



Future Technologies Group  
Computer Science and Mathematics Division  
Oak Ridge National Laboratory  
Oak Ridge, Tennessee  
USA

<http://ft.ornl.gov>

## IAA Interconnection Networks Workshop 2008

Authors K. Scott Hemmert, Sandia National Laboratories  
Jeffrey S. Vetter, Oak Ridge National Laboratory and Georgia Institute  
of Technology  
Keren Bergman, Columbia University  
Chita Das, NSF and Pennsylvania State University  
Azita Emami, California Institute of Technology  
Curtis Janssen, Sandia National Laboratories  
Dhabaleswar K. Panda, Ohio State University  
Craig Stunkel, IBM  
Keith Underwood, Intel  
Sudhakar Yalamanchili, Georgia Institute of Technology

Abstract The report summarizes the findings and recommendations of a  
workshop on Exascale Interconnection Networks in July 2008. The  
workshop was sponsored by the Institute for Advanced Architectures  
and Algorithms (IAA).

Future Technologies Group Technical Report FTGTR-2009-03  
Total Pages 24  
Publication Date 4/27/2009  
Also published as -

Report on

**Institute for Advanced Architectures and Algorithms  
Interconnection Networks Workshop 2008**

July 21-22, 2008  
San Jose Doubletree Hotel  
San Jose, California

<http://www.csm.ornl.gov/workshops/IAA-IC-Workshop-08/>

**Workshop Chairs**

Scott Hemmert, Sandia National Laboratories  
Jeffrey Vetter, Oak Ridge National Laboratory and Georgia Institute of Technology

**Organizing Committee**

Keren Bergman, Columbia University  
Ron Brightwell, Sandia National Laboratories  
Candace Culhane, DOD  
Bill Dally, Stanford University  
Bill Harrod, DARPA  
Thuc Hoang, NNSA  
Fred Johnson, DOE Office of Science  
Scott Pakin, Los Alamos National Laboratories

**Working Group Chairs**

Keren Bergman, Columbia University  
Chita Das, NSF and Pennsylvania State University  
Azita Emami, California Institute of Technology  
Curtis Janssen, Sandia National Laboratories  
DK Panda, Ohio State University  
Craig Stunkel, IBM  
Keith Underwood, Intel  
Sudhakar Yalamanchili, Georgia Institute of Technology

# TABLE OF CONTENTS

---

<b>1</b>	<b>INTRODUCTION .....</b>	<b>3</b>
1.1	WORKSHOP ORGANIZATION .....	3
1.2	REPORT OUTLINE.....	4
<b>2</b>	<b>REQUIREMENTS.....</b>	<b>4</b>
2.1	PERFORMANCE.....	4
2.2	EXASCALE SYSTEM CONFIGURATIONS AND REQUIREMENTS .....	5
<b>3</b>	<b>TECHNICAL THRUST: TOPOLOGY AND ROUTING .....</b>	<b>6</b>
3.1	STATUS .....	7
3.2	CHALLENGES .....	7
<b>4</b>	<b>TECHNICAL THRUST: PROCESSOR NETWORK INTERFACE.....</b>	<b>9</b>
4.1	STATUS .....	9
4.2	CHALLENGES .....	10
<b>5</b>	<b>TECHNICAL THRUST: SIMULATION AND PERFORMANCE PREDICTION .....</b>	<b>13</b>
5.1	STATUS .....	14
5.2	CHALLENGES .....	15
5.3	NON-TECHNICAL STRATEGIC ISSUES .....	16
<b>6</b>	<b>TECHNICAL THRUST: DEVICE TECHNOLOGY .....</b>	<b>17</b>
6.1	EMERGING INTERCONNECT TECHNOLOGIES .....	17
6.2	POTENTIAL IMPACT.....	18
6.3	CHALLENGES .....	18
6.4	MATURITY-IMPACT-COST ANALYSIS .....	18
<b>7</b>	<b>APPENDIX – WORKSHOP AGENDA.....</b>	<b>20</b>
	DAY 1 – MONDAY, JULY 21 .....	20
	DAY 2 – TUESDAY, JULY 22.....	20
<b>8</b>	<b>APPENDIX – ATTENDEES.....</b>	<b>21</b>
	<b>REFERENCES .....</b>	<b>23</b>

# 1 INTRODUCTION

---

A number of recent reports [8, 9, 12] have focused on the requirements for and technical challenges to constructing an Exascale computing system. These reports are consistent in the notion that high performance computing (HPC) continues to push the frontier of computing architectures, both in terms of single node performance and overall component count. Furthermore, these reports concur that supercomputer interconnection networks are the key hardware component that define the level of the architecture scalability [1, 4-6], which, in turn, defines application scalability. Current approaches to interconnect design do not deliver latencies and message throughputs that are commensurate with either the expected computational power of nodes or with the expected scale of systems under consideration. Thus, for the foreseeable future, the imbalance in baseline interconnect performance will be one of the key challenges in scaling HPC architectures to the Exascale. Based on HPC architectural roadmaps, future capability systems are likely to require the following from interconnection networks:

- Scalability: greater than 100,000 endpoints (impacting power, cabling, cost, failures, etc.);
- High Bandwidth: greater than 100 GB/s for 1 TF nodes;
- High Message Throughput: greater than 100 million messages/s for MPI and greater than 1000 million messages/s for load/store communication models;
- Low Latency: Maintain approximately 1  $\mu$ s latency across system; and,
- High Reliability: less than  $10^{-23}$  unrecovered bit error rates.

Although the industry's commodity offerings attempt to address the challenge with respect to peak bandwidth, industry is not adequately addressing the requirements of messaging rate, effective throughput, system/connection scalability, and reliability.

## 1.1 Workshop Organization

Our analysis, presented in this report, brings together experts from several organizations to study these gaps for HPC interconnection networks, not only for the applications and programming models of today but also of those expected to be common in the future. The remainder of this report details the findings generated at the Interconnection Networks Workshop held in San Jose, California on July 21 and 22, 2008. During the workshop, attendees participated in four working groups: topology and routing, processor-network interface, simulation and performance prediction, and device technology. Multiple speakers and presentations augmented the workshop discussions; Section 7 documents the agenda for the workshop.

The workshop attendees were given the goal of deploying an Exascale system in the ~2016 timeframe (cf. Section 2.2). Then, each working group was chartered with delivering a prioritized list of challenges for their specific topic as rated by the group. Although the scheme for rating priorities varied across working groups, in general, they ranked each specific challenge on two dimensions: likelihood and impact. *Likelihood* is the probability that current technology trends would not achieve our Exascale requirements in the given timeframe. That is, given current technology trends, do you expect the target technology to be ready for Exascale computing? *Impact* is the severity level for deploying an Exascale computer in the given timeframe. Said differently, how important will the lack of a solution for this challenge be for applications targeting Exascale architectures? Does a reasonable contingency technology exist? Taken together, these two dimensions were used to rate each technical item, and generate a specific rank.

The Device Technology working group used a bottom-up methodology to review and evaluate the maturity and potential impact of new emerging technologies that could play a role in this design. The organizing committee

emphasized that many critical areas were outside the scope of this charter; these important areas include system software and runtime systems, programming models, and I/O.

## 1.2 Report Outline

We begin in the following section with a short summary of expected Exascale hardware configuration requirements and trends. The next sections detail the results from the four working groups as technical thrusts. For each technical thrust, we review the current state of the art and the requirements for Exascale systems, after which, we detail the challenges and our recommendations for addressing these requirements.

## 2 REQUIREMENTS

Figure 1 shows the hardware organization and software architecture of a modern HPC system. The system hardware consists of a collection of compute nodes, each containing processors, memory, and a network interface controller (NIC). In most existing system node designs, the NIC is attached to the I/O subsystem (such as PCI-Express), but future designs are expected to use processor point-to-point connections to improve performance. A typical NIC organization is shown in the figure, containing a dedicated processor, some onboard memory, a DMA engine, command queues for software interaction, and queues for transmitting packets to and from the network. Figure 1b shows the software architecture of processes running on a typical HPC system. The figure shows a potential architecture; for example, some processes will not use a structured data library or parallel I/O library. The figure also shows several potential mechanisms for interacting with the NIC, including OS drivers, OS-bypass libraries, and even load-store instructions on systems with Global Address Space hardware support. The figure also reflects our belief in the advantages of offloading some communication support onto the NIC. For example, a NIC might include a specialized Queue Processor for accelerating list processing to improve MPI performance.

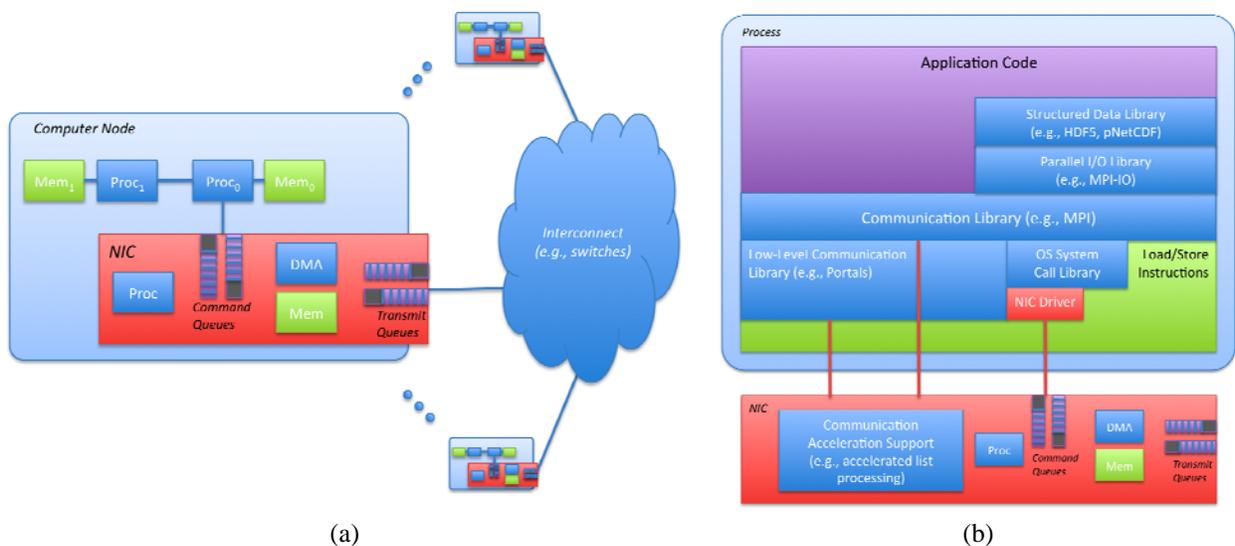


Figure 1: Hardware organization (a) and software architecture (b) including interaction with the NIC of a typical modern HPC machine.

## 2.1 Performance

Interconnect performance is broadly governed by a number of factors: peak bandwidth, small message latency, and message rate. In addition to the performance of the network interface and routers in the network itself, all three

parameters are greatly influenced by the overall node design. Memory performance, cache utilization, and even software architecture all can significantly impact overall interconnect performance.

Peak link bandwidth has recently grown at a steady pace that, while not keeping up with the exponential growth in processor performance, has outpaced the growth in memory performance. Although this trend is expected to reverse itself, there is a large amount of uncertainty as optical cables are adopted by the industry in response to prohibitive cost and physical limits of increasing signaling rates over copper cables. The adoption of optical cables and the growing cost of switching at peak bandwidth have begun to drive interest in alternative topologies, beyond the traditional mesh and fat tree.

Latency has been asymptotically decreasing, with MPI latency slightly higher than 1  $\mu$ s microsecond with modern interconnects [5, 6, 16]. Only a small fraction of the latency is due to the wire propagation delay for the message. In fact, much of the MPI latency is spent at the source and destination hosts, moving data from the network to the host processor and in the communication software stack. Although there is interest in decreasing the data movement latency in the host by bypassing PCI-Express interface, there are few networks that provide such technology. Furthermore, there is constant tension between specialized and commodity interfaces in terms of cost, flexibility, and leverage of software and other tools. Efforts to reduce the software overhead of communication have focused mainly on accelerating message processing tasks with hardware support.

Message rate has recently become an important performance metric for HPC networks, although current message rate performance has focused almost entirely on best-case MPI matching. That is, rates are measured when there are a limited number of entries in the posted receive queue and the incoming message always matches the first item in the queue. There is also little work done between messages, leading to all matching occurring in the CPU's cache. This promotes a network interface design where message processing occurs on the host processor, which is almost always more powerful at matching than the NIC processor. Unfortunately, in real world applications, it is rare that the posted receive queue is small and that the incoming message matches the first item on the list. Furthermore, an application's computation and communication share the same caching structure, increasing cache misses during message matching. This has a large adverse effect on messaging rates for real world applications [15].

## 2.2 Exascale System Configurations and Requirements

To set the stage for Exascale system requirements and their demands on the system interconnect, the workshop started with a sketch of system designs for 2011, 2015, and 2019. For each configuration, both the desired performance ('want') and the expected performance ('expected') were presented. Table 1 (from [14]) presents the interconnect requirements summary for these configurations.

Each of the metrics will be discussed in more detail later in the report; however, the table presents some daunting differences between the desired and expected performance of future interconnects. First, the rapid increase in performance of the processors due to increasing core counts and wider functional units will provide individual nodes with tremendous peak computing capability: an expected increase to at least 25 TF per socket by 2019. Meanwhile, the bandwidth between nodes will increase only slightly, relatively speaking. This gap will make applications even more sensitive to data locality. Second, the increased number of cores per socket will dramatically increase the number and decrease the size of messages that the interconnect must transmit. In this regard, the processor-network interface must be capable of not only high bandwidth for large messages, but also of high message throughput for small messages, which will be generated from either MPI message or globally addressable load and store memory operations. The gap for desired message throughput will range from 4 to 21 times the expected values, for MPI and Load/Store traffic, respectively. Third, given this dramatic imbalance in the node's computational resources and the interconnect, real applications will continue to extract a smaller fraction of performance from the overall system. In summary, it will become increasingly difficult to maintain system balance for the next decade without significant technological advances in chip I/O and system interconnects.

Table 1: System interconnect requirements and expectations (from [14]).



## System Interconnects

	2011		2015		2019	
System Size	32,768		32,768		32,768	
Sockets	32		200		800	
Peak PF	1.0		6.1		25.0	
TF/Socket						
	Expect	Want	Expect	Want	Expect	Want
NIC B/W (B/F)	0.01 - 0.1	1.0	0.005 - 0.03	1.0	0.025 - 0.25	1.0
Link B/W (B/F)	0.01 - 0.1	1.0	0.005 - 0.03	1.0	0.025 - 0.25	1.0
MPI Latency (ns)	750 - 1500	500	500 - 1000	400	400 - 750	300
MPI Throughput (M Msg/s)	20	50	80	300	300	1200
Load/Store (M Msg/s)	75	400	150	1,600	300	6400
Load/Store Latency (ns)	300	100	300	100	300	100

The HPC applications community has largely adopted message passing, specifically the Message Passing Interface (MPI) [13], as the parallel programming paradigm of choice for large-scale applications. Recent research has focused more on compiler-derived parallelism (OpenMP) [11] and Partitioned Global Address Space (PGAS) languages [2, 10], but neither has seen the adoption rate of MPI for large-scale computing. Modern networks provide consistent growth in peak bandwidth, asymptotically decreasing 0-byte message latency, and deceptively high peak message rates. In this context, vendors have generally optimized their networks for message passing, although not always providing a good match to MPI's strict ordering and message matching requirements.

### 3 TECHNICAL THRUST: TOPOLOGY AND ROUTING

Chairs: Chita Das, Craig Stunkel

The Topology and Routing working group concentrated on the design issues of Exascale interconnection networks contained in the Interconnect cloud of Figure 1(a). Specifically, the discussion focused on a number of key issues covering topology exploration, switch or router micro-architecture design (including internal data path, flow-control, and error detection/correction), routing algorithms, Exascale workload characteristics, performance, power, technology impacts, and QoS requirements. Although interconnection networks are a well researched area, the design of a system with about 100,000 endpoints, where each endpoint could have hundreds of cores in the 2015-2019 timeframe needs ingenious solutions on several fronts. In addition, the design and performance of a network are closely coupled with the foci of the other working groups: NICs and the messaging software, performance prediction with realistic workloads/traces, and design of the interconnection components with emerging technologies, such as optics and 3D integrated chip stacking.

## 3.1 Status

The discussion started with examining the state of networks today and lessons learnt from the prior capability systems. The main observations are the following:

1. Two topology families dominate: fat-trees, and k-ary n-cubes, such as tori and meshes. The convergence from a multitude of proposed topologies to a small number of flourishing topologies is due to several factors, including incremental scalability, ease of implementation, relative simplicity of providing deadlock-free routing with a small number of *virtual channels*, acceptable performance for common traffic patterns, fault-tolerance and adaptivity, and high global bandwidth.
2. Among fat-tree networks, the switch radix (the number of ports in a switch) has been growing steadily in recent years. This is a consequence of increases in both transistor density and I/O bandwidth per chip as well as a move to increased serialization. It is now possible to build chips with attractive port bandwidth for 32 ports or more. Deep sub-micron technology has facilitated larger buffers and high-performance switch micro-architectures to achieve high throughput for such a large number of ports.
3. Multi-core and many-core chips will require on-chip networks of their own. This may result in a hierarchy of networks (on-chip and off-chip) within an HPC system. Commodity switches, such as for InfiniBand and Ethernet, and common proprietary switches, such as Myrinet, are used most commonly in fat-tree networks (although they have some performance issues such as that they do not natively support deadlock-prevention mechanisms for tori). Such switches with enhanced features/performance may be used for the off-chip networks.
4. MPI has been the dominant programming model for HPC applications and is likely to remain prominent due to the wide base of installed applications using MPI. Although MPI would benefit from a flat system, that would restrict the network architecture in terms of designing a hierarchical system and performance. Hybrid models, such as OpenMP [3, 11] on-chip with MPI between chips, are examples of models that are attempting to address the hierarchy presented by today's emerging multicore clusters.
5. New application and traffic genres are emerging. Perhaps most interesting among these are the growing body of streaming applications. In many cases, streaming will drive requirements for high bandwidth, but low latency may be relatively less important for such applications. On the other hand, latency will be critical for small messages and collective communication in a MPI or PGAS programming environment.

## 3.2 Challenges

Based on the examination of current trends and discussions in which we predicted the most pressing needs from future Exascale applications, the working group identified a set of ten challenges that should guide research into high-performance networks for the Exascale era. The points are summarized below:

1. Minimize energy per bit. It was unanimously agreed that power efficiency will be the most important design objective for future interconnects. The choice of network topology will have a major impact on power efficiency. It was observed that indirect topologies utilizing high-radix switches may result in less overall power when combined with optical transmission. Newly-proposed topologies, such as the Dragonfly [7], may further improve power efficiency by reducing the total number of average message hops required to transmit a message. Although several low-power techniques have been recently proposed for on-chip interconnects (NoCs), new ideas/inventions are required to achieve an ambitious objective of 1 pj/bit design envelope (cf. Section 6). This would require energy optimizations across all

the components of a network and would be heavily influenced by the technology trend and circuit design techniques in the next decade.

2. Maximize bandwidth per cost for both local and global traffic patterns. Future applications would require high bandwidth (BW) for both global and local communication. Thus, enhancing BW/\$ would be a key focus and will be guided by technology and application characteristics. This would require development of appropriate models for understanding the interplay between network architectures, workload, and technology artifacts.
3. Reliability, including error detection and resilience. Transient/soft error rates and hard errors due to process variation will continue to increase with device scaling, and thus, will be major obstacles for designing reliable Exascale systems due to dramatic increase in the number of undetected errors. Thus, cost-effective error detection techniques and other fault-tolerant mechanisms like light-weight fast recovery would be essential for satisfying the stringent reliability requirements.
4. Optimal topology. The hierarchical network would require a good balance between on-chip and off-chip topologies, which will again be influenced by the integration of optics, 3D integrated stacking, and, possibly, Carbon Nano Tubes (CNTs) with silicon, resulting in a heterogeneous architecture. Without a cohesive design effort between device/circuit researchers, architects, and software/application experts, design of such a grand-scale architecture may run into several problems. While it may be difficult to find an optimal topology for a wide variety of applications, the target architecture should be flexible for deployment, adaptable for application mapping, and should possess network discovery capability.
5. Quality of Service (QoS) and congestion control. Instead of designing a network for high peak BW, the objective should be to provide good/consistent performance for difficult permutations. This would require support for adaptive routing in a hierarchical network for providing congestion control and QoS assurance. In addition, the network architecture should facilitate task migration in the presence of failures and performance isolation between different partitions. The working group felt that without new investments in these areas, novel ideas for QoS and congestion control would not make it into interconnect products.
6. Minimize latency for small messages. Small messages (typically less than 3KB) constitute a major fraction of many HPC applications, and thus, minimizing latency for small messages to about 1 microsecond end-to-end is essential for improved performance. This would cover most of the collective communications including reductions, global interrupts, and clock distribution. Although, user-level communication and efficient NIC design are critical for this purpose, reducing the queuing time and transfer time in the network would help small messages. Thus, topology and micro-architecture enhancements would play a role in this optimization. However, providing these facilities in routers and NICs would require investment.
7. Maximize efficiency for small messages. In addition to minimizing the latency of small messages, the network should support high message throughput in the range of 100 million messages/s. This would require rethinking of the current design trend of supporting larger flits in the micro-architecture, and inclusion of other energy-efficient architectural features.
8. Sustainability and transfer to/leverage from commodity technology. Since most of the HPC market will continue to rely on commodity products, a key to long-term HPC advancement would lie in utilizing the commodity products if possible and facilitating the transfer of plausible designs to commercial platforms.
9. Performance monitoring counters, debugging, and diagnosis. Current processors have several hardware counters, which are useful for performance monitoring, optimization and debugging. Including such counters in the interconnect hardware would also help in performance and power optimizations at

different granularity. Interconnect designs have avoided including such counters, but this should be considered as a mainstream design practice in the future. This should provide useful insight about many aspects of network behavior. The nature and placement of such counters, attribution of measurements to threads, and measurement techniques for these counters is an open area of research.

10. Related challenges that are closely tied with topology and routing. The Exascale architecture design needs coordinated effort in several fronts. For example, an accurate and efficient network simulator and appropriate tools are necessary to evaluate various design options for satisfying the power, performance and reliability requirements. This needs integration of a performance simulator/emulator with power measurement and fault injection tools. The design and placement of NICs are also crucial in optimizing the message overheads. All these issues are further complicated by the lack of a good set of representative Exascale workloads for driving the performance study. Thus, identification of such workloads is critical to such an ambitious endeavor.

In conclusion, the design of an Exascale platform is a very ambitious undertaking that needs innovative research in several areas of networking, and close interactions between device, circuit, architecture, software, and application levels. It needs development of appropriate tools for analyzing power-performance-reliability-cost trade-offs. Many of these efforts are challenging and will require significant, long-term research and development.

## 4 TECHNICAL THRUST: PROCESSOR NETWORK INTERFACE

---

**Chairs: DK Panda, Keith Underwood**

In recent years, commodity microprocessors coupled together with a high performance interconnect have dominated the landscape of HPC computing. This has driven the rise of a wide variety of network interface designs as a bridge between the native microprocessor interfaces and the high performance router interfaces. Over the last decade, these devices have ranged from little more than packetizing engines (e.g., IBM BG/L) to simple DMA engines (e.g., ASCI Red and IBM BG/P) to complex DMA engines (e.g., InfiniBand adapters) to multicore chips that offload many of the MPI semantics (e.g., Quadrics Elan5).

As part of our premise, this group made two key assumptions. Foremost, given that alternative programming models are not well supported on many current (2008) hardware platforms, the combination of inertia in application development and the good match of MPI (and hybrid MPI + OpenMP) semantics to some application needs will guarantee that MPI-based applications will still be prevalent when we reach Exascale computing. Second, given the inherent performance limitations of the MPI interface for certain types of operations, there will be applications that require some derivative of a partitioned global address space (PGAS) programming model.

### 4.1 Status

There is a broad diversity of capabilities in modern network interfaces. Here, we summarize the capabilities that are commercially available, including both their performance characteristics and the unique features that are available.

1. Bandwidth, Latency, Message Rate. Today, peak network data rates range from 2 to 4 GB/s, MPI-level latencies have hit a plateau at approximately 1 microsecond, and messaging rates are reaching over 10 Million Msgs/sec. Most existing networks exhibit these thresholds.
2. Offload. Offload of many message passing semantics was identified as a challenge at the Exascale (see challenge 3 below) due to its ability to minimize cache pollution and specialize the architecture to the task; however, today, it is a feature that is often debated in the high performance networking circles – a situation that is apparent from the current state of the market. Today, some networks have virtually no offload

capabilities at all (e.g., IBM’s BlueGene/L). At the other extreme, the Quadrics Elan5 interconnect will have 8 specialized cores to provide high performance offload of MPI tag matching and a rendezvous protocol implementation at the network interface. In between, we find adapters from Myrinet, which offload a mixture of small processing tasks to facilitate aspects of MPI processing.

3. Independent Progress. One of the features that is also extensively debated in the community is the ability of a network to deliver independent progress. Independent progress is defined as the ability of the network to match incoming messages to posted receives and complete message transfers without the application being in the MPI library. This feature is supported by systems like Quadrics Elan5 by offloading the MPI tag matching. Other systems provide independent progress by using a progress thread, and still others (like the InfiniPath/TrueScale NIC) support this feature through interrupt driven message processing.
4. Collectives and Atomics. Collective operations and atomic operations (a class of synchronization operations) were also identified as a challenge (challenges 5 and 6). In today’s systems, BG/L has a set of separate networks for handling collective communications efficiently. Elan5 supports atomic operations along with hardware support for barrier and multicast operations in the network routers. In the network interface, InfiniBand supports atomic operations and also provides hardware-based multicast support for collectives. Similarly, Myrinet adapters support a variety of atomic operations. These efforts provide a good baseline for future innovations to support the collective and synchronization needs at the Exascale.
5. Considering the Future. There is relatively little public information on the roadmaps of the major providers of advanced networks; however, there are trends that can be observed. As silicon resources become more abundant, there is a tendency to improve the hardware support on the network interface for the native programming interface. Unfortunately, in many cases, this support is constrained by the native programming API, as many APIs do not support the semantics needed to offload some important pieces of the MPI processing semantics. This is particularly common for many RDMA interfaces, as they typically do not support tag matching, even if they do support a send/receive interface. In other cases, such as IBM’s BlueGene/L and BlueGene/P, the network interface has been integrated with the processors into a single system-on-chip (SOC) configuration. The SOC trend is encouraging for the biggest challenge identified (challenge 1: on-chip NI + off-chip NI integration), but it places the network interface in competition with the processing logic for silicon resources. In both BG/L and BG/P, this led to relatively little logic dedicated to the network interface but limited the in-the-field upgradability and serviceability, which may be important in Exascale systems.

## 4.2 Challenges

The network interface working group defined eight challenges that network interfaces will face when approaching the Exascale. The summary of the eight challenges is presented in the table below.

Challenge	Risk	Impact
1. On-chip NI + Off-chip NI Integration	High	High
2. Light-weight communication protocol	High	Medium
3. Enhanced NIC design	High	High
4. End-to-End Reliability Support	Medium	Medium
5. Collectives	High	Low
6. Fine-grain synchronization	High	Medium
7. Connection Management Scalability	High	High
8. Converged Network Interface	Medium	Medium

1. On-chip NI + Off-chip NI Integration. The single biggest challenge identified as networks move toward supporting Exascale systems is the integration of the network interface with the processor core. In a perfect world, the network interface would provide a seamless integration from the CPU’s network-on-chip to the

system area network – the semantics of communicating between local and remote cores would be the same, the network interface would be treated as a first class citizen, and one programming model would be sufficient for all parallelism. In reality, we are far from the ideal world. Network interfaces are currently treated as I/O devices, which carries incredible amounts of overhead. They are not given first-class access into the socket’s coherency protocol. On-chip networks and communication semantics are oriented around cache coherency, and it is impractical to extend cache coherency across the system. Bridging the gap between where we are and where the group felt we truly need to be will be an expensive proposition that will require the cooperation of the people building the processor for the system. This cooperation was believed to be a critical step to achieve efficient Exascale systems. Fortunately, there were some intermediate steps that were proposed along the way. First, the processor designs can become more cognizant of the needs of the network interface. The current I/O based memory semantics (uncacheable and write-combining memory spaces) that are used to communicate with the network interface are two decades old and do not capture the performance characteristics or the semantics required by the network interface. Second, the network interfaces should have the ability to inject data into caches. This comes in two forms. The most requested form was simply to update the cache if the data that the network interface was writing was already in the cache. The other form would allow the network interface to selectively choose some data to inject directly into the processors cache – even if it was not already present. Third, create a single synchronization method that the local cores could use efficiently, but that the network interface could extend across the system to involve all cores in the system. The group felt that research and development along this direction is the single most important area to achieve systems with Exascale-level performance.

2. Light-weight communication protocol. An unfortunate barrier in the current high performance networking market is the fragmented nature of the lowest-level network APIs. Generally speaking, these APIs are vendor specific implementations, and not portable across NICs. While this is in some way convenient for the vendors, it greatly complicates the implementations of MPI, PGAS, and SHMEM over these networks. Worse, many of the existing APIs have a large semantic mismatch with either MPI or PGAS. For example, typical VERBS implementations with the OpenFabrics stack for InfiniBand and 10GigE/iWARP lack any semantic support for offloading the MPI matching semantics. In contrast, the Portals 3.3 API has excellent support for matching semantics, but unnecessarily imposes those matching semantics on PGAS libraries. Other APIs, such as the one for Quadrics Elan adapters, limit the “high performance” API support to processes within a job, and limit the interaction with other services (e.g., a file server). What is needed is a single, light-weight communication interface that contains elements that semantically match the requirements of MPI and PGAS programming models and would be relatively easy to support in hardware. This interface differs from what was developed by the MPI Forum in that it is not meant to be exposed to the end-user. The group felt that research and development in this area is of significant importance for designing Exascale systems and thus, needs additional investment.
3. Enhanced NIC design. Although there are many network interface designs available, there were numerous issues raised that must be addressed to reach the Exascale. The most obvious desire was for better hardware support of the most common network API semantics – including MPI and PGAS programming models. It was proposed that the network interface could evolve to have substantially better processing engines that would provide increased MPI and PGAS message rates while simultaneously minimizing the cache pollution experienced by the microprocessor core when it is forced to handle all MPI and PGAS processing. Essentially, the challenge was to build hardware to directly support the features designed into the lightweight communication protocol proposed as challenge 2. In following on to challenge 1, it was stated that the network interface must evolve to take advantage of the enhanced integration proposed as challenge 1. This includes such things as maintaining coherence with the core microprocessor. It also includes being better integrated into the cache hierarchy and leveraging cache injection when the data being

updated is already in the processor cache. The probability that these capabilities would *not* be available, and would therefore have a critical impact on the ability to build an Exascale system was very high.

4. End-to-End Reliability Support. End-to-end reliability was viewed by many as a critical issue for Exascale systems. While it fell behind several other items in the list of challenges (mostly because an infinitely reliable system is not useful if it does perform well), it was seen by many as a necessary building block to achieve the system level reliability needed by Exascale systems. One of the key challenges acknowledged by the group was the need to minimize the network state information required for supporting end-to-end reliability while increasing the support for network features like adaptive routing. There were two notable viewpoints opposing the need for end-to-end reliability support at the network interface. First, some asserted that the software would be required to treat the system as unreliable, and so this issue could be ignored. That viewpoint was generally opposed as the majority argued that some significant level of reliability would be needed to enable rational software solutions to other types of failures. Second, others asserted that end-to-end reliability would *not* be necessary, as the routers could deliver adequate reliability to prevent the need for an end-to-end solution; thus, the impact at the Exascale level was rated as medium. Although several network interfaces already include some level of end-to-end reliability support, it was not clear as to what level of support would be sufficient at the Exascale. Furthermore, it was noted that dealing with emerging features like adaptive routing while still providing end-to-end reliability support would be relatively difficult. Thus, it was not clear whether the evolution of existing networks would continue to improve support for end-to-end reliability; thus, this was rated as a medium.
5. Collectives. Collective operations are a prominent issue in HPC systems. They typically scale as  $\log(N)$  of the system size, which makes them an area of extensive concern when the system scale is projected to grow to extreme levels (millions of cores). The general sentiment was that collectives must improve, but there were mixed feelings about whether this required additional hardware support (offload) or whether it could be handled entirely in software through improved algorithms, just improved implementations, or new features, such as non-blocking collectives. The observation was made that many of the “collective problems” seen on modern systems were eventually traced back to operating system (OS) noise issues or application load imbalance issues. Thus, in the end, it was agreed that it was *desirable* to have offload for collective operations, but it was viewed as relatively low impact on Exascale systems.
6. Fine-Grain Synchronization. The issue of fine-grain synchronization has been a contentious point in HPC for many years; however, unlike collective operations, it has never been truly required for the MPI programming model. As the programming model evolves to enable more fine-grained communication and, therefore, a more fine-grained division of the work, fine-grain synchronization will become an increasingly important issue. Unfortunately, this is another area where HPC is seen as somewhat diverging from the needs of the broadest base of commodity systems – at least, in the near term. In many ways, fine-grain synchronization support is like the support of collectives and was given an approximately equal weighting with collectives. Like collectives, it is relatively easy to add to an existing network interface design as an extra feature, and, therefore, should be low cost. It was rated as having a higher impact on Exascale systems than collectives, because it was believed that the programming model must evolve to make the programming of Exascale systems practical. Fine-grain synchronization primitives do have some unique features that do make them somewhat harder than collective operations to deliver. They require a “good” interface between the processor and the network interface and they require some additional handling of out-of-order messages at the NIC.
7. Connection Management Scalability. An Exascale system is expected to contain on the order of 30,000 to 100,000 nodes with millions of cores. Given that MPI is likely to still be common in the application code base in this timeframe, the system must support the “fully connected” semantics of MPI. Unfortunately, there was concern that the future of applications will involve denser traffic patterns between the processes

in a job as the type of science evolves. There was a strong sentiment in the working group that traditional connection-oriented protocols would not scale to that size. Unfortunately, the issue of extreme scale with extreme connectivity among processes is not shared in most commodity environments. Thus, it is highly unlikely that commodity networks will solve this issue. Furthermore, the impact on Exascale is critical. An Exascale resource is not particularly useful if the entire resource cannot be deployed on a single capability application. Because it would be disruptive to many aspects of the current infrastructure, this would be much more disruptive than simply adding synchronization or collective support to the interface.

8. Converged Network Interface. The notion of a converged network interface is simply that it is desirable to have one network support all of the traffic for a supercomputer. While this was not defined to include traffic for the RAS (reliability, availability, serviceability) system, it was intended to include traffic for message passing, the file system, standard I/O, job launch, and any other traffic that was required by the applications or services running on the system. The notion was that a good network should not require the cost or complexity of having multiple network interfaces on a node. Given the other challenges faced to deliver a network interface for Exascale computing, this challenge would definitely be defined as insuring that the custom network interface for the Exascale computer would support different types of traffic (message passing, file system, standard I/O, etc). The ranking of this area was reduced because there was some debate as to whether it was a real challenge.

## 5 TECHNICAL THRUST: SIMULATION AND PERFORMANCE PREDICTION

---

**Chairs: Curtis Janssen, Sudhakar Yalamanchili**

Accurate and reliable performance prediction for applications running on large-scale architectures will be critical to the design of machines and algorithms that can deliver Exascale performance at a reasonable cost. An architect currently faces many choices in the design of a machine including characteristics, such as the network topology, switch radix, link bandwidth, number of cores per node, memory per node, not to mention entirely new device and network interface technologies. First, the sheer scale of these future systems present unprecedented modeling and performance prediction challenges. Second, fundamental system characteristics continue along trend lines which point to new, critical trade-offs necessary to extract high performance. For example, the number of pins/core and memory/core are dropping while link bandwidths are increasing. Architects will be forced to consider a variety of new memory system organizations, and options to hide latency and bandwidth deficiencies. Applications developers will have to revisit algorithms to optimize performance in this new design space.

In the presence of such trends, it is clear that modern applications in their current form (i.e., optimized for current overheads) are unlikely to be able to obtain the desired 1000-fold speedup on an Exascale machine. Therefore, algorithm and application developers will need early and accurate assessments of the impact key interconnection network attributes. Furthermore, non-network components such as core, cache, memory and I/O subsystems exhibit subtle and complex interactions with the interconnection network that can significantly impact performance. As a result, the performance prediction and simulation environment must include the ability to interact with, and reflect the influence of, detailed core, memory, and I/O system models. Such an approach is necessary to ensure that the application development process can both influence the design of future machines as well as prepare a core of Exascale-ready applications and algorithms in advance of a system's arrival.

The leading challenges facing the performance prediction and modeling for Exascale systems can be understood in the context of three constituencies of users of the tools. The first is the pool of application developers who will rely on these tools for exploring and developing algorithmic solutions and subsequently tuning these solutions for specific target configurations. This community emphasizes the ability to accurately sweep large machine parameter

spaces and explore subtle but critical interactions in the performance and efficiency of the interconnection network. The second constituency includes those involved in procurement activities. These tools provide neutral environments for comparing and contrasting hardware/software procurement options. Finally, the tools are intended to enable and catalyze research efforts amongst the third constituency which include both industrial and academic system design communities.

Performance prediction, in its most general form, includes the use of both analytic models and simulation models co-existing in the same environment. Thus, in the following we do not distinguish the two and use the term simulation in its most general sense to include all forms of models – analytic, discrete event, time-stepped, functional, etc.

## 5.1 Status

There are currently many simulators for covering processor core, network and memory subsystems, ranging in quality and availability from university research projects to production quality simulators used by chip and system vendors. The current state-of-the-practice could be described as “islands of simulation artifacts,” where each island is isolated both in terms of functionality (each simulator targets a particular hardware component) and in terms of interoperability (each simulator is a stand-alone piece of software). These artifacts are necessarily biased towards the individual components being modeled. Consequently, there is no neutral ground that can be used by application developers, systems researchers, system developers, or procurement teams to accurately model, analyze, and understand the impact of technology, architecture, and system decisions on application performance. Further, the integration of these specific tools to create system-level models is an ad-hoc, tedious, labor intensive, and error prone process.

As a practical matter, the execution time required by current point simulators makes it infeasible to model and understand the behavior and performance of large, complex applications. Today, processor simulators can execute in the range of 1000 to 200,000 simulated instructions per second, depending upon the complexity of the architecture being simulated. The feasible number of cores that can be currently simulated is around 64-128. Part of the problem is that modern system software, such as the operating system and BIOS, do not support larger numbers of cores, necessitating either developing solutions to these problems first or abstracting their behaviors. A major part of the problem is that the number of simulator cycles will have to keep pace with the number of cycles in the modeled systems necessitating the need for parallel simulation technologies. There have been some early research prototypes of parallel simulators, but the application to multicore/multiprocessor systems has been limited and of small scale.

Storage system simulators lag interconnect and processor simulators while datasets continue to grow with the science. Furthermore, long run times will require applications to frequently checkpoint their state. Consequently, storage systems play a vital role in Exascale machines particularly if the MTBF is relatively low. Storage simulators are critical to prevent the I/O subsystem from being a bottleneck by helping system architects understand the effects of storage area network (SAN) design choices and the interactions of application communication patterns and SAN communications.

Finally, a very important pragmatic issue is that modeling tools and simulators are largely inaccessible and opaque to application developers. The simulators are typically complex to use and may require access to specialized hardware, e.g., FPGA-based acceleration. Further, the loci of analysis centers around traditional hardware-centric metrics, such as instruction throughput and memory bandwidth. Correlating application characteristics with machine behavior and providing tools to explore these interactions are largely lacking while the complexity of such interactions and their criticality will grow non-linearly with system size. Simulation and modeling tools and interfaces are not typically developed from the perspective of application developers. Rather they are typically developed by, and from the perspective of, hardware and system software designers.

## 5.2 Challenges

The top 10 challenges are organized into a few classes below and presented in decreasing order of importance.

**Model/Simulator Construction:** The single most important impediment to modeling and simulation is the cost (in time and effort) to construct an accurate model of a target system. Based on the state of the practice it is infeasible to construct a customized detailed model for each target system variant or point in the design space. Consequently, critical to successful future design is addressing the following two challenges:

1. Technology for Composable Simulators: System simulators are composed from existing and future detailed point simulators. Composition must accommodate different timing models, degrees of fidelity, and event models. For example, both clock-tick-based and event-based models must coexist and even directly inter-operate.
2. Methodologies for Constructing Simulators: With the availability of precise compositional semantics, we clearly see the need for methodologies to (automatically) construct simulators for specific target system configurations largely by composing alternative point simulators at varying individual levels of abstraction. These methodologies intellectually have the flavor of system synthesis and may share similar solution philosophies.

**Accuracy:** Broadly speaking, accuracy can be either relative, where the simulator predicts the relative utility of design alternatives, or absolute, where properties such as power dissipation or time-to-solution are predicted to within a certain percent of the value that would be observed for an actual machine. Achieving accuracy requires addressing the next two challenges:

3. Model Calibration: The simulator components must be validated and calibrated. For example, calibration may be by comparison to measurements on existing actual hardware or prototypes of critical pieces.
4. Simulator Calibration: As the system simulator is now pushed beyond the calibration points of individual tools, we need methodologies for calibrating individual simulator models and bounding the accuracy of the simulations. Thus, a simulator composed to represent a specific point in the architecture design space is calibrated to provide a high level of confidence in the results.

**Performance:** The simulator must be able to simulate large applications running on large machines in a practical amount of time. The ability to do so necessitates addressing three main challenges:

5. Parallelism: Given the scale of next generation systems, we must use parallel machines to simulate parallel machines.
6. Multi-scale: This challenge refers to the ability to accurately co-simulate subsystem components modeled at different (possibly widely) levels of fidelity ranging from simple analytic models to cycle level time-stepped simulation.
7. Hardware Acceleration: Using FPGAs or some customized support can lend multiple orders of magnitude speedup of certain classes of simulations. Harnessing this technology into a multi-scale simulation would significantly expand the scale of systems that can be accurately modeled.

**Power and thermal models:** A key issue with an Exascale machine will be the high power requirement, which will be a major component of the overall machine cost. The simulator must be able to model the power requirements as well as thermal dissipation requirements:

8. Power and thermal models: Technology-driven power and thermal models should be an integral part of the environment. While chip scale models exist, models at the scale of tens of millions of cores, terabytes of memory, exabytes of disk currently do not exist.

*Ease of use*: The potential users of the simulator include not just designers, but procurement teams, application development teams, and researchers. It is a major challenge and a deviation from the current state of the practice to provide a simulator that is driven by needs of the first two audiences. The specific challenges are the following:

9. Visualization and Automation: This challenge goes directly to productivity of application developers and system developers. Automation in particular refers to tools for specifying simulator models, experiments, data analysis needs, and deploying them on a parallel infrastructure.
10. Documentation: Apart from traditional documentation, the specification of methodologies for model construction, model calibration, model development, and performance evaluation will have a major impact on productivity. The scale of these models makes this a challenging task whose solution has a direct impact on productivity and in many instances, feasibility. Construction of accurate and reliable models of Exascale systems will border in the complexity of assembling modern systems and this require a similar level of methodology support.

Table 2 summarizes the preceding challenges in the workshop format with the following deviations. First, “Impact” is interpreted as the degree to which the performance prediction tools will impact the cost, efficiency, and effectiveness of Exascale machines rather than as a barrier to feasibility.

**Table 2: Technical challenges facing developers of simulators for Exascale architectures. The risk that the challenge will not be met without Exascale- specific intervention and the impact of overcoming the challenge are each scored as high, medium, or low. Costs are annual costs for three years.**

Challenge	Risk			Impact		
	H	M	L	H	M	L
<b>Building a validated useful simulator</b>	X			X		
Technology for composable simulators	X			X		
Methodologies for constructing simulators	X			X		
<b>Accuracy</b>	X			X		
Model calibration	X			X		
Simulator calibration		X			X	
<b>Performance</b>	X				X	
Parallelism	X				X	
Multi-scale		X			X	
Hardware acceleration	X				X	
<b>Power and thermal models</b>			X	X		
<b>Ease of use</b>		X			X	
Visualization and automation		X			X	
Documentation and deployability	X			X		

### 5.3 Non-technical Strategic Issues

- *Demonstrable value*: As with modeling and simulation in any other field, the simulation tools must demonstrate their value as a design tool for Exascale machines. Validation of the simulator will play a

critical role in this. Strategically timed and formulated pilot projects with application developers can address this.

- **Integration with other Exascale activities:** The simulator work must be integrated with other Exascale design activities to have impact. These activities include memory and interconnect design and algorithm development and should be during the programs rather than after the fact.
- **Neutral simulator environment:** For the simulator to be most useful, a large community, including vendors and academia, must develop interoperable and widely available subsystem simulation modules. Procurement and funding incentives could be used to help promote this environment.
- **Adoption:** The simulator efforts must be surrounded by a healthy community of users that apply the simulator to a variety of architecture design problems. By funding early adopters of the simulator, we will obtain vital feedback, catalyze a community of users, seed new and otherwise infeasible research explorations, and will ultimately have a stronger simulation package

## 6 TECHNICAL THRUST: DEVICE TECHNOLOGY

**Chairs: Keren Bergman, Azita Emami**

Underlying interconnect device technologies determine fundamental limits to performance and to scalability of the overall system infrastructure. As described in the report introduction, even aggressive projections for conventional CMOS technologies fall well short of the needs of interconnection networks in the envisioned HPC systems. The Device Technologies Working Group (DTWG) aimed to evaluate the key challenges and to identify the most promising emerging technologies and associated research thrusts that could bridge this gap and deliver scalable performance to future HPC systems.

To assess the diverse range of emerging interconnect technologies, the DTWG initially established a common set of metrics for comparing them to conventional CMOS circuits. From this discussion it was clear that the most important metric for any emerging device technology is the power dissipation associated with transferring high-bandwidth data. This can be concisely summarized as the *energy per bit* associated with moving data within or across all levels of the system interconnect hierarchy. Two additional key measures critical to evaluating the benefits of potential new interconnect technologies were (1) the cost per bit per second, which encompasses the total cost of adopting a new interconnect technology; and (2) the bandwidth density, which may be measured in bits per second per chip-area, or similar units depending on the packaging and hierarchical level of the interconnect.

### 6.1 Emerging Interconnect Technologies

The DTWG assessed the most promising emerging interconnect devices: those with the potential, given sufficient investment, to deliver the required performance needed for future capability HPC systems as defined in the workshop. Aggressive scaling of CMOS interconnects was taken as the benchmark. The three clear leading emerging interconnect technologies were: silicon photonics, proximity communications (capacitive and inductive coupling), and MEMS. In addition, 3-D integration, which may be categorized as a packaging technology, was identified as a key enabler for the successful insertion of these heterogeneous device technologies. Several additional interconnect technologies, including carbon nanotubes, free-space optics, and RF communications, were deemed to be speculative and outside of the system roadmap considered by the workshop.

**Table 3: Energy.**

Name	Maximum Distance	Energy (pJ/bit)	Bandwidth (Tb/s)
On-chip global	2 cm	0.5-1	100

On-module	10 cm	2-5	1-10 (I/O of each chip)
On-board	40 cm	5-10	1-5
Intra-cabinet	1 m	10-15	1-5
Inter-cabinet	50 m	20-30	5-10 (optical)

Because different technologies may be applied across a system’s interconnect hierarchy, the DTWG defined several interconnect domains to assess the potential impact and challenges of the device technologies at multiple scales. The associated energy/bit expected from CMOS scaling was determined for benchmarking. These interconnect domains, electrical signaling benchmark energies, and the projected bandwidths were defined in Table 3.

## 6.2 Potential Impact

The emerging device technologies have the potential to eliminate the two most critical off-chip communications boundaries: intra-node processor-memory and inter-node interfaces. With conventional electronic signaling these two boundaries represent the most severe bottlenecks to scaling high-bandwidth interconnects, primarily due to energy consumption, and are, therefore, leading to increasingly imbalanced systems designs. By addressing these critical off-chip boundaries, emerging technologies, such as silicon photonics, proximity communication, and MEMS have the potential to deliver energy efficient interconnect with system wide communications bandwidths that are seamless on a chip, between a processor and memory, and inter-node, enabling scalable and balanced system architectures. For optical interconnects, the transparent switching capabilities of MEMS and silicon photonics can potentially deliver significant additional functionality via dynamic, energy efficient routing, and topology reconfigurability.

## 6.3 Challenges

The DTWG mapped the broad technologies challenges across three scales: the device level, the circuit level, and the system level. At the device level, the major challenge of chip-scale photonics was realizing a thermally stable, high-yield, and scalable integration for a production environment. Silicon photonics is still an immature technology platform pursued primarily in research. Significant challenges remain to realize this technology in HPC systems. These challenges include building innovative block components such as novel WDM, modulator, switch, and receiver devices with performance characteristics (insertion loss, density, speed, etc.) that supersede today’s research by an order of magnitude or more. Additional challenges were identified at the circuit and package level. These included reliable and manufacturable integration of analog CMOS devices with active photonic devices, and achieving high density electro/optic I/O interfaces in energy-efficient packaging. The mixed-signal transceiver circuitries need to be implemented in highly scaled technologies to support high data rates and consume very low power. Issues such as process and temperature variations, clocking and reliability continue to be major challenges and are similar to electrical signaling. At the system level, packaging was viewed as a key challenge to the successful insertion of photonic interconnects in large scale HPC. The realizations of optoelectronic interconnect subsystems with a high degree of uniformity and reliability was viewed as critically important to future HPC systems.

## 6.4 Maturity-Impact-Cost Analysis

To assess emerging interconnect technologies and their associated challenges for realizing Exascale HPC, the DTWG developed a matrix (shown below) which mapped technical challenges against their level of maturity, potential impact/reward, and required investment/cost. The technical challenges were grouped under three areas where emerging interconnect technologies will most impact future HPC systems: the two critical off-chip

boundaries of the intra-node processor-memory interface and the inter-node interface, and in providing dynamic interconnect reconfigurability and energy efficient routing via switching.

**Table 4: Emerging Technologies.**

Technical	Maturity			Impact/Rewards		
	High	Med	Low	High	Med	Low
<b>Intra-node processor-memory interface</b>						
CMOS photonics			x	x		
3-D stacking (parallel and orthogonal)		x		x		
proximity Comm.		x		x		
<b>Inter-node</b>						
dense integrated-photonics to fiber interface		x		x		
CMOS photonics			x	x		
Packaging (E&O)		x		x		
<b>Switching</b>						
optical switching		x			x	
MEMS	x				x	
hybrid (electrical/optical)		x		?	x	

## 7 APPENDIX – WORKSHOP AGENDA

---

### Day 1 – Monday, July 21

Time	Activity	Speakers/Chair
7:45	Continental Breakfast	
8:15	Welcome Introductions Goals DOE Roadmap, including Exascale goals	IAA Directors DOE Office of Science NNSA Workshop co-chairs
8:45	Panel: Application/Algorithms Requirements for Interconnects	John Shalf
10:00	Break	
10:30	Working Groups introduction	Scott Hemmert Jeffrey Vetter
10:45	Working Groups 1 and 2	WG Chairs
12:00	Lunch provided by workshop	
1:00	HEC Interconnects: Retrospective and Trends	William Dally
1:45	Panel: Programming Models Implications for Interconnects	Ron Brightwell
3:00	Break	
3:30	Panel: Performance Prediction and Simulation	Sudhakar Yalamanchili
4:45	Working Groups 3 and 4	WG Chairs
6:00	Adjourn	

### Day 2 – Tuesday, July 22

Time	Activity	Speakers/Chair
7:45	Continental Breakfast	
8:15	Panel: Future Disruptions in Interconnect Device Technology	Keren Bergman
9:30	Break	
10:00	Working Groups 1 and 3	WG Chairs
11:30	Lunch provided by workshop	
12:30	Working Groups 2 and 4	WG Chairs
2:00	Break	
2:30	Working Groups 1 and 2: Outbriefs	WG Chairs
3:00	Working Groups 3 and 4: Outbriefs	WG Chairs
3:30	Closing comments and action items	Scott Hemmert Jeffrey Vetter
3:45	Adjourn	

## 8 APPENDIX – ATTENDEES

---

<b>Last Name</b>	<b>First Name</b>	<b>Organization</b>
Ahn	Jung Ho	HP Labs
Albonesi	Dave	Cornell University
Ang	James	Sandia National Laboratories
Asanovic	Krste	University of California at Berkeley
Balfour	James	Stanford University
Barrett	Brian	Sandia National Laboratories
Becker	Daniel	Stanford University
Bergman	Keren	Columbia University
Binkert	Nathan	HP Labs
Bisant	David	Lab for Physical Sciences
Bloch	Gil	Mellanox Technologies
Brian	Towles	D. E. Shaw Research
Brightwell	Ron	Sandia National Labs
Camp	William	Intel Corp
Carlson	Bill	Center for Computing Sciences
Carter	Nicholas	Intel Corporation
Chen	James	Stanford University
Chiou	Derek	University of Texas at Austin
Culhane	Candace	Dept of Defense
Dally	William	Stanford University
Das	Chita	NSF/Pennsylvania State University
Dickman	Lloyd	QLogic
Dosanjh	Sudip	Sandia National Laboratories, IAA
Emami	Azita	Caltech
Fields	parks	Los Alamos National Lab
Friedman	Eby	University of Rochester
Geist	Al	Oak Ridge National Laboratory
Geoffray	Patrick	Myricom
Goh	Eng Lim	SGI
Grzybowski	Richard	Corning Incorporated
Hargrove	Paul	LBNL
Harrod	William	DARPA/IPTO
Hemmert	Scott	Sandia National Laboratories
Hempfling	Sherry	ORNL / National Center for Computational Sciences
Heroux	Michael	Sandia National Laboratories
Hiller	Jon	Science and Technology Associates, Inc. (STA)
Ho	Ron	Sun Microsystems Research Labs
Hoang	Thuc	NNSA Advanced Simulation & Computing
Holland	Charlie	DARPA/IPTO
Janssen	Curtis	Sandia National Laboratories
Jiang	Nan	Stanford University
Johnson	Fred	ASCR
Kim	John	Northwestern University
Leiningner	Matt	LLNL
Martinez	Jose	Cornell University

Macaluso	Antoinette	SAIC
Mayhew	David	AMD
McLaren	Moray	HP Labs
Michelogiannakis	Georgios	Stanford University
Miller	David	Stanford University
Oliker	Lenny	LBNL
Pakin	Scott	Los Alamos National Laboratory
Panda	Dhabaleswar K (DK)	The Ohio State University, Dept. of Computer Science & Engg.
Pant	Avneesh	National Center for Supercomputing Applications
Parker	Mike	Cray, Inc
Pinkston	Timothy	National Science Foundation
Rodrigues	Arun	Sandia National Labs
Roth	Philip	Oak Ridge National Laboratory
Schenfeld	Eugen	IBM T J Watson Research Center
Scott	Steve	Cray Inc.
Shainer	Gilad	Mellanox Technologies
Shalf	John	Lawrence Berkeley National Laboratory
Stunkel	Craig	IBM Research
Sugumar	Rabin	Sun Microsystems Inc
Thorson	Gregory	Silicon Graphics
Tipparaju	Vinod	Oak Ridge National Laboratory
Tomkins	James	Sandia National Laboratories
Underwood	Keith	Intel
Vetter	Jeffrey	ORNL and Georgia Tech
Vishkin	Uzi	University of Md Inst. for Adv Computer Studies (UMIACS)
Weisser	Deborah	Google
Winkler	Karl-Heinz	LANL
Yalamanchili	Sudhakar	Georgia Institute of Technology
Yates	Robert	DOE/NNSA NA-121.2
Yu	Weikuan	Oak Ridge National Laboratory

## REFERENCES

---

- [1] S.R. Alam, R.F. Barrett *et al.*, "An Evaluation of the ORNL XT3," *International Journal of High Performance Computing Applications*, 22(1):52-80, 2008.
- [2] W.W. Carlson, J.M. Draper *et al.*, "Introduction to UPC and language specification," Center for Computing Sciences, IDA, Bowie, MD, Technical Report CCS-TR-99-157, 1999.
- [3] R. Chandra, *Parallel programming in OpenMP*. San Francisco, CA: Morgan Kaufmann Publishers, 2001.
- [4] T.H. Dunigan, Jr., M.R. Fahey *et al.*, "Early Evaluation of the Cray X1," Proc. ACM/IEEE Conference on High Performance Networking and Computing (SC03), 2003.
- [5] T.H. Dunigan, Jr., J.S. Vetter *et al.*, "Performance Evaluation of the Cray X1 Distributed Shared Memory Architecture," *IEEE Micro*, 25(1):30-40, 2005.
- [6] T.H. Dunigan, Jr., J.S. Vetter, and P.H. Worley, "Performance Evaluation of the SGI Altix 3700," Proc. International Conf. Parallel Processing (ICPP), 2005.
- [7] J. Kim, W.J. Dally *et al.*, "Technology-Driven Highly-Scalable Dragonfly Topology," in *International Symposium on Computer Architecture (ISCA)*, 2008
- [8] P. Kogge, K. Bergman *et al.*, "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems," DARPA Information Processing Techniques Office 2008.
- [9] D. Kothe, "Science Prospects and Benefits with Exascale Computing," Oak Ridge National Laboratory ORNL/TM-2007/232, 2007.
- [10] R.W. Numrich and J. Reid, "Co-Array Fortran for parallel programming," *ACM SIGPLAN FORTRAN Forum*, 17(1998):1-31, 1998.
- [11] OpenMP, *OpenMP Reference*, <http://www.openmp.org>, 1999.
- [12] H. Simon, T. Zacharia, and R. Stevens, Eds., *Modeling and Simulation at the Exascale for Energy and the Environment*, 2007.
- [13] M. Snir, W.D. Gropp *et al.*, Eds., *MPI--the complete reference (2-volume set)*, 2nd ed. Cambridge, Mass.: MIT Press, 1998.
- [14] J. Tomkins, "Interconnects: A Buyers Point of View," Sandia National Laboratories ACS613-1407, 2007.
- [15] K. Underwood, "Challenges and Issues in Benchmarking MPI," *Recent Advances in Parallel Virtual Machine and Message Passing Interface: 13th European PVM/MPI Users' Group Meeting, Bonn, Germany, September 2006 Proceedings*:339-46, 2006.
- [16] P.H. Worley, S. Alam *et al.*, "Comparative Analysis of Interprocess Communication on the X1, XD1, and XT3," *Proc. Cray User Group Conference, USA*, 2005.