High-End Computing (HEC): PetaFlop to ExaFlop

100 PFlops in 2017

1 EFlops in 2020-2021?

143 PFlops in 2018

Expected to have an ExaFlop system in 2020-2021!
Drivers of Modern HPC Cluster Architectures

- Multi-core/many-core technologies
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- Solid State Drives (SSDs), Non-Volatile Random-Access Memory (NVRAM), NVMe-SSD
- Accelerators (NVIDIA GPGPUs and Intel Xeon Phi)
- Available on HPC Clouds, e.g., Amazon EC2, NSF Chameleon, Microsoft Azure, etc.
Supporting Programming Models for Multi-Petaflop and Exaflop Systems: Challenges

Application Kernels/Applications (HPC and DL)

Middleware

Programming Models
MPI, PGAS (UPC, Global Arrays, OpenSHMEM), CUDA, OpenMP, OpenACC, Cilk, Hadoop (MapReduce), Spark (RDD, DAG), etc.

Communication Library or Runtime for Programming Models
- Point-to-point Communication
- Collective Communication
- Energy-Awareness
- Synchronization and Locks
- I/O and File Systems
- Fault Tolerance

Networking Technologies
(InfiniBand, 40/100/200GigE, Slingshot, and Omni-Path)

Multi-/Many-core Architectures

Accelerators (GPU and FPGA)

Co-Design Opportunities and Challenges across Various Layers
Performance
Scalability
Resilience
Designing (MPI+X) at Exascale

- Scalability for million to billion processors
  - Support for highly-efficient inter-node and intra-node communication (both two-sided and one-sided)
  - Scalable job start-up
  - Low memory footprint

- Scalable Collective communication
  - Offload
  - Non-blocking
  - Topology-aware

- Balancing intra-node and inter-node communication for next generation nodes (128-1024 cores)
  - Multiple end-points per node

- Support for efficient multi-threading

- Integrated Support for Accelerators (GPGPUs and FPGAs)

- Fault-tolerance/resiliency

- QoS support for communication and I/O

- Support for Hybrid MPI+PGAS programming (MPI + OpenMP, MPI + UPC, MPI + OpenSHMEM, MPI+UPC++, CAF, ...)

- Virtualization

- Energy-Awareness
Overview of the MVAPICH2 Project

- High Performance open-source MPI Library for InfiniBand, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.1), Started in 2001, First version available in 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2011
  - Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2014
  - Support for Virtualization (MVAPICH2-Virt), Available since 2015
  - Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
  - Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
- Used by more than 3,050 organizations in 89 countries
- More than 615,000 (> 0.6 million) downloads from the OSU site directly
- Empowering many TOP500 clusters (Jun ‘19 ranking)
  - 3rd, 10,649,600-core (Sunway TaihuLight) at National Supercomputing Center in Wuxi, China
  - 5th, 448,448 cores (Frontera) at TACC
  - 8th, 391,680 cores (ABCI) in Japan
  - 15th, 570,020 cores (Neurion) in South Korea and many others
- Available with software stacks of many vendors and Linux Distros (RedHat, SuSE, and OpenHPC)
- [http://mvapich.cse.ohio-state.edu](http://mvapich.cse.ohio-state.edu) Partner in the TACC Frontera System
- Empowering Top500 systems for over a decade
MVAPICH2 Release Timeline and Downloads

Number of Downloads
# Architecture of MVAPICH2 Software Family (HPC and DL)

## High Performance Parallel Programming Models

- **Message Passing Interface (MPI)**
- **PGAS** (UPC, OpenSHMEM, CAF, UPC++)
- **Hybrid --- MPI + X** (MPI + PGAS + OpenMP/Cilk)

## High Performance and Scalable Communication Runtime

### Diverse APIs and Mechanisms

- Point-to-point Primitives
- Collectives Algorithms
- Job Startup
- Energy-Awareness
- Remote Memory Access
- I/O and File Systems
- Fault Tolerance
- Virtualization
- Active Messages
- Introspection & Analysis

### Support for Modern Networking Technology

(InfraBand, iWARP, RoCE, Omni-Path, Elastic Fabric Adapter)

- **Transport Protocols**: RC, SRD, UD, DC
- **Modern Features**: UMR, ODP, SR-IOV, Multi Rail

### Support for Modern Multi-/Many-core Architectures

(Intel-Xeon, OpenPOWER, Xeon-Phi, ARM, NVIDIA GPGPU)

- **Transport Mechanisms**: Shared Memory, CMA, IVSHMEM, XPMEM
- **Modern Features**: Optane*, NVLink, CAPI*

* Upcoming
## MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP, Omni-Path, and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI Features/Support, OSU INAM, PGAS and MPI+PGAS with IB, Omni-Path, and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB, RoCE &amp; GPU and Support for Deep Learning</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>HPC Cloud with MPI &amp; IB</td>
<td>MVAPICH2-Virt</td>
</tr>
<tr>
<td>Energy-aware MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2-EA</td>
</tr>
<tr>
<td>MPI Energy Monitoring Tool</td>
<td>OEMT</td>
</tr>
<tr>
<td>InfiniBand Network Analysis and Monitoring</td>
<td>OSU INAM</td>
</tr>
<tr>
<td>Microbenchmarks for Measuring MPI and PGAS Performance</td>
<td>OMB</td>
</tr>
</tbody>
</table>
Features and Improvement in MAVPIACH2-X for ARM

- Enhanced architecture and IB HCA detection for various ARM systems
- Optimization and tuning for
  - Intra-node and inter-node point-to-point operations
  - Intra-node shared memory communication protocols
  - Collective operations for different message sizes and job/system sizes using the existing collective algorithms in MVAPICH2-X
- Optimizations to job startup performance to achieve scalable job startup when running large-scale jobs on ARM systems
- Support for latest GCC and ARM compilers
Performance Evaluation of Optimized MVAPICH2-X

• **EPCC Fulhame Cluster**
  - Nodes: 16 x ARM ThunderX2
  - Processor: 2x 32 core ARM ThunderX2
  - Network: EDR 100Gbps MT4119
  - MPI and Communication Libraries
    - MVAPICH2-X (latest)
    - HPCX-v2.4.0-gcc-MLNX_OFED_LINUX-4.6-1.0.1.1-suse15.0-aarch64
    - OpenMPI-4.0.2 w/ latest UCX
  - OSU-Microbenchmarks-v5.6.2

• **Mayer Cluster**
  - Nodes: 14 x ARM ThunderX2
  - Processor: 2x 28 core ARM ThunderX2
  - Network: EDR 100Gbps MT4119
  - MPI and Communication Libraries
    - MVAPICH2-X (latest)
    - OpenMPI 4.0.1
    - UCX 1.5.2
  - OSU-Microbenchmarks-v5.6.2
Evaluation of Point-to-point on EPCC ARM System

- EPCC Fulhame ARM cluster with up to 16 dual-socket 32-core ThunderX2 nodes
- Comparison among MVAPICH2X (Next), OpenMPI+UCX, and HPCX communication libraries
- OSU Micro-benchmark Suite (OMB) v5.6.2
- Measure the MPI-level communication performance of latency, bandwidth, bi-directional bandwidth, and message rate
- Three different configurations
  - Intra-socket
  - Inter-socket
  - Inter-node
Point-to-point: Latency & Bandwidth (Intra-socket)

Latency - Small Messages
- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Latency - Medium Messages
- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Latency - Large Messages
- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Bandwidth - Small Messages
- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Bandwidth - Medium Messages
- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Bandwidth - Large Messages
- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

70% better
Point-to-point: Bi-Bandwidth (Intra-socket)

Bi-bandwidth - Small Messages

- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Bi-bandwidth - Medium Messages

- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Bi-bandwidth - Large Messages

37% better

Message Rate - Small Messages

- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

Message Rate - Medium Messages

Message Rate - Large Messages

Network Based Computing Laboratory
ARM-HUG (SC '19)
Point-to-point: Latency & Bandwidth (Inter socket)

Latency - Small Messages

Latency - Medium Messages

Latency - Large Messages

Bandwidth - Small Messages

Bandwidth - Medium Messages

Bandwidth - Large Messages

Network Based Computing Laboratory

ARM-HUG (SC ’19)
Point-to-point: Bi-Bandwidth (Inter-socket)

**Bi-bandwidth - Small Messages**

- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

**Message Size (Bytes)**

<table>
<thead>
<tr>
<th>1</th>
<th>4</th>
<th>16</th>
<th>64</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>100</td>
<td>200</td>
<td>300</td>
</tr>
</tbody>
</table>

**Bandwidth (MB/s)**

<table>
<thead>
<tr>
<th>0</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>400</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>100</td>
<td>200</td>
<td>300</td>
<td>400</td>
</tr>
</tbody>
</table>

**Message Rate - Small Messages**

<table>
<thead>
<tr>
<th>1</th>
<th>4</th>
<th>16</th>
<th>64</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

**Message Size (Bytes)**

<table>
<thead>
<tr>
<th>1</th>
<th>4</th>
<th>16</th>
<th>64</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

**Million Message / Sec**

<table>
<thead>
<tr>
<th>0</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
</tr>
</tbody>
</table>

**Bi-bandwidth - Medium Messages**

- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

**Message Size (Bytes)**

<table>
<thead>
<tr>
<th>256</th>
<th>1K</th>
<th>4K</th>
<th>16K</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1200</td>
<td>2400</td>
<td>3600</td>
</tr>
</tbody>
</table>

**Bandwidth (MB/s)**

<table>
<thead>
<tr>
<th>0</th>
<th>3000</th>
<th>6000</th>
<th>9000</th>
<th>12000</th>
<th>16000</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3000</td>
<td>6000</td>
<td>9000</td>
<td>12000</td>
<td>16000</td>
</tr>
</tbody>
</table>

**Message Rate - Medium Messages**

<table>
<thead>
<tr>
<th>256</th>
<th>1K</th>
<th>4K</th>
<th>16K</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1</td>
<td>1.5</td>
<td>2</td>
</tr>
</tbody>
</table>

**Message Size (Bytes)**

<table>
<thead>
<tr>
<th>64K</th>
<th>256K</th>
<th>1M</th>
<th>4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.04</td>
<td>0.08</td>
<td>0.12</td>
</tr>
</tbody>
</table>

**Million Message / Sec**

<table>
<thead>
<tr>
<th>64K</th>
<th>256K</th>
<th>1M</th>
<th>4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.04</td>
<td>0.08</td>
<td>0.12</td>
</tr>
</tbody>
</table>

**Bi-bandwidth - Large Messages**

- MVAPICH2-X-Next
- HPCX
- OpenMPI+UCX

**Message Size (Bytes)**

<table>
<thead>
<tr>
<th>64K</th>
<th>256K</th>
<th>1M</th>
<th>4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>24% better</td>
<td>24% better</td>
<td>24% better</td>
<td>24% better</td>
</tr>
</tbody>
</table>

**Bandwidth (MB/s)**

<table>
<thead>
<tr>
<th>64K</th>
<th>256K</th>
<th>1M</th>
<th>4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4000</td>
<td>8000</td>
<td>12000</td>
</tr>
</tbody>
</table>

**Message Rate - Large Messages**

<table>
<thead>
<tr>
<th>64K</th>
<th>256K</th>
<th>1M</th>
<th>4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
<td>1</td>
<td>1.5</td>
</tr>
</tbody>
</table>

**Message Size (Bytes)**

<table>
<thead>
<tr>
<th>64K</th>
<th>256K</th>
<th>1M</th>
<th>4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
<td>1</td>
<td>1.5</td>
</tr>
</tbody>
</table>
**Point-to-point: Latency & Bandwidth (Inter-Node)**

**Latency - Small Messages**
- MVAPICH2-X-Next
- HPX
- OpenMPI+UCX

**Latency - Medium Messages**
- MVAPICH2-X-Next
- HPX
- OpenMPI+UCX

**Latency - Large Messages**
- MVAPICH2-X-Next
- HPX
- OpenMPI+UCX

**Bandwidth - Small Messages**
- MVAPICH2-X-Next
- HPX
- OpenMPI+UCX

**Bandwidth - Medium Messages**
- MVAPICH2-X-Next
- HPX
- OpenMPI+UCX

**Bandwidth - Large Messages**
- MVAPICH2-X-Next
- HPX
- OpenMPI+UCX
Evaluation of Collectives Communication on EPCC ARM System

- Fulhame cluster with up to 16 dual-socket 32-core ThunderX2 nodes
- Comparison among MVAPICH2X (Next), OpenMPI+UCX, and HPCX communication libraries
- OSU Micro-benchmark Suite (OMB) 5.6.2
- Measure the MPI-level communication performance of collectives communication latency
- Evaluate single-socket (half-subscription) and dual-socket (full-subscription) scenarios on varying scale
Collectives: Single Node (64-ppn)

Allreduce – 64 ppn

Bcast – 64 ppn

Reduce – 64 ppn

Scatter – 64 ppn

Latency (us) vs Message Size (Bytes)

MVAPICH2-X-Next

OpenMPI+UCX

3x better

2x better

2.2x better
Collectives: 4 & 16 Nodes (32-ppn)

- Allreduce (4-node):
  - MVAPICH2-X-Next
  - HPCX
  - OpenMPI+UCX

- Allreduce (16-node):
  - MVAPICH2-X-Next
  - HPCX
  - OpenMPI+UCX

- Bcast (4-node):
  - MVAPICH2-X-Next
  - HPCX
  - OpenMPI+UCX

- Bcast (16-node):
  - MVAPICH2-X-Next
  - HPCX
  - OpenMPI+UCX

Latency (us) vs Message Size (Bytes)

- Allreduce:
  - 5.7x better
  - 3.7x better

- Bcast:
  - 7.6x better
  - 9.5x better
Collectives: 16 Nodes (64-ppn)

Allreduce – 64 ppn

- **MVAPICH2-X-Next**
- **OpenMPI+UCX**

<table>
<thead>
<tr>
<th>Message Size (Bytes)</th>
<th>Latency (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
</tr>
<tr>
<td>64</td>
<td>1000</td>
</tr>
<tr>
<td>256</td>
<td>10000</td>
</tr>
<tr>
<td>1K</td>
<td>100000</td>
</tr>
<tr>
<td>4K</td>
<td>1000000</td>
</tr>
<tr>
<td>16K</td>
<td>10000000</td>
</tr>
<tr>
<td>64K</td>
<td>100000000</td>
</tr>
<tr>
<td>256K</td>
<td>1000000000</td>
</tr>
<tr>
<td>1M</td>
<td>10000000000</td>
</tr>
</tbody>
</table>

- **10x better**

Bcast – 64 ppn

- **MVAPICH2-X-Next**
- **OpenMPI+UCX**

<table>
<thead>
<tr>
<th>Message Size (Bytes)</th>
<th>Latency (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
</tr>
<tr>
<td>64</td>
<td>1000</td>
</tr>
<tr>
<td>256</td>
<td>10000</td>
</tr>
<tr>
<td>1K</td>
<td>100000</td>
</tr>
<tr>
<td>4K</td>
<td>1000000</td>
</tr>
<tr>
<td>16K</td>
<td>10000000</td>
</tr>
<tr>
<td>64K</td>
<td>100000000</td>
</tr>
<tr>
<td>256K</td>
<td>1000000000</td>
</tr>
<tr>
<td>1M</td>
<td>10000000000</td>
</tr>
</tbody>
</table>

- **5x better**

Reduce – 64 ppn

- **MVAPICH2-X-Next**
- **OpenMPI+UCX**

<table>
<thead>
<tr>
<th>Message Size (Bytes)</th>
<th>Latency (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
</tr>
<tr>
<td>64</td>
<td>1000</td>
</tr>
<tr>
<td>256</td>
<td>10000</td>
</tr>
<tr>
<td>1K</td>
<td>100000</td>
</tr>
<tr>
<td>4K</td>
<td>1000000</td>
</tr>
<tr>
<td>16K</td>
<td>10000000</td>
</tr>
<tr>
<td>64K</td>
<td>100000000</td>
</tr>
<tr>
<td>256K</td>
<td>1000000000</td>
</tr>
<tr>
<td>1M</td>
<td>10000000000</td>
</tr>
</tbody>
</table>

- **7.5x better**

Scatter – 64 ppn

- **MVAPICH2-X-Next**
- **OpenMPI+UCX**

<table>
<thead>
<tr>
<th>Message Size (Bytes)</th>
<th>Latency (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
</tr>
<tr>
<td>64</td>
<td>1000</td>
</tr>
<tr>
<td>256</td>
<td>10000</td>
</tr>
<tr>
<td>1K</td>
<td>100000</td>
</tr>
<tr>
<td>4K</td>
<td>1000000</td>
</tr>
<tr>
<td>16K</td>
<td>10000000</td>
</tr>
<tr>
<td>64K</td>
<td>100000000</td>
</tr>
<tr>
<td>256K</td>
<td>1000000000</td>
</tr>
<tr>
<td>1M</td>
<td>10000000000</td>
</tr>
</tbody>
</table>

- **8x better**
MPI Job Startup Evaluation on different ARM clusters

EPCC Fulhame

- Time (millisecond)
- No. of Processes (64 ppn)
- MVAPICH2-X
- OpenMPI+UCX
- MVAPICH2-X-Next

Mayer

- Time (millisecond)
- No. of Processes (64 ppn)
- MVAPICH2-X
- OpenMPI+UCX
- MVAPICH2-X-Next

- Up to 1.6x speedup over OpenMPI w/ UCX on Catalyst Fulhame system
- Up to 6.4x speedup over OpenMPI w/ UCX on Mayer system
Evaluation of Application Kernels

- Evaluation of NAS Parallel Benchmarks, MiniAMR, and Cloverleaf kernels
- Comparison among MVAPICH2-X (Next), OpenMPI+UCX, and HPCX communication libraries
- Measure the application communication performance at varying scales with full-subscription scenarios on up to 1,024 processes
- Significant performance improvement is observed when using MVAPICH2-X
Application Evaluation – (NAS Parallel Benchmarks)

- NPB-3.4 Class-D comparing MVAPICH2-X (upcoming) and HPCX on EPCC Fulhame
- Up to 30% and 29% improvement over HPCX for CG and FT kernels.
Application Evaluation – (MiniAMR)

- MiniAMR kernel comparing MVAPICH2-X (upcoming) and HPCX on EPCC Fulhame
- Up to 23% improvement over HPCX is observed.

Input Parameters: --percent_sum 0 --num_vars 10 --stencil 21 --report_diffusion 0 --report_perf 2 --num_tsteps 100 --num_spikes 1
Conclusions

- ARM has emerged as a new platform for HPC systems
- Requires high-performance middleware designs while exploiting modern interconnects (InfiniBand)
- Provided the approaches being taken care of by the MVAPICH2 project to provide MPI support with high-performance
- Will continue to optimize and tune the MVAPICH2 stack for higher performance and scalability on ARM platforms
Commercial Support for MVAPICH2, HiBD, and HiDL Libraries

- Supported through X-ScaleSolutions (http://x-scalesolutions.com)
- Benefits:
  - Help and guidance with installation of the library
  - Platform-specific optimizations and tuning
  - Timely support for operational issues encountered with the library
  - Web portal interface to submit issues and tracking their progress
  - Advanced debugging techniques
  - Application-specific optimizations and tuning
  - Obtaining guidelines on best practices
  - Periodic information on major fixes and updates
  - Information on major releases
  - Help with upgrading to the latest release
  - Flexible Service Level Agreements
- Support provided to Lawrence Livermore National Laboratory (LLNL) for the last two years
Multiple Events at SC ‘19

• Presentations at OSU and X-Scale Booth (#2094)
  – Members of the MVAPICH, HiBD and HiDL members
  – External speakers

• Presentations at SC main program (Tutorials, Workshops, BoFs, Posters, and Doctoral Showcase)

• Presentation at many other booths (Mellanox, Intel, Microsoft, and AWS) and satellite events

• Complete details available at
  [http://mvapich.cse.ohio-state.edu/conference/752/talks/](http://mvapich.cse.ohio-state.edu/conference/752/talks/)
Funding Acknowledgments

Funding Support by

Equipment Support by
Personnel Acknowledgments

Current Students (Graduate)
- A. Awan (Ph.D.)
- M. Bayatpour (Ph.D.)
- C.-H. Chu (Ph.D.)
- J. Hashmi (Ph.D.)
- A. Jain (Ph.D.)
- K. S. Kandadi (M.S.)
- A. Augustine (M.S.)
- P. Balaji (Ph.D.)
- R. Biswas (M.S.)
- S. Bhagvat (M.S.)
- A. Bhat (M.S.)
- D. Buntinas (Ph.D.)
- L. Chai (Ph.D.)
- B. Chandrasekharan (M.S.)
- S. Chakraborthy (Ph.D.)
- N. Dandapanthula (M.S.)
- V. Dhanranj (M.S.)
- T. Gangadharappa (M.S.)
- K. Gopalakrishnan (M.S.)
- W. Huang (Ph.D.)
- W. Jiang (M.S.)
- J. Jose (Ph.D.)
- S. Kini (M.S.)
- M. Koop (Ph.D.)
- K. Kulkarni (M.S.)
- S. Krishnamoorthy (M.S.)
- K. Kandalla (Ph.D.)
- M. Li (Ph.D.)

Current Research Scientist
- Q. Zhou (Ph.D.)
- P. Lai (M.S.)
- J. Liu (Ph.D.)
- M. Luo (Ph.D.)
- A. Mamidala (Ph.D.)
- G. Marsh (M.S.)
- V. Meshram (M.S.)
- A. Moody (M.S.)
- S. Naravula (Ph.D.)
- R. Noronha (Ph.D.)
- X. Ouyang (Ph.D.)
- S. Pai (M.S.)
- S. Pottluri (Ph.D.)

Past Students
- A. Augustine (M.S.)
- P. Balaji (Ph.D.)
- R. Biswas (M.S.)
- S. Bhagvat (M.S.)
- A. Bhat (M.S.)
- D. Buntinas (Ph.D.)
- L. Chai (Ph.D.)
- B. Chandrasekharan (M.S.)
- S. Chakraborthy (Ph.D.)
- N. Dandapanthula (M.S.)
- V. Dhanranj (M.S.)
- T. Gangadharappa (M.S.)
- K. Gopalakrishnan (M.S.)
- W. Huang (Ph.D.)
- W. Jiang (M.S.)
- J. Jose (Ph.D.)
- S. Kini (M.S.)
- M. Koop (Ph.D.)
- K. Kulkarni (M.S.)
- S. Krishnamoorthy (M.S.)
- K. Kandalla (Ph.D.)
- M. Li (Ph.D.)

Past Post-Docs
- D. Banerjee
- X. Besseron
- H.-W. Jin
- J. Lin
- M. Luo
- E. Mancini

Past Programmers
- D. Bureddy
- J. Perkins

Past Research Scientist
- K. Hamidouche
- S. Sur
- X. Lu

Past Research Specialist
- M. Arnold

Current Students (Undergraduate)
- V. Gangal (B.S.)
- N. Sarkauskas (B.S.)

Current Research Specialist
- H. Subramoni

Current Post-doc
- M. S. Ghazimeersaeed
- A. Ruhela
- K. Manian

Current Research Specialist
- J. Smith

Past Post-Docs
- J. Lin
- M. Luo
- E. Mancini
- S. Marcarelli
- J. Vienne
- H. Wang
Thank You!

panda@cse.ohio-state.edu

Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu/

The High-Performance MPI/PGAS Project
http://mvapich.cse.ohio-state.edu/

The High-Performance Big Data Project
http://hibd.cse.ohio-state.edu/

The High-Performance Deep Learning Project
http://hidl.cse.ohio-state.edu/