FUN3D v13.4 Training

Session 20:
Running on GPU Systems

Eric Nielsen

Learning Goals

What we will cover

- Some GPU basics
- Running across multiple GPUs
- FUN3D performance studies
- Supported options
- Inputs
- Trial hardware

Some GPU Basics

- With the end of Moore's Law and increasing power constraints in the data center, hardware is trending towards high concurrency (parallelism) to achieve further performance: we are in a paradigm shift towards *many-core* architectures
- Graphics processing units, or GPUs, offer extremely high concurrency and have become increasingly common on large HPC systems
  - The most powerful system in the world and 5 of the top 10 rely on GPUs
- GPUs can be used to accelerate computationally-intensive kernels
- The NVIDIA Volta V100 offers ~900 GB/s memory bandwidth
  - Most large-scale science codes are memory-bound, so this a critical feature
  - However, there is generally far less memory available than with a typical CPU: Volta V100 comes with 32 GB

Some GPU Basics

- Common high-level languages such as Fortran, C, C++ cannot be used directly on GPUs
- Many programming models exist; some may be more appropriate/useful than others depending on the application
  - OpenACC, OpenMP (compiler directives)
  - CUDA Fortran (Portland Group)
  - CUDA C/C++, PTX (NVIDIA)
  - Kokkos (Sandia National Laboratory)
- We have explored each of these models over the past 7-8 years and currently use CUDA C/C++
  - Highly flexible, low-level control; essential for achieving performance of complex unstructured-grid kernels

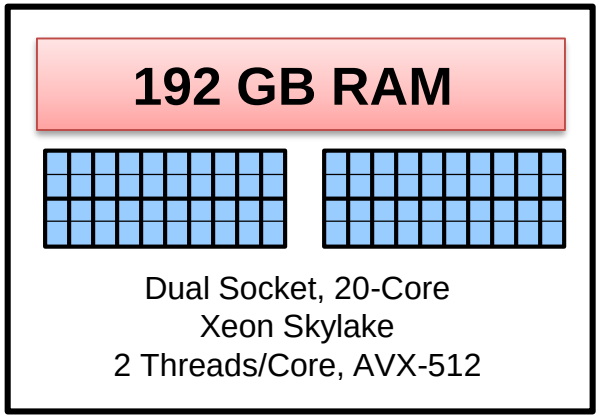
Some GPU Basics

- For FUN3D, we recommend NVIDIA Pascal P100 or Volta V100 hardware
  - Critical hardware features important to FUN3D performance introduced in these generations
  - NVIDIA Kepler K20/K40/K80 will outperform CPUs of that era, but not more recent processors
- GPUs still require a CPU, or host, to run the OS, applications, etc
  - Data movement between the CPU and GPU can be highly detrimental to performance
  - The GPU option in FUN3D uses the CPU to route all kernels to the GPU during execution with essentially no host/device data movement
- It is increasingly common to find multiple GPUs attached to a single CPU in new systems
  - Amortize the cost of much of the node infrastructure over multiple devices
  - Xeon-based systems often offer 4 GPUs per host; IBM Power systems offer 4 or 6; NVIDIA DGX systems offer as many as 16 GPUs per host

Some GPU Basics

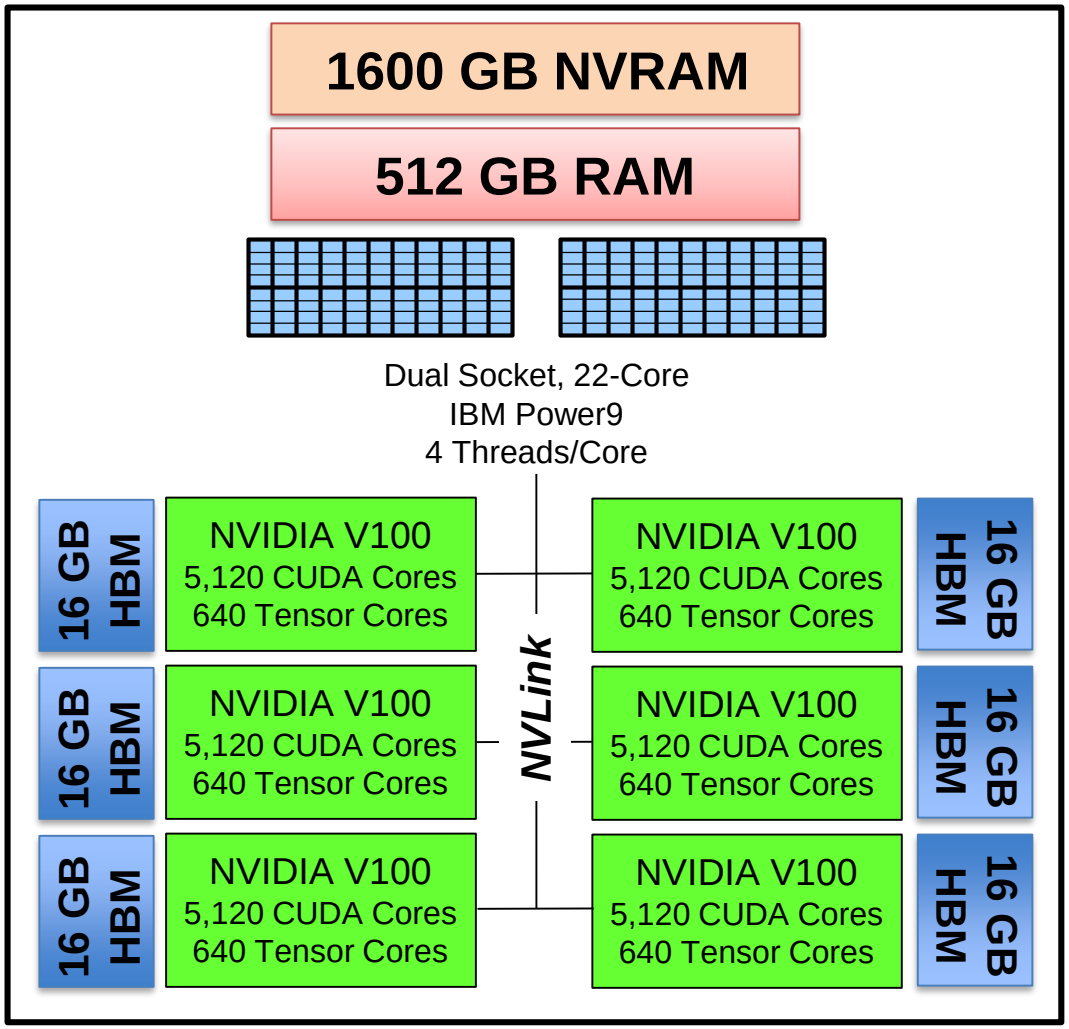
Typical x86 Node

~1 Teraflop



ORNL Summit Node

~40 Teraflops

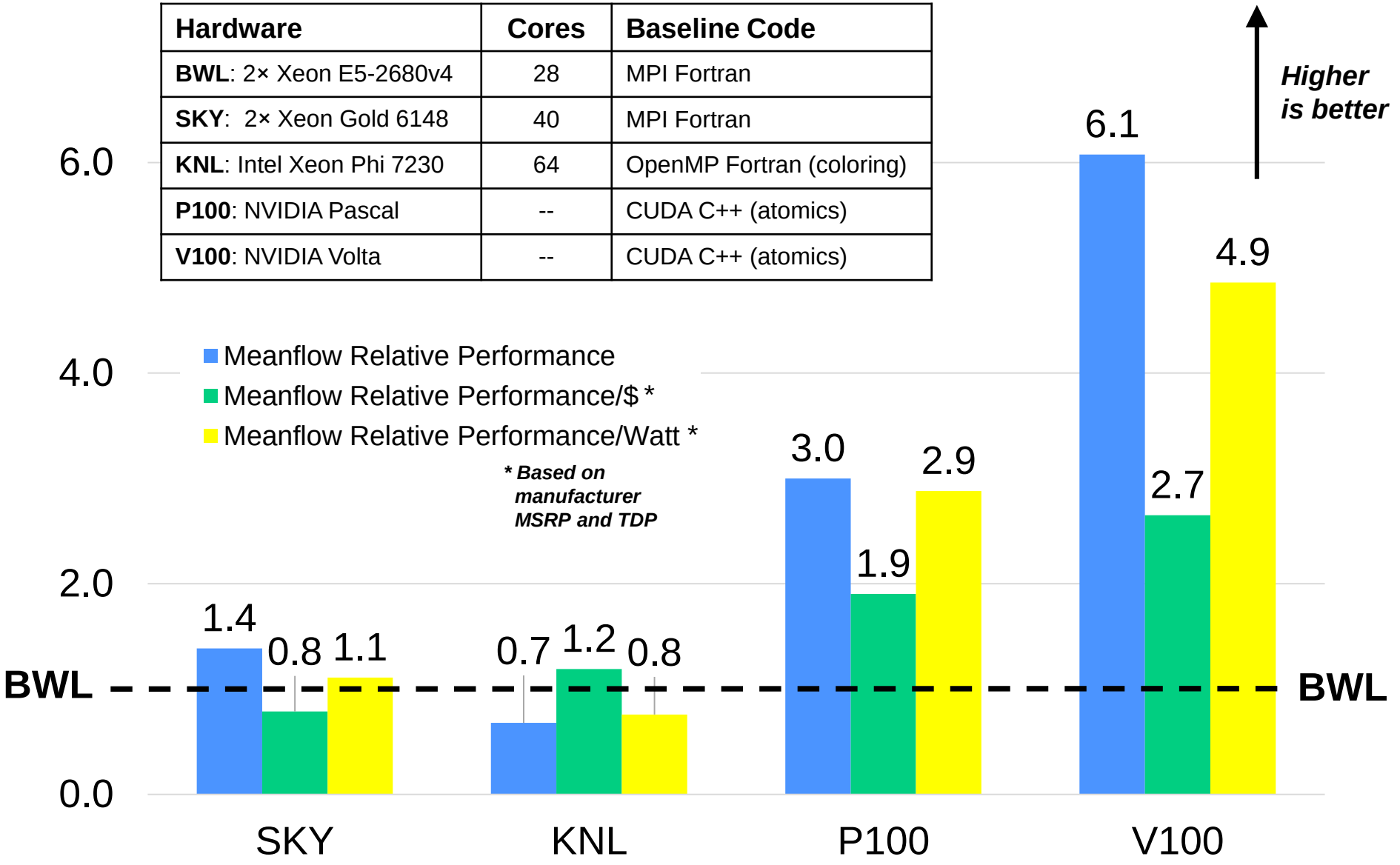


Running Across Multiple GPUs

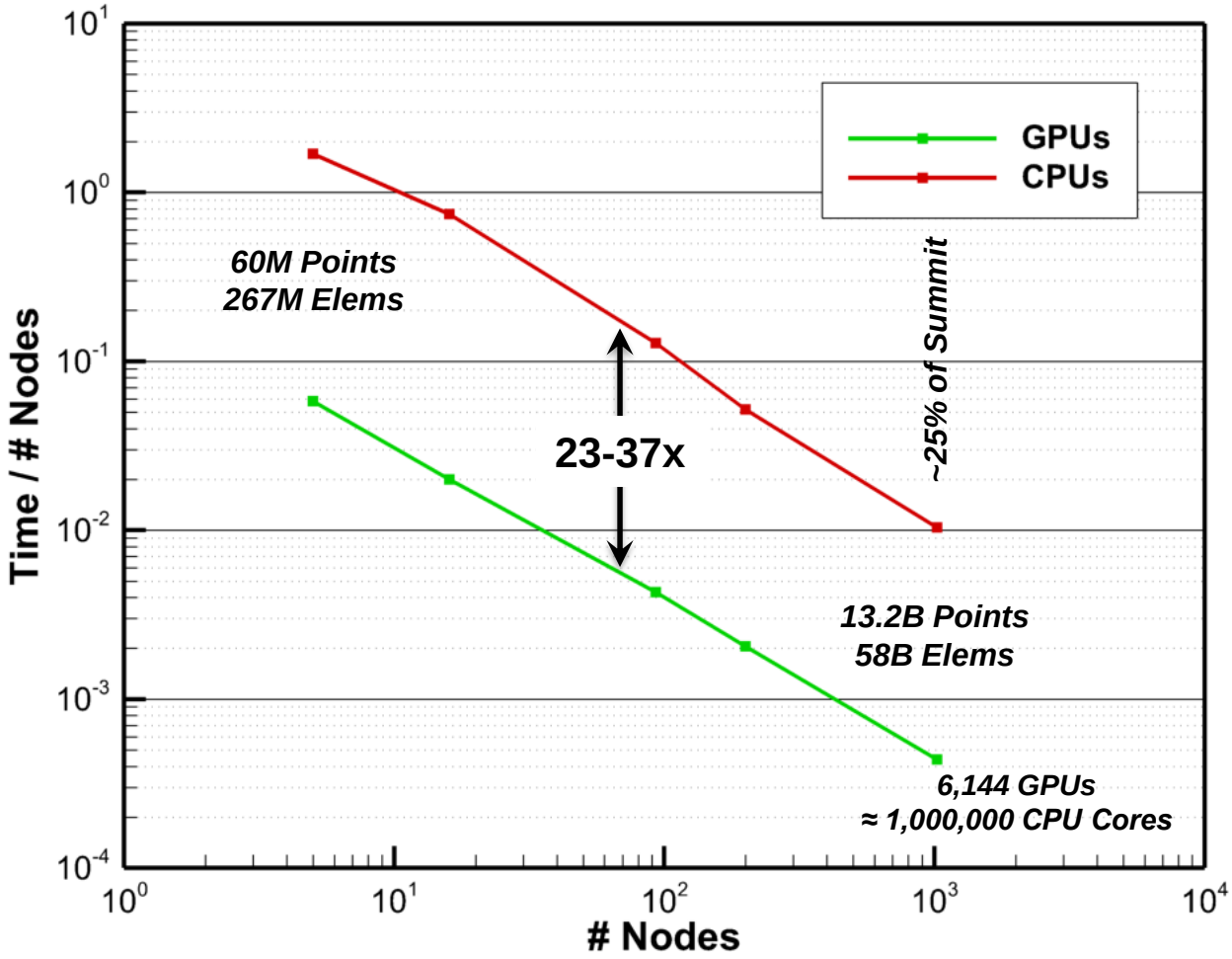
- The current approach for FUN3D places a single MPI rank on a CPU core targeted for a single GPU
  - That CPU core will simply babysit the GPU doing all of the work
  - So a 40-core Skylake host may only have 4 MPI ranks spread across 4 CPU cores if 4 GPUs are available on the node; all other CPU cores idle
  - Consult your MPI documentation for CPU binding strategies
- With multiple GPUs per CPU, the job launcher may automatically assign GPUs to different MPI ranks, or it may not
  - Speak with your system administrator
  - If not automatically assigned, you will have to tell FUN3D to do it
- A *CUDA-enabled* MPI installation is preferred
  - This allows MPI calls to recognize GPU memory addresses
  - Ideally, this will enable MPI messages to route directly between GPUs without having to stage through host memory; however, this depends on other aspects of your software stack and hardware – again, speak with your system administrator
  - FUN3D also offers optimized host-based MPI calls with only a minor performance hit

FUN3D Performance Studies

Hardware	Cores	Baseline Code
BWL: 2× Xeon E5-2680v4	28	MPI Fortran
SKY: 2× Xeon Gold 6148	40	MPI Fortran
KNL: Intel Xeon Phi 7230	64	OpenMP Fortran (coloring)
P100: NVIDIA Pascal	--	CUDA C++ (atomics)
V100: NVIDIA Volta	--	CUDA C++ (atomics)



FUN3D Performance Studies



- Weak scaling study on Summit shows ~30x speedup at scale

Status and Supported Options

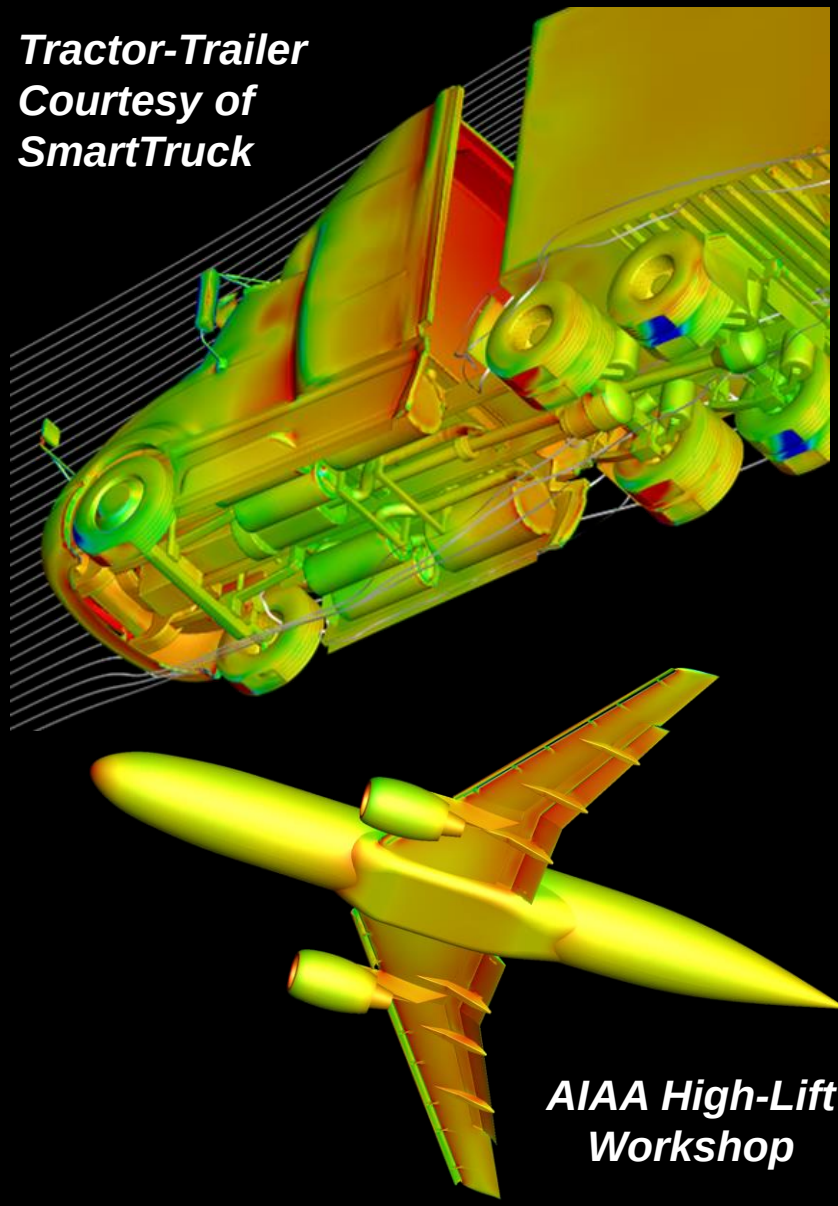
- Not all of FUN3D has been ported to the GPU at this time, but rather the most commonly-used options
- Did not make the v13.4 release, but looking for early adopters to try and help identify gaps/issues
  - We will provide you with an additional library to build FUN3D against that contains all of the necessary FUN3D CUDA kernels
- Approximately 100 kernels have been ported to enable compressible perfect gas simulations including SA-based RANS, URANS, and DES variants
- Supported options
  - Roe FDS
  - Mixed element grids only; use '—mixed' CLO if using a tetrahedral grid
  - h-van Albada, Barth limiters
  - Boundary conditions: 3000, 4000 (fixed wall temperature and adiabatic), 5000, 6662, 7011; specified surface velocities
  - All visualization options are available, but are performed on the host: frequent use will limit performance
  - Other features could be added upon request
- Ask if an option is available before attempting to use

Early GPU Applications

Computations Performed on Titan at the Oak Ridge Leadership Computing Facility

Supersonic Retropropulsion

Tractor-Trailer Courtesy of SmartTruck



AIAA High-Lift Workshop

Inputs and Default Settings

```
&gpu_support
  use_cuda       = .false. ! Instructs FUN3D to use GPUs
  use_cuda_mpi   = .true.  ! CUDA-enabled MPI is available
  use_setdevice  = .false. ! Assigns devices via rank % gpus_per_node
  gpus_per_node  = 1       ! Number of GPUs available per node
/
```

Launching MPI Jobs

- Syntax used to launch MPI jobs with GPU support varies by system
  - Manual setting of environment variables often necessary, especially for leveraging CUDA-aware MPI implementations
  - May need to bind to different network cards
  - May need explicit –hostfile $PBS_NODEFILE
  - Consult system documentation or administrator; feel free to contact us for help too

Cray XK7

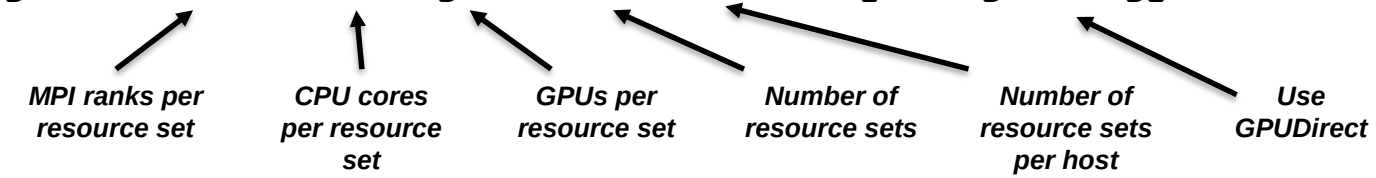
```
aprun –n 16 –N 1 ./a.out
```

IBM POWER system with Spectrum MPI, Xeon-based cluster with OpenMPI

```
mpirun –np 16 –npersocket 2 --bind-to core ./a.out
```

IBM POWER system with Spectrum MPI and jsrun launcher

```
jsrun -a2 -c2 -g2 -n2 -r2 --smpiargs "-gpu" ./a.out
```



Trial Hardware

- Most organizations probably do not have the hardware necessary to explore GPU-enabled FUN3D
- See NVIDIA representative available here on-site
- Speak with your favorite vendor about loaner options or remote login opportunities on trial systems
- Generally very difficult to receive approval for cloud-based services; work must be under a government contract and may still be a challenge
- Oak Ridge National Laboratory: Contact Suzy Tichenor, Director, HPC Industrial Partnerships Program, tichenorsp@ornl.gov
- Consult with Langley Software Release Authority for IT restrictions relevant to your Software Usage Agreement: Bonnie Lumanog, b.lumanog@nasa.gov

What We Learned

- GPU basics
- FUN3D performance
- Supported options and execution instructions
- Trial hardware opportunities

Looking for early adopters!

fun3d-support@lists.nasa.gov