Frontier: Exploring Exascale

The System Architecture of the First Exascale Supercomputer

Anonymous Author(s)

ABSTRACT

As the US Department of Energy (DOE) computing facilities began deploying petascale systems in 2008, DOE was already setting its sights on exascale. In that year, DARPA published a report on the feasibility of reaching exascale. The report authors identified several key challenges in the pursuit of exascale including power, memory, concurrency, and resiliency. That report informed the DOE's computing strategy for reaching exascale. With the deployment of Oak Ridge National Laboratory's Frontier supercomputer, we have officially entered the exascale era. In this paper, we discuss Frontier's architecture, how it addresses those challenges, and describe some early application results from Oak Ridge Leadership Computing Facility's Center of Excellence and the Exascale Computing Project.

ACM Reference Format:
Anonymous Author(s). 2023. Frontier: Exploring Exascale: The System Architecture of the First Exascale Supercomputer. In *Proceedings of The International Conference for High Performance Computing, Networking, Storage, and Analysis (SC23).* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In this paper, we provide a system architecture overview of the Frontier exascale supercomputer deployed as part of the Oak Ridge Leadership Computing Facility (OLCF) within Oak Ridge National Laboratory's National Center for Computational Science. While doing so, we reflect on the prescient exascale report published by DARPA in 2008. [37] In that report, the authors outlined the critical challenges to reaching exascale when the TOP500 was expecting its first petascale systems. They foresaw many of the challenges that we have experienced in fielding this machine. While even their most optimistic straw-man projections did not show a path to success, we will show that Frontier meets the spirit of their expectations as a usable exascale system.

For the rest of the paper, we review the four prime challenges outlined in the 2008 report; we then describe Frontier's architecture including compute hardware, interconnect, storage, and software; we evaluate micro-benchmarks for the various hardware types as well as real applications; and lastly we address how Frontier meets or fails to meet the expectations of the 2008 report's authors.

2 THE CHALLENGES OF EXASCALE

In 2007 with the imminent arrival of the world's first petascale systems, DARPA commissioned a working group to study the feasibility of deploying an *exascale* system in the 2015 time frame. The working group met nine times between May and November before drafting their report in 2008. The resulting 230+ page report, "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems" [37] outlined the technological trends at the time and identified critical challenges that would need to be addressed in order to build and deploy an exascale system. While the working group's report discussed many issues, they focused on these four fundamental challenges (in descending order of difficulty):

- Energy and Power
- Memory and Storage
- Concurrency and Locality
- Resiliency

First, they defined what they thought an exascale system would be. They argued that it was more than an exaflop of FP64 performance. They made allowances for up to twice the floor space and twice the power of the first petascale systems (i.e., up to 20 MW/EF). The threshold of 20 MW also was important so that a facility would not pay more for power over the life of the system than it paid for the system.[1] In many cases, they argued that system resources such as memory capacity and bandwidth, storage capacity and bandwidth, etc. should be 1,000X more than the two petascale systems deployed in 2008.[2] They noted that they did not consider cost when setting these targets. As we will describe later, technology costs did not decline by 1,000x and limited the growth of many resources.

2.1 The Energy and Power Challenge

The group reviewed the technology trends for silicon process nodes; data movement within processors, between processors, and throughout the system; as well as storing data in memory and persistent storage. They considered various straw man designs such as *heavy* nodes (also called *fat* nodes as used in OLCF's Frontier and Summit systems) versus *light* nodes (also called *thin* nodes as used in IBM BlueGene systems). They also performed an aggressive, bottom-up exercise of starting with Floating Point Units (FPUs) and then working up to full processors, nodes, and then the full system. None of these straw man designs could approach the 20 MW/EF target with these design points projecting between 68–155 MW/EF.

2.2 The Memory and Storage Challenge

The group then outlined the inability of current technology trends to provide enough memory capacity and bandwidth let alone within a reasonable power budget. They similarly detailed the lack of trends to provide enough storage capacity, bandwidth, and I/O operations per second.

[1] At that time, one definition of a supercomputer was whatever one could buy for 100 million US$. Given an expected service life of five years, the cost is 20 million US$ per year. DOE typically uses a rule-of-thumb that 1 MW of power costs 1 million US$, hence the 20 MW target.

[2] LANL's Roadrunner reported in June and OLCF's Jaguar reported in November.

2.3 The Concurrency and Locality Challenge

The group acknowledged the end of Dennard Scaling (i.e., the stagnation of clock frequencies since the early 2000s) leaving increased parallelism via core-count growth as the only way to increase performance. With clock frequencies stuck around 1 GHz, they projected that core counts would grow to 1 billion for the full system. The group described different applications' levels of spatial and temporal locality. Most of the studied applications had low or no spatial locality and temporal locality varied from low to high. Lower locality increased the overhead due to communication.

2.4 The Resiliency Challenge

Lastly, the group highlighted the explosive growth in component counts and complexity, the need to use advanced technology, the desire to use lower voltages, and potentially new types of failures due to smaller and smaller process geometries.

3 FRONTIER'S ARCHITECTURE

In this section, we describe the node design, the interconnect, the I/O subsystem, and the software environment.

3.1 Node Design

HPE's official designation for a Frontier node is Cray EX 235a, but we still use the development name, *Bard Peak*. The Bard Peak design continues the heterogeneous CPU+GPU (*heavy/fat*) node designs found in OLCF's Titan and Summit systems. While Titan had a ratio of 1:1 CPU to GPU and Summit has a ratio of 1:3, the Frontier's Bard Peak node has a ratio of 1:4, sort of. We describe both processors below and explain the "sort of" comment as well.

3.1.1 AMD's EPYC™ 7A53 "Trento" CPU. AMD's EPYC line of processors has highlighted AMD's chiplet and packaging technologies. Designed specifically for Frontier, the EPYC 7A53 CPU, code-named Trento, uses the same Zen3 cores as Milan. It has 64 cores distributed across eight Core Complex Dies (CCD) [2] and is similar to the Milan 7713/7713P. The difference is in the central I/O Die (IOD). AMD created a custom IOD that replaced the PCIe connections with InfinityFabric™ connections described below.

Each Trento has eight 64 GiB DIMMs of DDR4-3200 memory with a peak bandwidth of 205 GiB/s. EPYC CPUs support NUMA-Per-Socket (NPS) of one, two, or four NUMAs. [2] When run in NPS-1, all memory allocations are striped over all eight DIMMs with the benefits including more bandwidth available to a single process but at the cost of lower aggregate bandwidth and slightly higher latency. In NPS-4, allocations are striped over the two DIMMs in the same quadrant. The local access provides slightly lower latency and slightly higher aggregate bandwidth when processes in each NUMA access memory concurrently. Frontier uses NPS-4.

3.1.2 AMD's Instinct™ MI250X GPU. AMD's Instinct MI200 GPU is AMD's fourth generation in the Instinct line of CDNA™ GPGPUs. AMD has three versions including MI210, MI250, and MI250X. Bard Peak uses the MI250X. The MI250X uses the OCP Accelerator Module (OAM) package that requires water cooling. The MI250X is composed of two chiplets, called Graphics Compute Dies (GCD). The GCDs are why we used the "sort of" comment above. Each GCD presents itself to the operating system as a GPU. Because of this, the user sees eight GPUs when they query the node. The result is that each of the Trento CCDs is paired with a GCD.

The primary distinction between the MI250 and the MI250X that is used in Frontier is the connection to the host. While the MI250 uses traditional PCIe connections, Frontier's MI250X uses InfinityFabric instead to connect to the Trento CPU.

Compared to its MI100 predecessor, each MI250X GCD doubled the 64-bit floating point (FP64) performance. Support for fast hardware-based FP64 atomic operations was also added. Each GCD has four HBM stacks like the MI100, but the data rate is increased to an aggregate peak of 1.635 TB/s. When compared to the peak bandwidth of the CPU, the node's aggregate peak GPU HBM bandwidth of 13.08 TB/s is 64 times greater. This ratio is higher (worse) than the ratio of 50X on Titan and 16X on Summit. With a ratio this high, we expect most users will keep their data in the HBM and avoid moving it back and forth to the CPU as much as possible.

3.1.3 AMD's InfinityFabric™. AMD's InfinityFabric[3] is a coherent, bi-directional, memory-semantic interconnect used to connect processors. Designed for CPU-to-CPU connectivity, AMD extended it for GPU-to-GPU connectivity starting with the MI50 GPUs. For the Bard Peak node design, AMD enhanced xGMI to connect the Trento CPU to the eight MI250X GCDs using the xGMI 2.0 protocol with a theoretical peak of 36+36 GB/s per CPU-to-GCD connection.

The node also uses groups of 1, 2, and 4 xGMI 3.0 links running at 50+50 GB/s to anisotropically connect the eight GCDs in a *twisted ladder* topology as shown in Figure 1. First, the two GCDs within each MI250X OAM package have four north/south links between them for a theoretical peak of 200+200 GB/s. The north/south connections between OAM packages have two xGMI links for a theoretical 100+100 GB/s. All east/west links are single links running at 50+50 GB/s. In the image, the diagonal links are shown in different colors for easier visualization.

3.1.4 HPE's Slingshot NIC. One of the chief innovations of the Bard Peak design is how the node connects to the Slingshot fabric. Traditionally, the Network Interface Cards (NIC) have been attached directly to the CPU or in some newer designs to a PCIe switch that is connected to the CPU and the GPUs. Because the data will mostly reside in the GCD's HBM as mentioned above, each of the four NICs is attached to one of the MI250X OAM packages.

Each Slingshot NIC, code-named Cassini, is a 200 Gb/s Ethernet NIC that can operate using a HPE-proprietary *HPC Ethernet* mode that reduces latency and provides OS-bypass. [46]

3.2 Slingshot Interconnect

Connecting Frontier's nodes is the Slingshot interconnect. Slingshot is a superset of Ethernet that includes proprietary HPE extensions to reduce average latency, reduce tail latency, improve bandwidth, and improve message rates. [32] Frontier has a three-hop dragonfly [35] topology comprised of 80 groups: 1 management, 5 I/O, and 74 compute. The management and I/O groups consist of 16 fully-connected, top-of-rack switches each, while the compute groups each include 32 fully-connected, water-cooled blade switches.

[3]InfinityFabric refers both to chiplet-to-chipet connections within a socket and to socket-to-socket connections. AMD calls the chiplet connections *Global Memory Interconnect* or GMI. AMD calls the socket-to-socket connections *xGMI* (external GMI). [2]

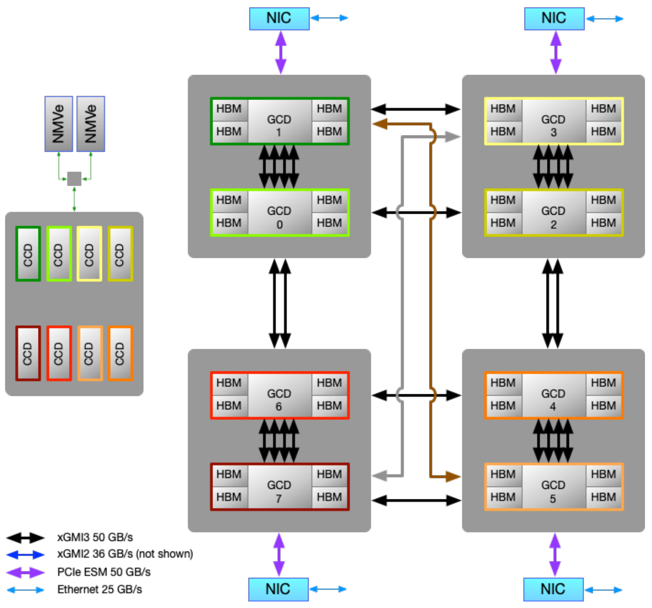

Figure 1: Connectivity within the Bard Peak node. The CPU<->GPU links are omitted for clarity.

Connections between groups are typically counted as *bundles*, with each bundle consisting of a QSFP-DD active optical cable with two 200Gb/s links. Connections between Frontier compute groups use a bundle size of two, which provisions 7.3 TB/s of global connectivity per group compared to 12.8 TB/s of injection bandwidth. This is sometimes referred to as the network *taper*. Frontier's ratio of global-to-injection bandwidth is 57%. This reduces the effective global bandwidth available to the nodes. The total global bandwidth between the compute groups is 270+270 TB/s. There is one bundle from each compute group to each storage group and the management group. There are five bundles between storage groups. The storage groups also have three bundles to the management group.

The dragonfly topology is a *direct* network, which means all switches have some endpoints to them in contrast to an *indirect* network such as a Clos or fat-tree that has some switches that only have connections to other switches. Direct networks, such as dragonfly, slimfly, and hyper-x, use non-minimal routing to take advantage of additional paths through the fabric to achieve higher bandwidth compared to only using minimal paths. As we will show later, the impact is that the global bandwidth is reduced yet again.

3.3 Storage Subsystem

Frontier's I/O subsystem is an evolution of the CORAL-1 designs [70] and consists of two main subcomponents: node-local storage and a center-wide parallel file system (PFS).

Like Summit, each Frontier node features node-local storage. Each Frontier node has two NVMe M.2 drives configured in RAID-0 (striping, no redundancy) to increase bandwidth and I/O Operations Per Second (IOPS). The NVMe mount provides ~3.5 TB of capacity as well as 8 GB/s for reads, 4 GB/s for writes, and up to 2.2 million IOPS, per Frontier node. The node-local storage is user-managed and is primarily for caching writes from modeling/simulation jobs and caching reads for machine learning jobs. OLCF is developing software for both of these use cases.

I/O Subsystem Specifications			
Tier	**Capacity** PB	**Read BW** TB/s	**Write BW** TB/s
Node-Local	32.9 PB	75.3 TB/s	37.6 TB/s
Orion Metadata	10.0 PB	0.8 TB/s	0.4 TB/s
Orion Performance	11.5 PB	10.0 TB/s	10.0 TB/s
Orion Capacity	679.0 PB	5.5 TB/s	4.6 TB/s

Table 1: I/O Subsystem capacity and theoretical read/write bandwidths.

Unlike Summit's GPFS file system, Frontier uses Lustre for its center-wide PFS, named Orion. Orion comprises 225 Scalable Storage Units (SSU), each of which has two controllers, two Cassini NICs per controller, (24) 3.2 TB NVMe drives, and (212) 18 TB hard drives. Within an SSU, the NVMe drives and hard disks are configured into two distinct sets of ZFS DRAID-2 groups for redundancy and performance. These two distinct sets are then aggregated at the PFS layer across all SSUs as an NVMe-based flash performance tier and a hard disk-based performance tier, under the same single POSIX namespace. Also, as part of the Orion namespace, the Lustre metadata servers host NVMe flash devices to enable improved metadata and small I/O performance.

While both HPE and OLCF are developing software that would allow writes to go to the performance tier and then auto-migrate to the capacity tier, neither is deemed production ready at the time of this writing. Instead, OLCF has configured Orion with Lustre's Progressive File Layout (i.e., a self-extending layout) that lands the first 256 KB of data of each file in the flash-based metadata servers using Lustre's Data-on-Metadata (DoM) feature. The range greater than 256 KB up to 8 MB lands in the performance tier. All remaining data lands in the capacity tier. The intent for using this PFL layout is to cache really small files in the metadata servers such that the contents are returned when the file is opened without having to then contact an object server. The data stored in the performance tier will hopefully include the file's index data if it has any. The I/O subsystem's specifications are listed in Table 1.

3.4 Software Ecosystem

3.4.1 Operating System. The operating system environment for Frontier is HPE Cray OS, a specialized version of SUSE Enterprise Linux. While this would appear to continue the established dominance of Linux-based systems in the Top500, it must be noted that node-level OS architectures are becoming much less significant on systems where the vast majority of performance capabilities are delivered via GPUs and other accelerators. This stands in contrast to early predictions made during the lead-up to exascale which assumed that billions of threads running on massive numbers of general-purpose computing elements would require specialized and lightweight operating systems to meet the scaling challenges posed by projected exascale node architectures [6].

On Frontier, in terms of total node-level compute capacity, Linux directly manages only a small portion of the system (CPUs) while the majority of compute is staged to GPU cores, which are managed by their own system software residing in device firmware.

Therefore, while Linux does still manage performance-critical sections running on the CPUs, the majority of the computation is managed by a specialized, lightweight OS executing directly on the GPU itself. In this way, Frontier's system software resembles more of a multi-kernel architecture [24, 56, 72] than one with a single monolithic OS, validating the substance of the initial exascale OS/R projections if not their exact implementation.

3.4.2 *System Management.*

Frontier is powered by HPE's Performance Cluster Manager (HPCM), a traditional system management tool that is flexible and straightforward to use. OLCF developed tools to pre-define hardware in the HPCM database prior to delivery, simplifying the process to install the system in phases. HPCM includes a daemon that periodically queries the chassis management modules for hardware changes, so hardware additions or maintenance activities are noticed and reflected in the HPCM database without human intervention.

One admin node and (21) leader nodes provide shared utility storage based on Gluster (for logging, node images, etc.), console and syslog management, and reliable, scalable boot. Leader-node failure is transparently handled by HPCM's CTDB implementation – another leader node takes over the virtual IP of the failed node and assumes responsibility for all of its clients. Access to the center-wide NFS home and software areas is provided by (12) dedicated nodes that run Data Virtualization Services (DVS) to cache and forward I/O requests. Two dedicated nodes run the Slurm controller and database daemons, and a single dedicated node runs the Slingshot Fabric Manager software.

Slurm serves as the system-level scheduler. Compute nodes are scheduled exclusively to a single job at a time, which simplifies security requirements and node cleanup procedures. At boot and between every job, Slurm runs a *checknode* script that verifies the health of every compute node. Slurm can also manage application launch *steps*, or separate workflow software can manage individual processes. Slurm integrates with the Slingshot software to allocate a unique Virtual Network Identifier (VNI) per jobstep to support isolation between applications. Additionally, Slurm is topologically aware, so it will optimize job placement for ideal network performance. For small jobs able to fit within a single rack/group, Slurm will pack allocations tightly to minimize global hops. For larger jobs, Slurm will attempt to spread a job evenly across as many Slingshot groups as possible to maximize the number of global connections (and thus global bandwidth) available to minimal routing.

HPE Slingshot switches boot without any configuration applied, and it is up to the Slingshot Fabric Manager to send port configuration and routing instructions to each Slingshot switch. The fabric manager periodically sweeps all the switches in the fabric to search for failures or changes to the topology and sends updated routing tables to all affected network switches.

3.4.3 *Programming Environment.*

The programming environment on Frontier is anchored by two vendor-provided software stacks, HPE's Cray Programming Environment (CPE) [31] and AMD's Radeon Open Ecosystem (ROCm) [4], together with additional software installed and managed by OLCF staff originating from a variety of sources, including the Exascale Computing Project (ECP). [14] Both stacks are full-featured, including compilers, performance, and correctness tools, and a variety of numerical, machine learning, and other libraries. They support C, C++, and Fortran programming languages, as well as OpenMP [54] and AMD's Heterogeneous Interface for Portability (HIP) [5]. HIP is an open-source, work-a-like to NVIDIA's Compute Unified Device Architecture (CUDA) environment [48]. The compilers generally support most features of OpenMP 5.0 [7], 5.1 [8], and 5.2 [9] at present, with the implementation of the remainder in progress. The C and C++ compilers in both stacks are based on the open-source LLVM compiler suite [43]. Cray's Fortran compiler is not LLVM-based but does provide comparable support for OpenMP to their C/C++ compilers. ROCm's Fortran compiler is based on what is now referred to as "classic" Flang [42] and lags in the implementation of OpenMP features.

Both programming environments provide a suite of libraries that have been tuned for the hardware architectures (both CPU and GPU in the case of CPE; primarily GPU for ROCm). These libraries support low-level numerical operations, such as BLAS, LAPACK, FFT, and sparse linear algebra; as well as low-level primitives for communication, concurrency, and mixed-precision matrix multiplication. The ROCm stack includes two versions of many libraries. The "hip"-branded libraries are thin compatibility layers offering interfaces similar to the corresponding NVIDIA "cu" libraries that call vendor-optimized backend device libraries (e.g., AMD "roc*" and NVIDIA "cu*").

For debugging, ROCm includes ROCgdb, a source-level debugger based on gdb, while CPE adds gdb4hpc (a parallel-capable version of gdb), stack trace analysis tool (STAT), and abnormal termination processing (ATP). For performance, the ROCm stack includes the rocProfiler and rocTracer libraries, which underpin the ports of most performance tools to the Frontier platform. The primary user-facing tool provided by the ROCm stack is rocprof. In CPE, the primary user-facing tool is the Performance Analysis Tool (PAT), while Reveal is a parallelization assistant.

As mentioned above, the OLCF supplements the two vendor stacks with additional tools. For performance, the open-source HPCToolkit [1, 59], Tuning and Analysis Utilities (TAU) [62, 68], and Score-P [36, 71] tools are available, as well as the VAMPIR trace visualizer [27]. Linaro Forge [41] (until recently, Arm Forge), which includes the MAP performance tool and DDT debugger is also available on the system. In the compiler area, OLCF deploys gcc [19], which they have been working with Siemens Digital Industries to support OpenMP offload (5.0 is nearly feature complete with 5.1 is in progress). Through a collaboration with the Argonne Leadership Computing Facility (ALCF) and Codeplay, a pilot port of the open-source Intel DPC++ SYCL compiler [33] is also available on the system. SYCL is one of the primary programming models for ALCF's Aurora supercomputer.

The programming environment strategy for Frontier is, in many respects, the straightforward evolution of that provided on Titan, OLCF's first production GPU-accelerated system. For Titan (and also Summit), CUDA was the low-level programming tool for GPU offload. To encourage the development of a higher-level, more portable programming approach, OLCF worked with its vendor partners and others in the community to launch OpenACC [53] as a directive-based solution for GPU offload. In this time frame, the well-established OpenMP community, also a directive-based approach, was also beginning to address the GPU offload programming challenge, though it was not available as a production tool during Titan's lifetime. Titan's successor, Summit, also used NVIDIA GPUs and offers CUDA, OpenACC, and OpenMP for GPU offload programming. With Frontier's AMD GPUs, the low-level programming tool has transitioned to HIP, which is very similar to CUDA in approach and capability. OpenMP has become the leading standards-based offload approach for Frontier, overtaking OpenACC in terms of application uptake. This may in part be a response to the fact that there is no commitment to support OpenACC from either of the Frontier vendors. Cray Fortran supports OpenACC 2.0 [49], which dates from 2013; the current version is 3.2 [52]. The gcc compiler suite is the main vehicle for teams requiring OpenACC on Frontier; it currently supports version 2.6 [50] of the standard (released in 2017), with 2.7 [51] support planned. Additionally, while OpenMP started supporting GPU offload later than OpenACC, it was still in time for many, already familiar with OpenMP for CPU threading, to adopt it for GPU offload as well.

4 INITIAL EVALUATION

4.1 Node-Level Performance

We present traditional micro-benchmarks for the CPUs, GPUs, and the links between them.

4.1.1 AMD EPYC™ 7A53 "Trento" CPU. With over 99% of the FLOPs in Frontier coming from the GPUs, the primary metric for Frontier's CPUs are their ability to move data to and from memory. Trento is able to achieve up to 180 GB/s using non-temporal loads and stores in NPS-4 mode. When operating in NPS-1, that rate drops to ~125 GB/s. Table 2 shows our STREAM benchmark results with temporal and non-temporal stores for an array size of ~7.6 GB, which illustrates how caching can negatively affect bandwidth when data are not expected to fit into cache.

Function	Temporal (MB/s)	Non-Temporal (MB/s)
Copy	176780.4	179130.5
Scale	107262.2	172396.2
Add	125567.1	178356.8
Triad	120702.1	178277.0

Table 2: CPU STREAM bandwidth results using temporal and non-temporal stores.

4.1.2 AMD Instinct™ MI250X GPU. As mentioned, AMD's Instinct MI250X GPU consists of two GCDs, where each GCD has a peak FP64 performance of 23.95 TFLOP/s and 64 GB of HBM which the GCD can access at 1.635 GB/s (double numbers for the full MI250X).

In Figure 2, we show our FP64, FP32, and FP16 performance achieved using the CoralGemm benchmark. [3] CoralGemm uses the hipBLAS library, which can use a combination of vector- and matrix-core instructions based on internal heuristics (this cannot currently be toggled on/off). In the figure, we see the FP32 and FP64 results exceed the peak performance of the GCD, reaching values of 24.1 and 33.8 TFLOP/s, respectively. This is likely due to the use of matrix-core instructions, which we verified were being used at all precisions using AMD's *rocprof*. The FP16 results reached 111.2 TF/s for the matrix sizes we tested.

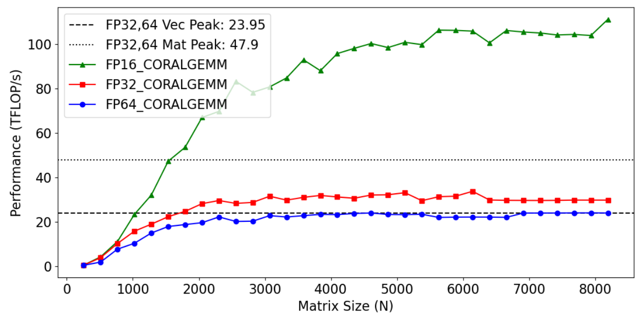

Figure 2: Comparison of peak FP64, FP32, and FP16 performance with achieved values of a single MI250X GCD.

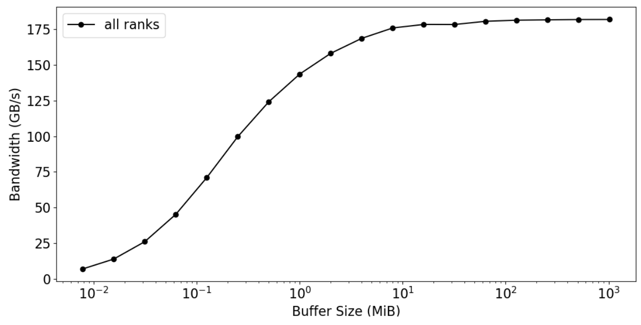

Figure 3: Aggregate CPU-to-GPU bandwidth for 8 MPI ranks concurrently targeting their own GCD.

Table 3 shows our results from the STREAM GPU benchmark for an 8 GB array size, showing that we achieve between 79% and 84% of peak HBM bandwidth depending on the test being performed.

STREAM Function	Bandwidth (MB/s)
Copy	1336574.8
Mul	1338272.2
Add	1288240.3
Triad	1285239.7
Dot	1374240.6

Table 3: GPU STREAM bandwidth results.

4.2 Interconnect Performance

4.2.1 Intra-Node. The intra-node bandwidth between processors on the Bard Peak nodes is enabled by AMD's InfinityFabric. When we measure bandwidth from a single CPU core to the GCD and vice versa, we see it reach 25.5 GB/s, ~71% of the peak xGMI 2.0 bandwidth. The more likely scenario is that eight processes will contend for memory access and Figure 3 shows the total CPU-to-GCD bandwidth from eight MPI ranks reaches about 180 GB/s, matching the Trento's STREAM performance.

Figure 4 shows the achieved GCD-to-GCD bandwidths for 1-, 2- and 4-xGMI link GCD pairs. These results show that System Data Memory Access (SDMA) transfers are capped at ~50% GB/s regardless of the number of xGMI links between the GCDs, while

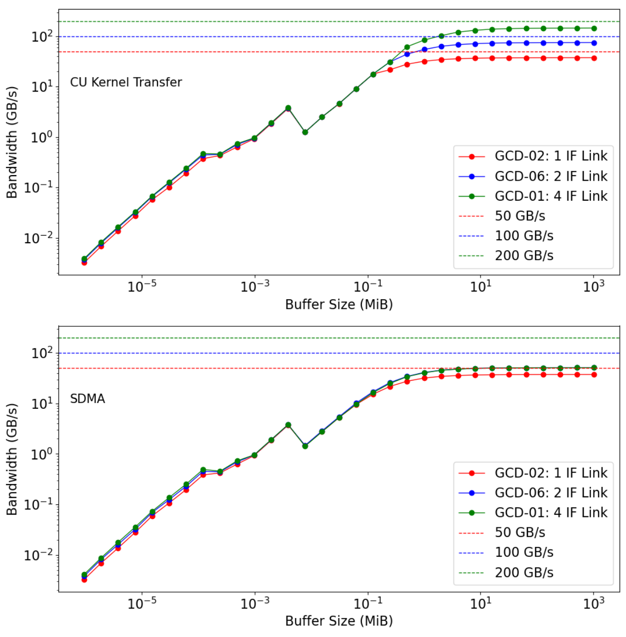

Figure 4: Achieved GCD-to-GCD bandwidth between GCD pairs on a Bard Peak node. Top: Results when using CU kernel transfers. Bottom: Results when using SDMA transfers.

the GPU Compute Unit (CU) kernel transfers can reach 37.5 GB/s, 74.9 GB/s, and 145.5 GB/s for GCD pairs with 1-, 2-, and 4-xGMI links, respectively.

4.2.2 Inter-Node. When designing a system for a fixed budget, the integrator has to optimize the system to provide the highest application speedups by balancing competing resource needs. One of the trade-offs that HPE made was changing the topology from a Clos (i.e., non-blocking fat-tree) in Summit to a dragonfly in Frontier. A dragonfly has ~50% less ports and cables compared to a Clos and is similar to a 2:1 over-subscribed fat-tree.

In Figure 5, we compare the network bandwidth of Slingshot 11 in Frontier with InfiniBand EDR in Summit. Slingshot provides 200 Gbps (25 GB/s) per endpoint; EDR provides 100 Gbps (12.5 GB/s) per endpoint. In these measurements, we use mpiGraph [45] to show the receive-side bandwidth for each transferring pair plotted in a histogram. In the histogram for Summit, a non-blocking fat tree, we see a tight distribution of measurements of ~8.5 GB/s per NIC out of the theoretical max of 12.5 GB/s. Nearly all of Summit's traffic achieves this level of performance. On Frontier, we measure a much wider distribution of bandwidths ranging from 3 GB/s to 17.5 GB/s out of a theoretical max of 25 GB/s. This distribution can be explained by the full connectivity within a dragonfly group, the 57% global-to-local ratio between groups, and the impact of non-minimal global routing. Each Frontier compute dragonfly group includes 128 nodes with 512 endpoints (i.e., 1/74th or ~1.4% of the total) and is the very small grouping in the figure around 17.5 GB/s. This very small distribution achieves a similar percentage of peak as Summit's tight distribution. The lowest performance of ~3

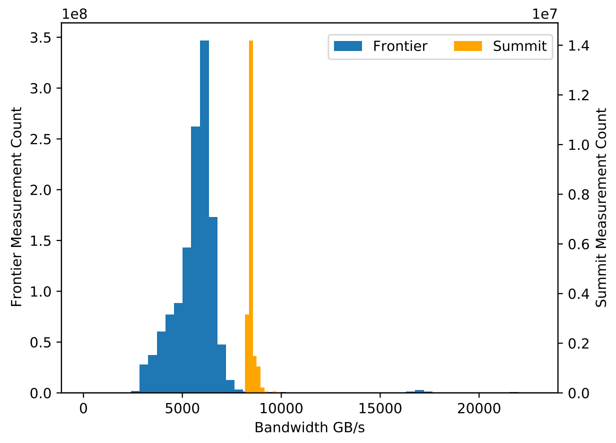

Figure 5: mpiGraph Per NIC Measurements

GB/s occurs when all traffic is using the global links (i.e., no intra-group traffic) that divides the available 270.1 TB/s global bandwidth between all 37,632 endpoints and then non-minimal routing divides that in half due to non-minimal traffic competing for the same links. The rest of the histogram accounts for various ratios of intra- versus inter-group traffic. This figure highlights the impact of topology on usable bandwidth when running full-system jobs. For all-to-all communication, for example, we see ~30-32 GB/s/node (~7.5-8.0 GB/s/NIC) when running 8 PPN with 128 KiB messages.

A major technology feature of the Slingshot network is state-of-the-art hardware congestion control. This plays an important role in production supercomputers that are often running dozens or more jobs at a given time. Congestion control reduces the impact of adversarial communication patterns on neighboring jobs. To understand the impact of congestion control in Frontier we use GPCNeT [12], a network benchmark, that induces adversarial traffic while victim nodes take performance measurements. We ran GPCNeT on 9,400 nodes with 7,520 *congestor* nodes performing various communication patterns (i.e., all-to-all, one- and two-sided incast, one- and two-sided broadcasts) and 1,880 victim nodes. The measurements shown in Table 4 were run using 8 PPN, the expected use-case for most applications. The isolated and congested tests show identical performance (i.e., a impact factor of 1.0x), which is ideal. When we ran with 32 PPN, the results, on the other hand, show some degradation in both the average and 99% of 1.2-1.6x and 1.8-7.6x, respectively, although both are much better than Summit's EDR InfiniBand results [73]. The *Isolated Network Tests* results in Table 4 also provide the average and tail latency for point-to-point and allreduce.

4.3 Storage Evaluation

4.3.1 Node-Local Storage. With exclusive access per node, the node-local storage is consistent in performance and scales with the number of nodes in the job. Compared to the contracted peak performance per node of 8 GB/s for reads, 4 GB/s for writes, and 1.6 million IOPS, we measured 7.1 GB/s for sequential reads, 4.2 GB/s for sequential writes, and 1.58 million 4k IOPS for random-read I/O workloads using the industry standard `fio` benchmark. For a job using all of Frontier's nodes, it should expect an aggregate,

Isolated Network Tests - 8 PPN			
Name	**Average**	**99%**	**Units**
RR Two-sided Lat (8 B)	2.6	4.8	usec
RR Two-sided BW+Sync (131072 B)	3497.2	2514.4	MiB/s/rank
Multiple Allreduce (8 B)	51.5	54.1	usec
Network Tests running with Congestion - 8 PPN			
Name	**Average**	**99%**	**Units**
RR Two-sided Lat (8 B)	2.6	4.7	usec
RR Two-sided BW+Sync (131072 B)	3472.2	2487.0	MiB/s/rank
Multiple Allreduce (8 B)	51.6	54.3	usec

Table 4: GPCNeT running on 9,400 nodes with 8 processes per node (8 PPN). With 8 PPN, the result is ideal (congested is no worse than isolated).

node-local performance of 66.8 TB/s for reads, 39.5 TB/s for writes, and ~14.9 billion IOPS.

4.3.2 Lustre Parallel File System. Orion easily meets the needs for Frontier's applications. Based on historical Titan and Summit usage data, 90% of applications write 15% or less of the GPU memory per hour. Compared to the contracted streaming performance of 10.0 TB/s for reads and writes, we measured up to 11.7 TB/s for reads and up to 9.4 TB/s for writes if the application has small files that fit within the Flash tier. Large files will see 4.9 TB/s and 4.3 TB/s for reads and writes, respectively. With 4,704 TiB of HBM memory, Orion should be able to ingest ~700 TiB (~776 TB) in ~180 seconds. At this rate, most apps will spend less than 5% of walltime per hour doing I/O.

4.4 Initial CAAR, INCITE, and ECP Application Results

As discussed below, DOE chose to focus on the speedup of real applications relative to previous peta-scale machines (Titan, Sequoia, Cori, Mira, Theta, or Summit) in order to test the performance of exascale systems. Here we provide initial computational results originating from OLCF CAAR (Center for Accelerated Application Readiness), INCITE (Innovative and Novel Computational Impact on Theory and Experiment), and ECP (Exascale Computing Project) codes that cover a wide range of science domains. These domains span topics that include genomics, materials science, plasma physics, astrophysics, computational fluid dynamics, particle accelerators, cosmology, molecular dynamics, nuclear fission reactor design, and fusion plasmas. The KPP (Key Performance Parameter) goals for the various codes were set at a 4x performance increase for the CAAR codes relative to Summit and a 50x increase for the ECP codes relative to the ~20 PF machines listed above.

4.4.1 Initial CAAR and INCITE Application Successes.

CoMet. The CoMet (Combinatorial Metrics) application [34] computes similarity metrics between vectors stored in very large datasets. It has been used to solve clustering problems in areas such as genomics, climate, bioenergy, and pandemics [40]. Under the OLCF CAAR project, CoMet was optimized to achieve high performance on the AMD GPU architecture by making effective use of mixed-precision matrix multiplies. The primary algorithm targeted for CAAR was the 3-way Custom Correlation Coefficient (CCC) method. On Frontier, the measured number of element comparisons per second (a measure of science output) for CoMet was 419.9 quadrillion comparisons/second on 9,074 compute nodes, a factor of 5.16X faster than the Summit baseline of 81.2 quadrillion comparisons per second. The compute rate for this run reached 6.71 Exaflops mixed-precision on Frontier.

CAAR and INCITE Application Results			
Application	**Baseline**	**Target**	**Achieved**
CoMet	Summit	4.0x	5.2x
LSMS	Summit	4.0x	7.5x
PIConGPU	Summit	4.0x	4.7x
Cholla	Summit	4.0x	20.0x
GESTS	Summit	4.0x	5.9x
AthenaPK	Summit	4.0x	4.6x

Table 5: CAAR and INCITE applications that have exceeded their KPP of 4.0x over Summit.

LSMS. LSMS (Locally Self-consistent Multiple Scattering) is a code for calculating the electronic structure in materials and condensed matter systems from first principles. The quantum mechanical behavior of the electrons in LSMS is treated using Kohn-Sham density functional theory and the resulting effective one-particle equations are solved in real space using multiple scattering theory to calculate the Green's function of the electrons in the system. These calculations are dominated by dense linear algebra on double complex numbers. LSMS can achieve linear scaling of the computational effort with the number of atoms in the system compared to the usual cubic scaling of conventional density functional theory electronic structure codes. This optimal scalability allows for the modeling of much larger systems than were accessible previously. A figure of merit that captures both the potential for weak and strong scaling of the code as well as the underlying computational complexity of the LSMS algorithm was developed. The Summit GPU code utilizes CUDA and cuSolver for its computational kernels. These kernels were ported to AMD GPUs by translating the kernels to their HIP and rocSolver equivalents. For the matrix inversion kernel for the $l_{max} = 7$ test case this resulted in a per GPU speedup averaging approximately 7.5x compared to Summit's V100 GPUs when including additional kernels ported and optimized during the CAAR project. For a 1,048,576 atom simulation LSMS achieves a FOM of 1.027e16 on 8,192 Frontier nodes, while on 4,500 Summit nodes the same simulation reached a FOM of 4.513e14 for the baseline pre-CAAR code and 3.106e15 for the code base including the CAAR developments.

PIConGPU. PIConGPU (Particle-In-Cell on GPUs) uses a particle-in-cell model to simulate time-dependent electron motion in laser-driven plasmas. At each time-step, macroparticles representing electrons interact with a local electromagnetic field, and then the field within each cell is updated following Maxwell's equations [10]. PIConGPU's figure of merit therefore measures the number of particle and cell updates completed per second, weighted by 90% and 10%, respectively. A full-scale Summit run of PIConGPU completed in late 2019 showed 14.7e12 updates per second, with about 92% of PIConGPU's run-time spent executing GPU kernels. Porting to AMD hardware was enabled by Alpaka – a performance portability framework which quickly adopts emerging new hardware accelerators. When running on Frontier at nearly full scale (9,216 Frontier nodes in July, 2022), PIConGPU achieved 90% weak scaling efficiency and 65.7e12 updates per second, a factor of 4.5x higher than full-scale Summit. This overall speedup can be traced to a 25% speedup in the single MI250x GCD vs. V100 comparison, multiplied by the greater number of GPUs available on Frontier.

Cholla. Cholla (computational hydrodynamics on parallel architecture) started as a hydrodynamics code which has since been developed to include radiative cooling, self-gravity solver, and particle tracking suitable for astrophysics and cosmology simulations. Cholla was originally written in C++ and CUDA. During the CAAR program Cholla was ported and optimized to HIP. Cholla achieved 20X speedups on Frontier from its baseline run on Summit. About 4-5X of these speedups can be attributed to the intensive algorithmic optimizations while the rest comes from hardware improvements from Summit to Frontier.

GESTS. The GESTS (GPUs for Extreme Scale Turbulence Simulations) project developed a Pseudo-Spectral Direct Numerical Simulation (PSDNS) algorithm to study fundamental behavior of turbulent flows across a wide range of scales according to the Navier-Stokes equations. The GESTS codes are written in Fortran 95 using GPU-Aware MPI for communication and are built around a custom-designed 3D FFT algorithm that computes the FFTs on the AMD GPUs with ROCm rocFFT. OpenMP offloading functionality is used to manage data movement between the host and device, to enable GPU-Direct MPI communications, and to accelerate a variety of array operations on the GPUs. The original GPU-enabled algorithm along with previous results from Summit are presented in [58] and [11].

The Figure of Merit (FOM) chosen for GESTS is defined as $FOM = N^3/t_{wall}$ where N^3 is the total number of grid points in the simulation and t_{wall} is the average time to compute each time step. The reference FOM was computed on Summit with a 1D decomposition as part of an INCITE 2019 project. [58] GESTS exceeds the CAAR project goal of a 4x speedup on Frontier for both the 1D (5.87x) and 2D (5.06x) decompositions of the domain with $N^3 = 32768^3$ grid points. These cases are the largest known DNS computations to date with a point total in excess of 35 trillion grid points. No other computational resource in the world besides Frontier has the memory capacity to complete these simulations.

AthenaPK. AthenaPK [26] (Athena-Parthenon-Kokkos) is a general purpose astrophysical magnetohydronamics code which serves as a performance-portable (through use of Kokkos [67]), AMR-capable (through use of Parthenon [26]) conversion of Athena++ [63]. It implements the hydrodynamics solvers from Athena++ and supplemented them with a divergence cleaning magnetohydrodynamics solver. The code is used for simulations of magnetized galaxy clusters with feedback from active galactic nuclei, cf., cloud crushing in galactic outflows, and magnetohydrodynamic turbulence. AthenaPK is available on GitHub [25] with contributions welcome and encouraged.

Single node and full system experiments were performed using a 3D linear wave problem sized to use a large portion of the available high-bandwidth memory on each node. Comparisons evaluate problem size and raw performance in terms of cell-updates/s. For single node runs, a Frontier node achieved 1.2x more cell-updates/s with an 8x larger problem than on a Summit node. When weak-scaled, 9,200 Frontier nodes achieved 4.6x more cell-updates/s with an overall 16x larger problem than on 4,600 Summit nodes. Weak-scaling results were achieved with 96% and 48% parallel efficiency on Frontier and Summit, respectively. The difference in parallel efficiency is attributed to Frontier's improved node design, specifically each GPU having a network interface card connected to it [26].

ECP Application Results			
Application	**Baseline**	**Target**	**Achieved**
WarpX (vs. Warp)	Cori	50x	500x
ExaSky	Theta	50x	234x
EXAALT	Mira	50x	398.5x
ExaSMR	Titan	50x	70x
WDMApp	Titan	50x	150x

Table 6: ECP applications that have exceeded their KPP of 50x over Titan, Sequoia, Cori, Mira, or Theta.

4.4.2 *Initial ECP Application Successes.*

WarpX. In the US DOE Exascale Computing Project, WarpX is used to develop the next generation of particle accelerators. In particular, WarpX is used to further the reach of laser-driven, plasma-wakefield stages, chained towards the development of compact, future colliders for high-energy physics exploration [69]. The code implements advanced algorithms for the electromagnetic and -static particle-in-cell (PIC) loop, such as embedded boundaries of particle accelerator structures. The team spear-headed the development of novel algorithms such as mesh-refinement in electromagnetic PIC, the Lorentz-boosted frame method, pseudo-spectral field solvers for long-term stability, among others. Beyond particle acceleration, the code is actively applied to research in diverse domains such as modeling in fusion-energy sciences, high-energy-density laboratory plasma physics, astrophysical plasmas, and microelectronic devices.

WarpX was the first application in ECP to achieve the KPP goal in July 2022 by running on nearly the full size of Frontier. The development of laser-plasma devices depends critically on high-performance, high-fidelity modeling to capture the full complexity of acceleration processes that develop over a large range of space and timescales – and high weak-scaling efficiency at scale addresses this need. WarpX achieves near-ideal weak-scaling over multiple orders of magnitude of system utilization and realistic strong-scaling over an order of magnitude in node-numbers for its 3D, block-structured domain-decomposition.

In 2022, work using WarpX was awarded the ACM Gordon Bell Prize demonstrating platform-independent scaling to the largest supercomputers in the world, including Frontier, and researching a novel particle-beam injection approach for laser-plasma acceleration [16].

ExaSky. The ExaSky ECP project is focused on simulation development intended for investigations of large-scale structure formation of the universe. Provided here are the performance results from the particle-based solver HACC (Hardware/Hybrid Accelerated Cosmology Code). Algorithmically, HACC integrates the gravitational Vlasov-Poisson equation using a spectral particle-mesh (PM) force solver in combination with direct particle interaction kernels [29]. To model gas physics and its coevolution with dark matter, HACC includes a modified *Smoothed Particle Hydrodynamics* (SPH) routine that utilizes higher-order reproducing kernels for accuracy [20, 22]. The hybrid software approach was designed to scale performantly on all modern supercomputing platforms, evolving trillions of particles on multi-core CPU and GPU configurations [15, 21, 28].

The performance of HACC on different machines is quantified by utilizing a figure of merit defined as the geometric mean of running simulations in gravity-only and hydrodynamic configurations. In simulations with gas, two fluids are sampled with particles, one representing dark matter interacting only gravitationally and the second, baryonic matter interacting both gravitationally and via gasdynamic forces; both are modeled with the same number of particles.

To establish a baseline measurement, HACC was run on the Theta supercomputer [39] using 3072 nodes and $n_p^3 = 2304^3$ particles. This measurement was rescaled to correspond to a full machine (4392 node) baseline. Simulations were run on 4096 nodes of each machine, and further weak-scaled up to 8192 nodes on Frontier. The simulation volume was adjusted to maintain consistent mass resolution for each run. One expects roughly a factor of two hardware single precision performance improvement between individual Summit and Frontier nodes, which is consistent with the 4096 node FOM measurements on both systems. Furthermore, HACC historically achieves near ideal weak-scaling up to millions of ranks [28], demonstrated here by the consistent timings between the 4096–8192 node Frontier runs. The target FOM improvement of 50× is far exceeded by the HACC runs presented.

EXAALT. The EXAALT project seeks to extend accuracy, length, and time scales of material science simulations using using molecular dynamics (MD) approaches. While EXAALT aims at being very general, initial targets focus on the simulation of defects in energy-relevant materials, including the first wall of fusion reactors as well as nuclear fuels. To achieve this goal, it integrates a replica-based accelerated MD method called Parallel Trajectory Splicing (ParSplice) [57] with the high-performance MD engine LAMMPS [65]. ParSplice is a time-wise parallelization technique that allows for long-timescale simulations on small systems, where traditional space-wise domain decomposition approaches become communication bound. The execution of the large number of MD instances required in ParSplice is orchestrated by a run-time environment called EXAALT that manages task execution as well as data caching, storage, and motion.

The simulations on Frontier used a variant of ParSplice called Sub-Lattice ParSplice, where domain decomposition is introduced so that each sub-domain can be accelerated separately. In contrast to traditional approaches, synchronization between domains is only needed when a topological transition occurs and not at every timestep. The simulation consisted of a surface of tungsten described by a SNAP (Spectral Neighborhood Analysis Potential) machine learning potential [66] in LAMMPS using the HIP backend of the Kokkos performance portability library [67]. The system contained 100,000 atoms in total, while each ParSplice replica contained 4000 atoms and ran on 4 MI250X GCDs, for a total of 13,856 instances of LAMMPS executing simultaneously on 7000 nodes (~75% of full Frontier).

The simulation sustained an extremely high throughput of 3.57×10^9 atom timestep/wall-clock second over the course of 1 hour of runtime. This corresponds to a 398.5x increase over the pre-ECP baseline obtained on Mira. This increase in performance was enabled by a ~25x performance increase on a single V100 GPU due to a near complete rewrite of the SNAP kernels and their optimization for GPUs, as well as by the increase in peak flop rate between Mira and Frontier. The details of the optimization process are described in [23, 44, 47].

ExaSMR. The ExaSMR project application combines continuous-energy Monte Carlo neutronics, including depletion, with computational fluid dynamics (CFD) to model the behavior of nuclear reactors. Specifically, a nonlinear Picard iteration scheme is used to converge the moderator temperature and densities in a coupled neutronics/CFD simulation [60]. The coupled driver uses the Shift [30] and OpenMC [61] Monte Carlo codes and the NekRS [18] CFD code, each of which has been optimized to run on multiple GPU architectures from NVIDIA, AMD, and Intel.

While the Monte Carlo and CFD application codes are general with respect to different nuclear reactor configurations and designs, the performance of these codes are demonstrated on a challenge problem that is a representative NuScale Small Modular Reactor (SMR) core. The challenge problem features a representative model of the complete in-vessel coolant loop, uses a Reynolds-Averaged-Navier-Stokes (RANS) turbulent model with an Large-Eddy-Simulation (LES) informed-momentum source for treatment of mixing vanes, and calculates pin-resolved spatial fission power and reaction rates in depleted fuel in the Monte Carlo simulation. The Monte Carlo part of the calculation tallies six different reactions in 213,860 spatial cells and simulates 51.2B particles per cycle over 40 eigenvalue cycles. The CFD model contained 1.098B unstructured mesh spatial elements and solved 376B degrees of freedom (DOF) over 1,500 timesteps.

The performance figure of merit (FOM) for this application is a harmonic average of the Monte Carlo and CFD work rates, particles and DOF per second of wall time, respectively. The maximum work rate of 912M particles/s for Shift was achieved in a non-coupled version of the challenge problem on 8,192 Frontier nodes with 8 ranks per node distributed across 4 MI250X AMD GPUs (1 rank per GCD). Shift also attained a weak-scaling efficiency of 97.8% from 1 to 8,192 nodes. The Monte Carlo/CFD coupled problem was run on 6,400 nodes of Frontier and also used 8 ranks per node. The Shift and NekRS FOMs for this simulation were 54 and 99.6 versus Titan, respectively, yielding a combined FOM of 70. The total runtime for the simulation was 2,556s (Shift) and 2,113s (NekRS).

5 FRONTIER AND THE EXASCALE REPORT

Going back to the exascale report [37], how does Frontier measure up to the authors' expectations and how well does it address their primary concerns? Before we discuss the four challenges, we have to confront their choice to ignore cost. They rightfully argued that exascale should not simply mean an exaflop as measured by HPL for the TOP500. Their expanded definition called for 1,000 times more resources (e.g., memory capacity and bandwidth, storage capacity and bandwidth). They then based their models and projections on such a richly resourced system and that drove many of the challenges.

While costs per unit have declined for storage in the past 15 years, they have not decreased by 1,000x. Some items have become more expensive such as memory. Large HPC systems have moved from DDR memory to much more performant but also much more expensive HBM. HBM pricing is opaque to the end user. We use a rule-of-thumb that HBM costs 3-5x more than top-of-the-line DDR, but that is simply a guess on our part. The cost per mm^2 silicon might actually be increasing given the extraordinary costs of the latest process nodes. Nor did DOE budgets increase by 1,000x. The overall budget limit set in the CORAL-2 Request for Proposals was 400-600 million US$ [13], 4-6x more than the definition of a supercomputer in 2008. Given this constraint, DOE had to set a different target for exascale.

In light of the above, DOE chose to focus on real application speedup. When writing the RFP, the reigning DOE systems were in the ~20 PF range and DOE set the target performance to 50x over those systems. The 50x could mean strong scaling (i.e., solve an existing problem 50x faster), weak scaling (i.e., solve a 50x larger problem in the same time), or some combination of the two. Within ECP, DOE selected 30 applications that represented a broad range of science domains and algorithms. In addition to the ECP applications, OLCF selected eight CAAR [55] applications that represented the large user allocation programs, INCITE and ALCC. These applications' speedups are measured against Summit because many did not exist before Titan and peers were decommissioned.

As discussed in Section 4.4, these applications are exceeding their target speedups, some significantly. If we measure success then by real application performance and not the arbitrary 1000x for all system resources, we argue that Frontier meets the spirit of the exascale definition.

5.1 Energy and Power

Frontier clearly excels in this area. Frontier debuted on the top of both the TOP500 and the Green500 on the June 2022 lists. [17, 64] This was unprecedented to have the largest system on the list also be the most energy efficient. This is a tribute to the AMD processors and HPE's overall design. Frontier's 1.1 EF using 21.1 MW gives an impressive 52 GF/watt, exceeding the report's 50 GF/watt target.

5.2 Memory and Storage

We estimate that memory alone accounts for over 25% of Frontier's cost. The storage, both node-local and Lustre, represents another ~15% of the cost. Together, memory and storage claim at least 40% of the system cost and place a limit on how much a system can have. Moving to HBM from DDR addressed the performance and power concerns of the exascale report. The authors considered flash media but disregarded it because it was too expensive, required too much power, and had a limited lifetime. Frontier's heterogeneous combination of flash for performance and hard drives for capacity meet the applications' needs within a reasonable budget.

5.3 Concurrency and Locality

The report's authors correctly identified the lack of frequency scaling and that exascale systems would need a vast amount of concurrency (e.g., 1 billion cores at 1 GHz). They also noted new developments such as GPUs, although they did not focus on them as a viable technology. Frontier's 37,632 MI250X GPUs with 220 Compute Units, with 64 threads each, provide over 500,000 threads operating close to 1 GHz. Each is able to perform two operations per cycle, meeting the target performance. GPUs' simpler consistency model, SIMD ability, and specialized run-times successfully addressed the challenge of concurrency while 2.5D packaging helped address their locality concerns.

5.4 Resiliency

While Frontier fares well in the above three challenges, it struggles with the resiliency challenge. The exascale report authors projected a Mean Time To Interrupt (MTTI) of 24 minutes for hardware and that a factor of 10x improvement in FIT rates would still incur a failure every four hours. They correctly identified memory and power supplies as leading contributors as we have seen on Frontier. Even with lower number of components due to larger processors configured as *fat nodes* and with smaller amounts of memory and storage due to cost, Frontier's resiliency is not much better than their projected four-hour target with the 10x improvement. The level of uncorrectable errors is in line with the rate seen on Summit's HBM2, once you scale up based on Frontier's HBM2e capacity. Power supplies continue to be a large source of upsets and HPE has a plan to mitigate this source of upsets. Over time, we expect Frontier's resiliency to increase and hopefully reach the levels of the first terascale systems, with failures on the order of 8-12 hours. [37]

6 CONCLUSION

In this paper, we presented Frontier's system architecture including hardware and software, evaluated micro-benchmarks, and demonstrated real application speedups. We presented this in the context of the 2008 DARPA report on the four primary exascale challenges. We do not provide a point-by-point discussion of the very detailed, exascale report, but two of the 2008 report's authors provide a very good analysis of Frontier's low-level technologies compared to the 2008 report's projections. [38] In their 2022 paper, they focus on HPCG as a better metric than HPL. In this paper, we focus on actual production applications, some dense and some sparse.

We highlighted at a high-level how Frontier's architecture addresses (or fails to address) their four critical challenges – energy/power, memory/storage, concurrency/locality, and resiliency. Given that a facility cannot ignore cost when deploying this class of system, we believe that the real application speedups demonstrate that Frontier's architecture meets the spirit of the exascale report.

REFERENCES

[1] L. Adhianto, S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey, and N. R. Tallent. 2010. HPCTOOLKIT: tools for performance analysis of optimized parallel programs. *Concurrency and Computation: Practice and Experience* 22, 6 (2010), 685–701. https://doi.org/10.1002/cpe.1553 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.1553

[2] AMD. 2020. Tuning Guide AMD EPYC 7003. https://www.amd.com/system/files/documents/high-performance-computing-tuning-guide-amd-epyc7003-series-processors.pdf. [Online; accessed 16-March-2023].

[3] AMD. 2021. CoralGemm - Matrix Multiply Stress Test. https://github.com/AMD-HPC/CoralGemm. [Online; accessed 30-March-2023].

[4] AMD. 2023. AMD ROCm Open Ecosystem. https://www.amd.com/en/graphics/servers-solutions-rocm. [Online; accessed 21-March-2023].

[5] AMD. 2023. Fundamentals of HIP Programming. https://www.amd.com/en/graphics/rocm-learning-center/fundamentals-of-hip-programming. [Online; accessed 21-March-2023].

[6] Pete Beckman, Ron Brightwell, Maya Gokhale, Bronis R. de Supinski, Steven Hofmeyr, Sriram Krishnamoorthy, Mike Lang, Barney Maccabe, John Shalf, and Marc Snir. 2012. *Exascale Operating Systems and Runtime Software Report.* Technical Report. DOE ASCR.

[7] OpenMP Architecture Review Board. 2018. *OpenMP Application Programming Interface Version 5.0.* Technical Report. OpenMP. https://www.openmp.org/wp-content/uploads/OpenMP-API-Specification-5.0.pdf, [Online; accessed 21-March-2023].

[8] OpenMP Architecture Review Board. 2020. *OpenMP Application Programming Interface Version 5.1.* Technical Report. OpenMP. https://www.openmp.org/wp-content/uploads/OpenMP-API-Specification-5-2.pdf, [Online; accessed 21-March-2023].

[9] OpenMP Architecture Review Board. 2021. *OpenMP Application Programming Interface Version 5.2.* Technical Report. OpenMP. https://www.openmp.org/wp-content/uploads/OpenMP-API-Specification-5-2.pdf, [Online; accessed 21-March-2023].

[10] M. Bussmann, H. Burau, T. E. Cowan, A. Debus, A. Huebl, G. Juckeland, T. Kluge, W. E. Nagel, R. Pausch, F. Schmitt, U. Schramm, J. Schuchart, and R. Widera. 2013. Radiative Signatures of the Relativistic Kelvin-Helmholtz Instability. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado) *(SC '13)*. Association for Computing Machinery, New York, NY, USA, Article 5, 12 pages. https://doi.org/10.1145/2503210.2504564

[11] Barbara Chapman, Buu Pham, Charlene Yang, Christopher Daley, Colleen Bertoni, Dhruva Kulkarni, Dossay Oryspayev, Ed D'Azevedo, Johannes Doerfert, Keren Zhou, Kiran Ravikumar, Mark Gordon, Mauro Del Ben, Meifeng Lin, Melisa Alkan, Michael Kruse, Oscar Hernandez, P. K. Yeung, Paul Lin, Peng Xu, Swaroop Pophale, Tosaporn Sattasathuchana, Vivek Kale, William Huhn, and Yun (Helen) He. 2021. Outcomes of OpenMP Hackathon: OpenMP Application Experiences with the Offloading Model (Part I). In *OpenMP: Enabling Massive Node-Level Parallelism: 17th International Workshop on OpenMP, IWOMP 2021, Bristol, UK, September 14–16, 2021, Proceedings* (Bristol, United Kingdom). Springer-Verlag, Berlin, Heidelberg, 67–80. https://doi.org/10.1007/978-3-030-85262-7_5

[12] Sudheer Chunduri, Taylor Groves, Peter Mendygral, Brian Austin, Jacob Balma, Krishna Kandalla, Kalyan Kumaran, Glenn Lockwood, Scott Parker, Steven Warren, Nathan Wichmann, and Nicholas Wright. 2019. GPCNeT: Designing a Benchmark Suite for Inducing and Measuring Contention in HPC Networks. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado) *(SC '19)*. Association for Computing Machinery, New York, NY, USA, Article 42, 33 pages. https://doi.org/10.1145/3295500.3356215

[13] DOE. 2018. CORAL-2 Request for Proposal (RFP). https://procurement.ornl.gov/rfp/CORAL2/01_CORAL-2_RFP%20LetterRev8.pdf. [Online; accessed 19-March-2023].

[14] DOE. 2023. Exascale Compute Project. https://www.exascaleproject.org. [Online; accessed 16-March-2023].

[15] J. D. Emberson, Nicholas Frontiere, Salman Habib, Katrin Heitmann, Patricia Larsen, Hal Finkel, and Adrian Pope. 2019. The Borg Cube Simulation: Cosmological Hydrodynamics with CRK-SPH. *The Astrophysical Journal* 877, 2 (may 2019), 85. https://doi.org/10.3847/1538-4357/ab1b31

[16] Luca Fedeli, Axel Huebl, France Boillod-Cerneux, Thomas Clark, Kevin Gott, Conrad Hillairet, Stephan Jaure, Adrien Leblanc, Rémi Lehe, Andrew Myers, Christelle Piechurski, Mitsuhisa Sato, Neïl Zaim, Weiqun Zhang, Jean-Luc Vay, and Henri Vincenti. 2022. Pushing the Frontier in the Design of Laser-Based Electron Accelerators with Groundbreaking Mesh-Refined Particle-In-Cell Simulations on Exascale-Class Supercomputers. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1109/SC41404.2022.00008

[17] Wu Feng and Kirk Cameron. 2022. Green500 June 2022 List. https://top500.org/lists/green500/2022/06/. [Online; accessed 19-March-2023].

[18] Paul Fischer, Stefan Kerkemeier, Misun Min, Yu-Hsiang Lan, Malachi Phillips, Thilina Rathnayake, Elia Merzari, Ananias Tomboulides, Ali Karakus, Noel Chalmers, and Tim Warburton. 2022. NekRS, a GPU-Accelerated Spectral Element Navier–Stokes Solver. *Parallel Comput.* 114, C (dec 2022), 13 pages. https://doi.org/10.1016/j.parco.2022.102982

[19] Free Software Foundation. 2023. GCC, the GNU Compiler Collection. https://gcc.gnu.org/. [Online; accessed 21-March-2023].

[20] Nicholas Frontiere, J. D. Emberson, Michael Buehlmann, Joseph Adamo, Salman Habib, Katrin Heitmann, and Claude-André Faucher-Giguère. 2023. Simulating Hydrodynamics in Cosmology with CRK-HACC. *The Astrophysical Journal Supplement Series* 264, 2 (jan 2023), 34. https://doi.org/10.3847/1538-4365/aca58d

[21] Nicholas Frontiere, Katrin Heitmann, Esteban Rangel, Patricia Larsen, Adrian Pope, Imran Sultan, Thomas Uram, Salman Habib, Silvio Rizzi, and Joe Insley. 2022. Farpoint: A High-resolution Cosmology Simulation at the Gigaparsec Scale. *The Astrophysical Journal Supplement Series* 259, 1 (feb 2022), 15. https://doi.org/10.3847/1538-4365/ac43b9

[22] Nicholas Frontiere, Cody D Raskin, and J Michael Owen. 2017. CRKSPH – A Conservative Reproducing Kernel Smoothed Particle Hydrodynamics Scheme. *J. Comput. Phys.* 332 (2017), 160–209.

[23] Rahulkumar Gayatri, Stan Moore, Evan Weinberg, Nicholas Lubbers, Sarah Anderson, Jack Deslippe, Danny Perez, and Aidan P. Thompson. 2020. Rapid Exploration of Optimization Strategies on Advanced Architectures using TestSNAP and LAMMPS. arXiv:2011.12875 [cs.DC]

[24] Balazs Gerofi, Masamichi Takagi, and Yutaka Ishikawa. 2019. IHK/McKernel. In *Operating Systems for Supercomputers and High Performance Computing*. Springer Singapore, Singapore. https://doi.org/10.1007/978-981-13-6624-6

[25] Philipp Grete. 2022. AthenaPK: a performance portable version of Athena++ built on Parthenon and Kokkos. https://github.com/parthenon-hpc-lab/athenapk. [Online; accessed 30-March-2023].

[26] Philipp Grete, Joshua C Dolence, Jonah M Miller, Joshua Brown, Ben Ryan, Andrew Gaspar, Forrest Glines, Sriram Swaminarayan, Jonas Lippuner, Clell J Solomon, Galen Shipman, Christoph Junghans, Daniel Holladay, James M Stone, and Luke F Roberts. 0. Parthenon—a performance portable block-structured adaptive mesh refinement framework. *The International Journal of High Performance Computing Applications* 0, 0 (0), 10943420221143775. https://doi.org/10.1177/10943420221143775

[27] GWT-TUD GmbH. 2023. Vampir - Performance Optimization. https://vampir.eu/. [Online; accessed 21-March-2023].

[28] Salman Habib, Vitali Morozov, Nicholas Frontiere, Hal Finkel, Adrian Pope, Katrin Heitmann, Kalyan Kumaran, Venkatram Vishwanath, Tom Peterka, Joe Insley, David Daniel, Patricia Fasel, and Zarija Lukić. 2016. HACC: Extreme Scaling and Performance Across Diverse Architectures. *Commun. ACM (Research Highlight)* 60, 1 (Dec. 2016), 97–104. https://doi.org/10.1145/3015569 Originally published in: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis,* page 6. ACM, 2013..

[29] Salman Habib, Adrian Pope, Hal Finkel, Nicholas Frontiere, Katrin Heitmann, David Daniel, Patricia Fasel, Vitali Morozov, George Zagaris, Tom Peterka, et al. 2016. HACC: Simulating sky surveys on state-of-the-art supercomputing architectures. *New Astronomy* 42 (2016), 49–65.

[30] Steven P. Hamilton and Thomas M. Evans. 2019. Continuous-energy Monte Carlo neutron transport on GPUs in the Shift code. *Annals of Nuclear Energy* 128 (2019), 236–247. https://doi.org/10.1016/j.anucene.2019.01.012

[31] Hewlett-Packard Enterprise. 2023. HPE Cray Programming Environment. https://www.hpe.com/psnow/doc/a50002303enw. [Online; accessed 21-March-2023].

[32] HPE. 2023. HPE Slingshot interconnect. https://www.hpe.com/us/en/compute/hpc/slingshot-interconnect.html. [Online; accessed 16-March-2023].

[33] Intel. 2023. oneAPI DPC++ compiler. https://github.com/intel/llvm. [Online; accessed 21-March-2023].

[34] Wayne Joubert, Deborah Weighill, David Kainer, Sharlee Climer, Amy Justice, Kjiersten Fagnan, and Daniel Jacobson. 2018. Attacking the Opioid Epidemic: Determining the Epistatic and Pleiotropic Genetic Architectures for Chronic Pain and Opioid Addiction. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*. Association for Computing Machinery, New York, NY, USA, 717–730. https://doi.org/10.1109/SC.2018.00060

[35] John Kim, Wiliam J Dally, Steve Scott, and Dennis Abts. 2008. Technology-Driven, Highly-Scalable Dragonfly Topology. *ACM SIGARCH Computer Architecture News* 36, 3 (2008), 77–88.

[36] Andreas Knüpfer, Christian Rössel, Dieter an Mey, Scott Biersdorff, Kai Diethelm, Dominic Eschweiler, Markus Geimer, Michael Gerndt, Daniel Lorenz, Allen Malony, Wolfgang E. Nagel, Yury Oleynik, Peter Philippen, Pavel Saviankou, Dirk Schmidl, Sameer Shende, Ronny Tschüter, Michael Wagner, Bert Wesarg, and Felix Wolf. 2012. Score-P: A Joint Performance Measurement Run-Time Infrastructure for Periscope,Scalasca, TAU, and Vampir. In *Tools for High Performance Computing 2011*, Holger Brunst, Matthias S. Müller, Wolfgang E. Nagel, and Michael M. Resch (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 79–91.

[37] Peter Kogge, Keren Bergman, Shekhar Borkar, Dan Campbell, William Carlson, William Dally, Monty Denneau, Paul Franzon, William Harrod, Kerry Hill, Jon Hiller, Sherman Karp, Stephen Keckler, Dean Klein, Robert Lucas, Mark Richards,

Al Scarpelli, Steven Scott, Allan Snavely, Thomas Sterling, R. Stanley Williams, and Katherine Yellick. 2008. *ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems*. Technical Report. AFRL.

[38] Peter M. Kogge and William J. Dally. 2022. Frontier vs the Exascale Report: Why so long? and Are We Really There Yet?. In *2022 IEEE/ACM International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*. IEEE, Piscataway, NJ, USA, 26–35. https://doi.org/10.1109/PMBS56514.2022.00008

[39] Argonne National Lab. 2017. Theta/ThetaGPU. https://www.alcf.anl.gov/alcf-resources/theta. [Online; accessed 31-March-2023].

[40] John Lagergren, Mikaela Cashman, Veronica Melesse Vergara, Paul Eller, Joao Gabriel Felipe Machado Gazolla, Hari Chhetri, Jared Streich, Sharlee Climer, Peter Thornton, Wayne Joubert, and Daniel Jacobson. 0. Climatic Clustering and Longitudinal Analysis with Impacts on Food, Bioenergy, and Pandemics. *Phytobiomes Journal* 0, ja (0), null. https://doi.org/10.1094/PBIOMES-02-22-0007-R

[41] Linaro Limited. 2023. Linaro Forge. https://www.linaroforge.com/. [Online; accessed 21-March-2023].

[42] LLVM. 2023. Flang (a.k.a. "classic" Flang). https://github.com/flang-compiler/flang. [Online; accessed 21-March-2023].

[43] LLVM. 2023. The LLVM Compiler Infrastructure. https://llvm.org/. [Online; accessed 21-March-2023].

[44] Susan M Mniszewski, James Belak, Jean-Luc Fattebert, Christian FA Negre, Stuart R Slattery, Adetokunbo A Adedoyin, Robert F Bird, Choongseok Chang, Guangye Chen, Stéphane Ethier, Shane Fogerty, Salman Habib, Christoph Junghans, Damien Lebrun-Grandié, Jamaludin Mohd-Yusof, Stan G Moore, Daniel Osei-Kuffuor, Steven J Plimpton, Adrian Pope, Samuel Temple Reeve, Lee Ricketson, Aaron Scheinberg, Amil Y Sharma, and Michael E Wall. 2021. Enabling particle applications for exascale computing platforms. *The International Journal of High Performance Computing Applications* 35, 6 (2021), 572–597.

[45] A. Moody. 2009. Contention-free Routing for Shift-based Communication in MPI Applications on Large-scale Infiniband Clusters. https://doi.org/10.2172/967277

[46] Timothy Prickett Morgan. 2022. Cray'S Slingshot Interconnect is at the Heart of HPE'S HPC and AI Ambitions. https://www.nextplatform.com/2022/01/31/crays-slingshot-interconnect-is-at-the-heart-of-hpes-hpc-and-ai-ambitions/. [Online; accessed 16-March-2023].

[47] Kien Nguyen-Cong, Jonathan T. Willman, Stan G. Moore, Anatoly B. Belonoshko, Rahulkumar Gayatri, Evan Weinberg, Mitchell A. Wood, Aidan P. Thompson, and Ivan I. Oleynik. 2021. Billion Atom Molecular Dynamics Simulations of Carbon at Extreme Conditions and Experimental Time and Length Scales. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (St. Louis, Missouri) *(SC '21)*. Association for Computing Machinery, New York, NY, USA, Article 4, 12 pages. https://doi.org/10.1145/3458817.3487400

[48] NVIDIA. 2023. CUDA Toolkit. https://developer.nvidia.com/cuda-toolkit. [Online; accessed 21-March-2023].

[49] OpenACC. 2013. *The OpenACC Application Programming Interface Version 2.0*. Technical Report. OpenACC-Standard.org. https://www.openacc.org/sites/default/files/inline-files/OpenACC_2_0_specification.pdf, [Online; accessed 21-March-2023].

[50] openACC. 2017. *The OpenACC Application Programming Interface Version 2.6*. Technical Report. OpenACC-Standard.org. https://www.openacc.org/sites/default/files/inline-files/OpenACC_2_0_specification.pdf, [Online; accessed 21-March-2023].

[51] openACC. 2018. *The OpenACC Application Programming Interface Version 2.7*. Technical Report. OpenACC-Standard.org. https://www.openacc.org/sites/default/files/inline-files/OpenACC.2.7.pdf, [Online; accessed 21-March-2023].

[52] openACC. 2021. *The OpenACC Application Programming Interface Version 3.2*. Technical Report. OpenACC-Standard.org. https://www.openacc.org/sites/default/files/inline-images/Specification/OpenACC-3.2-final.pdf, [Online; accessed 21-March-2023].

[53] OpenACC. 2023. OpenACC. https://www.openacc.org/. [Online; accessed 21-March-2023].

[54] OpenMP. 2023. OpenMP. https://www.openmp.org/. [Online; accessed 21-March-2023].

[55] ORNL. 2018. Frontier Center for Accelerated Application Readiness (CAAR). https://www.olcf.ornl.gov/caar/frontier-caar/. [Online; accessed 19-March-2023].

[56] Jiannan Ouyang, Brian Kocoloski, John R. Lange, and Kevin Pedretti. 2015. Achieving Performance Isolation with Lightweight Co-Kernels. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing* (Portland, Oregon, USA) *(HPDC '15)*. Association for Computing Machinery, New York, NY, USA, 149–160. https://doi.org/10.1145/2749246.2749273

[57] Danny Perez, Ekin D Cubuk, Amos Waterland, Efthimios Kaxiras, and Arthur F Voter. 2016. Long-time dynamics through parallel trajectory splicing. *Journal of chemical theory and computation* 12, 1 (2016), 18–28.

[58] Kiran Ravikumar, David Appelhans, and P. K. Yeung. 2019. GPU Acceleration of Extreme Scale Pseudo-Spectral Simulations of Turbulence Using Asynchronism. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado) *(SC '19)*. Association for Computing Machinery, New York, NY, USA, Article 8, 22 pages. https://doi.org/10.1145/3295500.3356209

[59] Rice University. 2023. HPCToolkit. http://hpctoolkit.org/. [Online; accessed 21-March-2023].

[60] Paul K. Romano, Steven P. Hamilton, Ronald O. Rahaman, April Novak, Elia Merzari, Sterling M. Harper, Patrick C. Shriwise, and Thomas M. Evans. 2021. A Code-Agnostic Driver Application for Coupled Neutronics and Thermal-Hydraulic Simulations. *Nuclear Science and Engineering* 195, 4 (2021), 391–411. https://doi.org/10.1080/00295639.2020.1830620

[61] Paul K. Romano, Nicholas E. Horelik, Bryan R. Herman, Adam G. Nelson, Benoit Forget, and Kord Smith. 2015. OpenMC: A state-of-the-art Monte Carlo code for research and development. *Annals of Nuclear Energy* 82 (2015), 90–97. https://doi.org/10.1016/j.anucene.2014.07.048 Joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2013, SNA + MC 2013. Pluri- and Trans-disciplinarity, Towards New Modeling and Numerical Simulation Paradigms.

[62] Sameer S. Shende and Allen D. Malony. 2006. The Tau Parallel Performance System. *The International Journal of High Performance Computing Applications* 20, 2 (2006), 287–311. https://doi.org/10.1177/1094342006064482 arXiv:https://doi.org/10.1177/1094342006064482

[63] James M. Stone, Kengo Tomida, Christopher J. White, and Kyle G. Felker. 2020. The Athena++ Adaptive Mesh Refinement Framework: Design and Magnetohydrodynamic Solvers. *The Astrophysical Journal Supplement Series* 249, 1 (jun 2020), 4. https://doi.org/10.3847/1538-4365/ab929b

[64] Erich Strohmaier, Jack Dongarra, Horst Simon, and Martin Meuer. 2022. TOP500 June 2022 List. https://top500.org/lists/top500/2022/06/. [Online; accessed 19-March-2023].

[65] Aidan P Thompson, H Metin Aktulga, Richard Berger, Dan S Bolintineanu, W Michael Brown, Paul S Crozier, Pieter J in't Veld, Axel Kohlmeyer, Stan G Moore, Trung Dac Nguyen, et al. 2022. LAMMPS-a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Computer Physics Communications* 271 (2022), 108171.

[66] Aidan P Thompson, Laura P Swiler, Christian R Trott, Stephen M Foiles, and Garritt J Tucker. 2015. Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials. *J. Comput. Phys.* 285 (2015), 316–330.

[67] Christian R. Trott, Damien Lebrun-Grandié, Daniel Arndt, Jan Ciesko, Vinh Dang, Nathan Ellingwood, Rahulkumar Gayatri, Evan Harvey, Daisy S. Hollman, Dan Ibanez, Nevin Liber, Jonathan Madsen, Jeff Miles, David Poliakoff, Amy Powell, Sivasankaran Rajamanickam, Mikael Simberg, Dan Sunderland, Bruno Turcksin, and Jeremiah Wilke. 2022. Kokkos 3: Programming Model Extensions for the Exascale Era. *IEEE Transactions on Parallel and Distributed Systems* 33, 4 (2022), 805–817. https://doi.org/10.1109/TPDS.2021.3097283

[68] University of Oregon. 2023. Tuning and Analysis Utilities. https://www.cs.uoregon.edu/research/tau/home.php. [Online; accessed 21-March-2023].

[69] J.-L. Vay, A. Huebl, A. Almgren, L. D. Amorim, J. Bell, L. Fedeli, L. Ge, K. Gott, D. P. Grote, M. Hogan, R. Jambunathan, R. Lehe, A. Myers, C. Ng, M. Rowan, O. Shapoval, M. Thévenet, H. Vincenti, E. Yang, N. Zaïm, W. Zhang, Y. Zhao, and E. Zoni. 2021. Modeling of a chain of three plasma accelerator stages with the WarpX electromagnetic PIC code on GPUs. *Physics of Plasmas* 28, 2 (2021), 023105. https://doi.org/10.1063/5.0028512

[70] Sudharshan S. Vazhkudai, Bronis R. de Supinski, Arthur S. Bland, Al Geist, James Sexton, Jim Kahle, Christopher J. Zimmer, Scott Atchley, Sarp Oral, Don E. Maxwell, Veronica G. Vergara Larrea, Adam Bertsch, Robin Goldstone, Wayne Joubert, Chris Chambreau, David Appelhans, Robert Blackmore, Ben Casses, George Chochia, Gene Davison, Matthew A. Ezell, Tom Gooding, Elsa Gonsiorowski, Leopold Grinberg, Bill Hanson, Bill Hartner, Ian Karlin, Matthew L. Leininger, Dustin Leverman, Chris Marroquin, Adam Moody, Martin Ohmacht, Ramesh Pankajakshan, Fernando Pizzano, James H. Rogers, Bryan Rosenburg, Drew Schmidt, Mallikarjun Shankar, Feiyi Wang, Py Watson, Bob Walkup, Lance D. Weems, and Junqi Yin. 2018. The Design, Deployment, and Evaluation of the CORAL Pre-Exascale Systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis* (Dallas, Texas) *(SC '18)*. IEEE Press, Piscataway, NJ, USA, Article 52, 12 pages. http://dl.acm.org/citation.cfm?id=3291656.3291726

[71] Virtual Institute - High Productivity Supercomputing. 2023. SCORE-P: Scalable Performance Measurement Infrastructure for Parallel Codes. https://www.vi-hps.org/projects/score-p/. [Online; accessed 21-March-2023].

[72] Robert W. Wisniewski, Todd Inglett, Pardo Keppel, Ravi Murty, and Rolf Riesen. 2014. MOS: An Architecture for Extreme-Scale Operating Systems. In *Proceedings of the 4th International Workshop on Runtime and Operating Systems for Supercomputers* (Munich, Germany) *(ROSS '14)*. Association for Computing Machinery, New York, NY, USA, Article 2, 8 pages. https://doi.org/10.1145/2612262.2612263

[73] Christopher Zimmer, Scott Atchley, Ramesh Pankajakshan, Brian E. Smith, Ian Karlin, Matthew L. Leininger, Adam Bertsch, Brian S. Ryujin, Jason Burmark, André Walker-Loud, M. A. Clark, and Olga Pearce. 2019. An Evaluation of the CORAL Interconnects. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado)

(SC '19). Association for Computing Machinery, New York, NY, USA, Article 39, 18 pages. https://doi.org/10.1145/3295500.3356166