OPENFABRICS ALLIANCE

OFA Workshop 2023 Abstracts

Keynotes

The Next Decade of Networking: Challenges and Opportunities

Aditya Akella, University of Texas at Austin

Networking technology has undergone rapid transformations over the past 1-2 decades, and breakthroughs such as software-defined networking, network virtualization, and programmable switches have led to great improvement in edge, cloud, and data center infrastructure and applications. Networking continues to evolve rapidly, with transformative advances across the stack ushering in a new breed of applications with extreme demands running at the edge, the cloud, or across geo-distributed data centers. In this talk, I will describe some of these emerging advances – spanning network transport and congestion control, network switches and interface cards (NICs), and network software stacks – outlining the opportunities they offer and the open challenges that must be overcome to effectively support future applications.

UCIe™: Building an open ecosystem of chiplets for on-package innovations

Debendra Das Sharma, Intel Corp., UCIe Consortium

UCIe™ (Universal Chiplet Interconnect Express™) is an open specification that defines the interconnect between chiplets within a package, enabling an open chiplet ecosystem and ubiquitous interconnect at the package level. UCIe technology addresses the projected growing demands of compute, memory, storage, and connectivity across the entire compute continuum spanning cloud, edge, enterprise, 5G, automotive, high-performance computing, and hand-held segments. UCIe provides the ability to package dies from different sources, including different fabs, different designs, and different packaging technologies.

This presentation will introduce features in the UCIe 1.0 specification and explore how UCIe will enable users to easily mix and match chiplet components from a multi-vendor ecosystem for System-on-Chip (SoC) construction, including customized SoC.

Technical Sessions

“Hey CAI” - Conversational AI Enabled UserInterface for HPC Tools

Hari Subramoni, The Ohio State University; Dhabaleswar Panda, The Ohio State University; Pouya Kousha, The Ohio State University; Aamir Shafi, The Ohio State University

HPC system users depend on profiling and analysis tools to obtain insights into the performance of their applications and tweak them. The complexity of modern HPC systems have necessitated advances in the associated HPC tools making them equally complex with various advanced features and complex user interfaces. While these interfaces are extensive and detailed, they require a steep learning curve even for expert users making them harder to use for novice users. While users are intuitively able to express what they are looking for in words or text (e.g., show me the process transmitting maximum data), they find it hard to quickly adapt to, navigate, and use the interface of advanced HPC tools to obtain desired insights.

In this talk, we present the challenges associated with designing a conversational (speech/text) interface for HPC tools. We use state-of-the-art AI models for speech and text and adapt it for use in the HPC arena by retraining them on a new HPC dataset we create. We demonstrate that our proposed model, retrained with an HPC specific dataset, can deliver higher accuracy than the existing state-of-the-art pre-trained language models. We also create an interface to convert speech/text data to commands for HPC tools and show how users can utilize the proposed interface to gain insights quicker leading to better productivity. To the best of our knowledge, this is the first effort aimed at designing a conversational interface for HPC tools using state-of-the-art AI techniques to enhance the productivity of novice and advanced users alike.

Accelerating Scientific Computing Workloads with InfiniBand DPUs

Gilad Shainer, NVIDIA; Richard Graham, NVIDIA

Complex scientific workloads are placing enormous demands on supercomputing infrastructures. In addition, cloud providers hosting scientific workloads require their infrastructures to deliver bare-metal performance. All of these complexities require a best-in-class networking platform, that's programmable, and includes hardware acceleration engines to provide faster time to insight.

The session will discuss InfiniBand, In-Network Computing and the capabilities of the BlueField DPU to boost performance and drive scientific computing innovation.

Addressing Endpoint-induced Congestion in a Scale-Out Accelerator Domain

Timothy Chong, Intel Corp.; Venkata Krishnan, Intel Corp.

From a system standpoint, the domain size of interest for accelerator scaling is not several 1000s of nodes but a medium-scale domain that span a few 100s of nodes – the round-trip latencies in such a domain would typically be < 5µs. This is arguably considered the sweet spot for AI and data analytics workloads which do not scale well when the communication domain becomes exceedingly large. The accelerators also require significant bandwidth when attached to the network. Indeed, Ethernet bandwidths are continuing to increase by 4x-10x each generation, with 800Gbps on the horizon, and will provide sufficient bandwidths to the accelerators. However, Ethernet is inherently lossy and RDMA over Ethernet (RoCEv2) enforces an artificial "lossless" behavior with mechanisms such as PFC, which are not appropriate for lossy networks with huge bandwidths. Congestion handling is also poor in such a "lossless" model with congestion tree spread and flows getting victimized. At the same time, mechanisms such as TCP/IP congestion control mechanisms are very conservative, slow to react and suitable only for very large networks.

In our ongoing work, we have developed an RDMA mechanism over UDP/IP/Ethernet, one that is targeted for a lossy network without the use of PFC-like or link-level credit mechanisms. The end-to-end reliable transport mechanism is architected for an accelerator domain of a few 100s of nodes with highly reactive congestion avoidance/control mechanisms. The scheme has been implemented on an FPGA-based integrated networking/accelerator platform called COPA and currently achieves end-to-end reliable bandwidth of 200Gbps while keeping the FPGA resource usage to < 10% thereby enabling complex acceleration functions to be integrated. There are occasions where the FPGA is unable to process incoming packets due to a variety of factors such as host-FPGA interaction, memory bandwidth bottlenecks, host software overheads etc. In this presentation, we will describe how we address the endpoint-induced congestion problem. As part of this work, we have developed heuristics that allow the target endpoint to transparently modify the standard end-to-end ACK mechanism (without the need for additional control messages) such that the initiator endpoint would react to endpoint congestion quickly. Our scheme leads to better goodput, and fewer dropped packets when compared to conventional schemes.

Booting Your OS Across the NVMe® over Fabrics (NVMe-oF™) Transport – NVMe Boot Specification

Phil Cayton, Intel Corp., NVM Express

As large deployments become more common, our industry needs a standardized, multi-vendor solution to boot computer systems from OS images stored on NVMe® devices across a network. The newly published NVM Express® Boot Specification and ecosystem partnership with UEFI and ACPI enables this by leveraging the NVMe over Fabrics (NVMe-oF™) standard.

This talk is a dive into the details of the new specification and the design of an open-source prototype for booting over NVMe/TCP transport using a UEFI implementation.

Congestion Management for Multicast on ROCE v2

Christoph Lameter, Deustche Boerse AG

We are trying to move our infrastructure from Infiniband to Ethernet (ROCE v2). We want to preserve the way our application operates with RDMA and with the Fabric as much as possible. The RDMA subsystem supports transparent verbs operation on Ethernet with ROCE and with ROCE v2 even multicast is possible. We verified the operations of our software on Ethernet. However, we require Infiniband "backpressure" to slow down our multicast senders. This slowing down of the sender has been key to the reliability of our multicast system over the last decade or so. Here we cover the experiences with hardware, vendors and protocols to come up with a similar mechanism. Two approaches are the ECN congestion management as implemented by ROCE v2 and the Cisco Pause Frames.

Designing Networking Stacks for ML Frameworks

Raghu Raja, Enfabrica

This talk aims to provide a primer to Neural Networks and the ML framework ecosystem as it pertains to network architects, discusses the challenges of designing networking software and hardware for the fast-evolving ML models and applications, presents a case study we conducted and a prototype we built with NCCL to study some of these challenges on our platform, and concludes with some call to actions for the OFA community.

Diving Into the New Wave of Storage Management

Richelle Ahlvers, Intel Corp.

As the NVM Express® (NVMe®) family of specifications continue to develop, the corresponding Swordfish management capabilities are evolving: the SNIA Swordfish™ specification has expanded to include full NVMe and NVMe-oF™ enablement and alignment across DMTF™, NVMe, and SNIA for NVMe and NVMe-oF use cases.

If you haven't caught the new wave in storage management, it's time to dive in and catch up on the latest developments of the SNIA Swordfish specification. These include:

- Expanded support for NVMe and NVMe-oF Devices using the NVMe 2.0 family of specifications

- Managing Storage Fabrics

- Extending Storage Management into Composable Managed Infrastructure

This presentation provides an update on the latest NVMe-oF configuration and provisioning capabilities available through Swordfish, and an overview of the most recent work adding detailed implementation requirements for specific configurations, ensuring NVMe and NVMe-oF environments can be represented entirely in Swordfish and Redfish environments.

DPFS: DPU-powered File System Virtualization

Peter-Jan Gootzen, IBM Research Zurich; Jonas Pfefferle, IBM Research Zurich; Radu Stoica, IBM Research Zurich;

Animesh Trivedi, VU University Amsterdam

Today many cloud infrastructure services, such as storage and networking, run in the hypervisor on the host CPU alongside the tenants' virtual machines. Such an approach, although flexible and hardware agnostic, has several disadvantages. It requires taking CPU cycles away from the tenants to run the client-side drivers, provides poor isolation between the virtual tenants, and requires baremetal tenants to install, configure, and maintain additional drivers. State-of-art Data Processing Units (DPU), that support offload of advanced network and storage protocols, now provide an opportunity to move client-side drivers out of the host CPU. An offload approach reduces client CPU overhead, improves the isolation between the cloud tenants, and provides support for baremetal tenants.

In this talk, we propose to decouple the file system client from its backend implementation by virtualizing it with an off-the-shelf DPU using the Linux virtio-fs/FUSE framework. The decoupling allows us to offload the file system client execution to a DPU, which is managed and optimized by the cloud provider, while freeing the host CPU cycles. Our proposed framework, is 4.4× more CPU efficient per I/O, delivers comparable performance to a tenant with zero-configuration or modification to their host software stack, while allowing workload-specific backend optimizations.

High-Performance and Scalable Support for Big Data Stacks with MPI

Aamir Shafi, The Ohio State University; Dhabaleswar Panda, The Ohio State University; Jinghan Yao, The Ohio State University; Kinan Alattar, The Ohio State University

A key bottleneck faced by modern Big Data platforms, including Spark and Dask, is that they are not capable of exploiting high-speed and low-latency networks like InfiniBand, Omni-Path, Slingshot, and others. In the High Performance Computing (HPC) community, the Message Passing Interface (MPI) libraries are widely adopted to tackle this issue by executing scientific and engineering applications on parallel hardware connected via fast interconnect.

This talk provides a detailed overview of MPI4Spark and MPI4Dask that are enhanced versions of Spark and Dask frameworks, respectively. These stacks are capable of utilizing MPI for communication in a parallel and distributed setting on HPC systems.

MPI4Spark can launch the Spark ecosystem using MPI launchers to utilize MPI communication. It also maintains isolation for application execution on worker nodes by forking new processes using Dynamic Process Management (DPM). It bridges semantic differences between the event-driven communication in Spark compared to the application-driven communication engine in MPI.

MPI4Dask is an MPI-based custom Dask framework that is targeted for modern HPC clusters built with CPU and NVIDIA GPUs. MPI4Dask provides point-to-point asynchronous I/O communication coroutines, which are non-blocking concurrent operations defined using the async/await keywords from the Python's asyncio framework.

The talk concludes by evaluating the performance of MPI4Spark and MPI4Dask on the state-of-the-art HPC systems.

iCheck: Leveraging RDMA and Malleability for Application-Level Checkpointing in HPC Systems

Jophin John, Technical University of Munich; Michael Gerndt, Technical University of Munich

The estimate that the mean time between failures will be in minutes in exascale supercomputers should be alarming for application developers. The inherent system's complexity, millions of components, and susceptibility to failures make checkpointing more relevant than ever. Since most high performance scientific applications contain an in-house checkpoint restart mechanism, their performance can be impacted by the contention of parallel file system resources. A shift in checkpointing strategies is needed to thwart this behavior. With iCheck, we present a novel checkpointing framework that supports malleable multilevel application-level checkpointing. We employ an RDMA enabled configurable multi-agent-based checkpoint transfer mechanism where minimal application resources are utilized for checkpointing.

We use the libfabric library to deliver RDMA support into iCheck. RDMA allows a remote process to perform the data access of preregistered memory regions by avoiding CPU interference. This significantly improves the throughput and reduces the roundtrip latency. There are two ways to perform a Checkpoint and Restart operation in iCheck, and it is based on the type of RDMA operations provided by libfabric, namely read and write. iCheck uses a combination of these to perform checkpoint storage and retrieval. Push and pull are the two techniques iCheck supports to transfer the data during checkpoint/restart. In the former, the application writes/reads the checkpoint to/from the iCheck's memory, while in the latter, the iCheck reads/writes the checkpoint from/to the application memory. Additionally, agents can also use multithreading to parallelise the transfers.

The high-level API of iCheck facilitates easy integration and malleability. We have added the iCheck library into the ls1 mardyn application providing performance improvement up to five thousand times over the in-house checkpointing mechanism. LULESH, Jacobi 2D heat simulation, and a synthetic application were also used for extensive analysis.

Introducing Compute Express Link™ (CXL™) 3.0: Expanding Fabric Capabilities and Management

Mahesh Wagh, AMD, CXL Consortium

Modern cloud datacenters require fabric-based heterogenous composable architecture to support compute-intensive workloads for applications such as Artificial Intelligence and Machine Learning. To meet these ever-increasing performance and scale requirements, the CXL Consortium has continued to evolve its standard through the development of the Compute Express Link™ (CXL™) 3.0 specification.

CXL 3.0 supports double the data rate to 64GTs with no added latency over CXL 2.0 and introduces enhanced fabric capabilities and management, improved memory sharing and pooling, coherency, and peer-to-peer communication. The CXL fabric management framework brings fully composable and disaggregated memory to next-generation datacenters and server architectures while enabling switching, memory pooling, and fabrics management capabilities. CXL 3.0 enhances these fabric management capabilities, allowing improvements in scalability and resource utilization while maintaining full backward compatibility with all previous generations.

Presentation attendees will gain insight into the new features in the CXL 3.0 specification and explore the expanded fabric capabilities and management including multi-headed and fabric attached devices, enhanced fabric management, and composable disaggregated infrastructure.

Open MPI and Libfabric on the Frontier Supercomputer

Amir Shehata, Oak Ridge National Labs

Each node of the Frontier supercomputer contains [4x] AMD MI250X, each with 2 Graphics Compute Dies (GCDs) for a total of 8 GCDs per node. The programmer can think of the 8 GCDs as 8 separate GPUs, each having 64 GB of high-bandwidth memory (HBM2E).

Each Frontier node is connected to four of HPE's SLINGSHOT 200 Gbps (25 GB/s) NIC providing a node-injection bandwidth of 800 Gbps (100 GB/s).

In collaboration with the Exascale Computing Project (ECP), Oak Ridge Leadership Computing Facility staff are porting and optimizing Open MPI and libfabric for use on the Frontier supercomputer. The HPE MPI software stack provides the libfabric CXI provider, CXI libraries and drivers which enable the use of the HPE Slingshot NICs.

This project focuses on the following work items:

- Integrate the CXI provider and relevant components in the latest libfabric to enable Slingshot NIC usage with Open MPI.
- Develop a new libfabric provider, LINKx, which allows linking multiple providers, including utility providers. LINKx currently links the SHM provider and CXI providers together. It exposes a LINKx abstraction to higher level

applications, ex: Open MPI, which the application can then bind to. This allows the application to use shared memory and CXI providers for intra and inter node communication respectively. This necessitates the integration into the new libfabric PEER API and the implementation of
  - shared receive queues in LINKx, CXI and SHM providers
  - shared completion queues in LINKx, CXI and SHM providers.
- Add IPC Caching for use in the SHM provider.
- Add IPC communication functionality for ROCm to be used by the SHM provider
- Add asynchronous memory operation workflow for ROCm to be used by the SHM provider
- Add XPMEM support for use by the SHM provider
- Optimize SHM provider locking infrastructure

The intent of this project is to match the functionality and performance of the HPE MPICH software stack. This will provide application developers an alternative to the HPE stack if necessary.

OpenFabrics Management Framework for Composable Distributed Systems

Michael Aguilar, Sandia National Labs; Richelle Ahlvers, Intel Corp.; Phil Cayton, Intel Corp.; Russ Herrell, HPE; Christian Pinto, IBM

HPC and Enterprise computing systems integrate memory, storage components, and accelerator resources to provide effective, versatile, and efficient platforms for user requested computation. Current large-scale parallel computing architectures have limitations to resource provisioning and computational efficiency. The addition of Composable Resource Management can provide dynamic resource availability, as needed.

Management of Composable resources can make possible dynamic insertion and removal of resources, dynamically compiled network architectures to provide network localized traffic, and even dynamic resource distribution.

Implementation of centralized Composable Resource Management provides a common set of network and resource integration tools to allow clients to monitor, aggregate, and subdivide resources and network fabrics. The OpenFabrics Alliance (OFA), together with its partners, the DMTF, SNIA, and the CXL Consortium are launching an effort to design and develop an open fabric management framework that consists integration of Redfish and Swordfish tools designed for managing and manipulating resources using client-friendly abstract avatars.

The OpenFabrics Management Framework provides dynamic resources to mitigate out of memory conditions, mitigate IO page swap 'thrashing', improve network fail-over, improve security and multi-tenancy capabilities, improve user-environment portability, and other features.

OPX Libfabric Provider - Update and Discussion of Performance Improvement Techniques

Tim Thompson, Cornelis Networks; Ben Lynam, Cornelis Networks; Charles Shereda, Cornelis Networks

Omni-Path Express (OPX) is the high performance libfabric provider for Omni-Path networks, and is the preferred library for Cornelis Networks’ future 400G CN5000 product. OPX was designed to provide very low latency for small message sizes, and it succeeded in this initial goal. In the 2022 workshop, we discussed the origins of OPX and its performance relative to its predecessor, PSM2. In this talk, we discuss work completed over the past year implementing components critical to a production libfabric provider, such as SDMA, auto progress, and tag matching, as well as work still in progress, such as expected receive. We also share several techniques we are deploying for identifying opportunities for performance improvement, including debug counters and storing test results in an Elastic data store for charting. Finally, we discuss the challenges encountered from performance tradeoffs that arise when implementing new provider components. In service of these discussions, we present some ‘before and after’ views showing both incremental performance improvements or regressions, as well as final results.

Peer Provider Composability in libfabric

Sean Hefty, Intel Corp.; Alexia Ingerson, Intel Corp.; Jianxin Xiong, Intel Corp.

libfabric defines low-level communication APIs. Implementations of those APIs over a specific network technology is known as a provider. There are providers for RDMA NICs, shared memory, standard Ethernet NICs, as well as customized high-performance NICs. In general, an application running over libfabric typically uses a single provider for all communication. This model works well when there's a single fabric connecting all communicating peers. An application written to libfabric can migrate between different network technologies with minimal effort.

However, such a simple model is unable to achieve maximum performance out of current and future system configurations. For example, Intel's DSA (data-streaming accelerator) offers significant benefits for communication between processes within a single operating system domain. Similarly, GPUs have their own back-end fabrics for communicating between devices, which offers significant bandwidth improvements over standard networks. It is expected that GPU fabrics will attach to devices under the control of different operating systems, and that use case will become more common. Additionally, more traditional HPC networks (e.g. Infiniband) offer in-network accelerations for communications, such as switch-based collectives. It is conceivable that such accelerations could exist in GPU fabrics, or even within the local node using custom devices (e.g. FPGAs or PCIe/CXL plug-in devices).

To achieve the best performance, an application must be able to leverage all of these components well -- local node accelerations, GPU fabrics, GPU fabric switches, HPC NICs, HPC switches, and other attached devices. A significant difficulty in doing so is that these different components may come from different vendors and the application must be able to work across a variety of evolving hardware and network transport configurations.

To support this anticipated complexity, libfabric has introduced a new concept known as peer providers and peer APIs. The peer APIs target independent development and maintenance of highly focused providers, which can then be assembled to present themselves to an application as a single entity. This allows mixing and matching providers from different vendors for separate purposes, as long as they support the peer APIs.

This talk will discuss the peer provider architecture, the current status, and peer API design. It is comprised of 3 related presentations, listed as one submission for ease of review. Together, the 3 presentations will require 60-75 minutes total.

Sample presentations for each section are attached to the submission, but may show up as different presentation versions. However, each presentation is separate.

The presentations are: 1. Introduction to the peer provider architecture and API design. 2. Pairing the shared memory provider with scale-out (i.e. HPC NIC) providers. Two separate, complimentary methods are discussed. 3. Using the peer architecture to support highly focused providers with a scale-out provider. In this example, we'll look at integrating support for a provider focused on offloading collective operations onto switches and how it can be paired with core providers.

RDMA and Linux TCP

Shrijeet Mukherjee, Enfabrica; David Ahern, Enfabrica

The talk will cover our experience at using the RDMA interface to provide zero-copy, asynchronous kernel bypass interfaces to the standard Linux TCP stack and what it takes to run that at 800Gbps.

Status of OpenFabrics Interfaces (OFI) Support in MPICH

Yanfei Guo, Argonne National Laboratory

This session will give the audience an update on the OFI integration in MPICH. MPICH underwent a large redesign effort (CH4) in order to better support high-level network APIs such as OFI. We will show the benefits realized with this design, as well as ongoing work to utilize more aspects of the API and underlying functionality. This talk has a special focus on how MPICH is using Libfabric for GPU support and the development updates on GPU fallback path in Libfabric.

Supporting an Upstream First Kernel Driver for HPC Fabrics

Dennis Dalessandro, Cornelis Networks

The hfi1 driver has been at the heart of Omni-Path systems since its inception. The road started out as a rocky one with the driver making a stint in the staging location in the Kernel tree. Lots of promises and work had to be done to make it into mainline. This included massive code deduplication and creation of a new kernel driver, called rdmavt, which was the subject of a prior OFA talk.

Intel, which is where the driver started, supported customers using a back-ported driver in a software release that included a great number of other components, known as IFS, or Intel Fabric Suite. Upon becoming Cornelis Networks, we renamed this to OPXS, or Omni-Path Express Suite. The concept was the same, a driver that would load on various Linux distributions that included upstream kernel content, including in some cases code which was not upstreamed.

This approach has some issues that make it less than desirable and we will explore these in the accompanying talk. We will also explore our solution to provide customers access to the latest and greatest driver that is tied more closely with the Linux distribution of their choice. Our long standing commitment to upstream first development and dedication to free and open source software are what enable this new method of driver delivery for customers.

Using the FSDP for upstream CI on RDMA Hardware

Doug Ledford, Red Hat, Inc.

The FSDP cluster has the ability to run automated CI tests on upstream repos, both user and kernel. This presentation will demo that ability, look at where we can define the tests for upstream repos, the results and how they are passed back, and how to add new repos and new tests to the framework. In the end, maintainers of upstream repos should have the knowledge they need to get their repo added to the FSDP's CI testing.