiprocessor networks," in *Proceedings of the 2nd Workshop on Massively Parallel Processing using Optical Interconnects (MMPOI'95)*, (IEEE, 1995), pp. 90–97.
27. Y. Yang and J. Wang, "A fault-tolerant rearrangeable permutation network," IEEE Trans. Comput. **53**, 414–426 (2004).
28. B. Helvik and R. Andreassen, "Fault tolerance in optical networks; a study of electronic in- and egress interconnections in torus topologies," in *Proceedings of the 9th Conference on Optical Network Design and Modelling, ONDM* (IEEE, 2005).
29. S. S. Mukherjee, P. Bannon, S. Lang, A. Spink, and D. Webb, "The alpha 21364 network architecture," IEEE Micro **22**, 26–35 (2002).
30. M. Galles, "Spider: a high-speed network interconnect," IEEE Micro **17**, 34–39 (1997).
31. Mellanox Technologies, http://www.mellanox.com/
32. F. Petrini, E. Frachtenberg, A. Hoisie, and S. Coll, "Performance evaluation of the quadrics interconnection network," J. Cluster Computing **6**, 125–142 (2003).
33. A. Singh, W. J. Dally, A. Gupta, and B. Towles, "GOAL: a load balanced adaptive routing algorithm for torus networks," in *Proceedings of the 30th Annual International Symposium on Computer Architecture* (ACM, 2003), 194–205.
34. A. Singh, W. J. Dally, B. Towles, and A. K. Gupta, "Globally adaptive load-balanced routing in tori," Comput. Arch. Lett. **3**, (2004).
35. Y. Qian, A. Afsahi, N. R. Fredrickson, and R. Zamani, "Performance evaluation of the sun fire link SMP clusters," in *Proceedings of the the 18th International Symposium on High Performance Computing Systems and Applications* (IEEE, 2004), pp. 145–156.