MVAPICH2-DPU: Efficient MPI Offloading on BlueField DPUs for Accelerating Scientific Applications

Dhabaleswar K. (DK) Panda, Nick Sarkauskas, Donglai Dai, Hari Subramoni

March 10, 2022

E-mail: contactus@x-scalesolutions.com

X-ScaleSolutions

Requirements for Next-Generation Communication Libraries

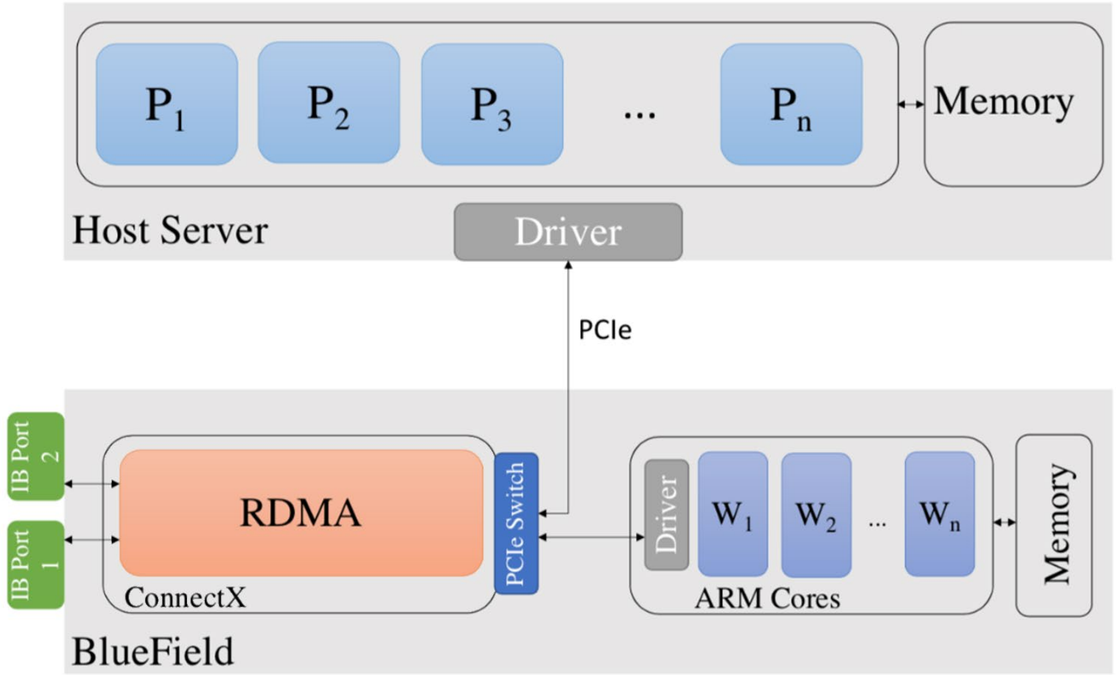
- Message Passing Interface (MPI) libraries are widely used for HPC and AI applications
- Requirements for a high-performance and scalable MPI library:
  - Low latency communication
  - High bandwidth communication
  - Minimum contention for host CPU resources to progress non-blocking collectives
  - High overlap of computation with communication
- CPU based non-blocking communication progress can lead to sub-par performance as the main application has less CPU resources for useful application-level computation

Can MPI Functions be Offloaded?

- The area of network offloading of MPI primitives is still nascent
- State-of-the-art BlueField DPUs bring more compute power into the network
- Exploit additional compute capabilities of modern BlueField DPUs into existing MPI middleware to extract
  - Peak pure communication performance
  - Overlap of communication and computation

Overview of BlueField-3 DPU

- ConnectX-6 network adapter with 200Gbps InfiniBand
- System-on-chip containing eight 64-bit ARMv8 A72 cores with 2.7 GHz each
- 16 GB of memory for the ARM cores



MVAPICH2-DPU MPI library is designed to take advantage of DPUs and accelerate scientific applications

MVAPICH2-DPU Library 2022.02 Release

- Implemented by X-ScaleSolutions
- Based on MVAPICH2 2.3.6, compliant to MPI 3.1 standard
- Supports all features available with the MVAPICH2 2.3.6 release (http://mvapich.cse.ohio-state.edu)
- Novel framework to offload non-blocking collectives to DPU
- Offloads non-blocking collectives (MPI_Ialltoall, MPI_Iallgather, MPI_Ibcast, etc) to DPU
- Up to 100% overlap of computation with non-blocking collective
- Accelerates scientific applications using non-blocking collectives

Running Applications using MVAPICH2-DPU

There are five steps for running an application using MVAPICH2-DPU library on the HPCAC Thor cluster.

Step 1. On the Thor cluster, there is a ConnectX-6 Host Channel Adapter (HCA) as well as the BlueField HCA. Select the BlueField HCA if there are multiple HCAs installed in the system by adding the following to ~/.bashrc:

```
STR=`hostname`
SUB="bf"
if [[ "$STR" == *"$SUB"* ]]; then
  export MV2_IBA_HCA=mlx5_0
else
  export MV2_IBA_HCA=mlx5_2
fi
```

Running Applications using MVAPICH2-DPU (cont.)

Step 2. Allocate resources with Slurm, making sure to allocate the corresponding BlueField to each host:

```
salloc -N 8 -p thor -w thor[001-004],thor-bf[01-04] -t 2:00:00
```

Step 3. Create a hostfile using allocated nodes with the format hostname:processes per node. Example hostfile:

```
thor001:32
thor002:32
thor003:32
thor004:32
```

Running Applications using MVAPICH2-DPU (cont.)

Step 4. Create a dpufile by adding allocated BlueField hostnames one per line without any duplicates. Note: the ':' operator is unsupported in a dpufile. MVAPICH2-DPU will determine the optimal number of processes per BlueField at runtime.

Example dpufile:

```
thor-bf01
thor-bf02
thor-bf03
thor-bf04
```

A dpufile can be generated using the following command:

```
scontrol show hostnames | grep bf > ./dpufile
```

Running Applications using MVAPICH2-DPU (cont.)

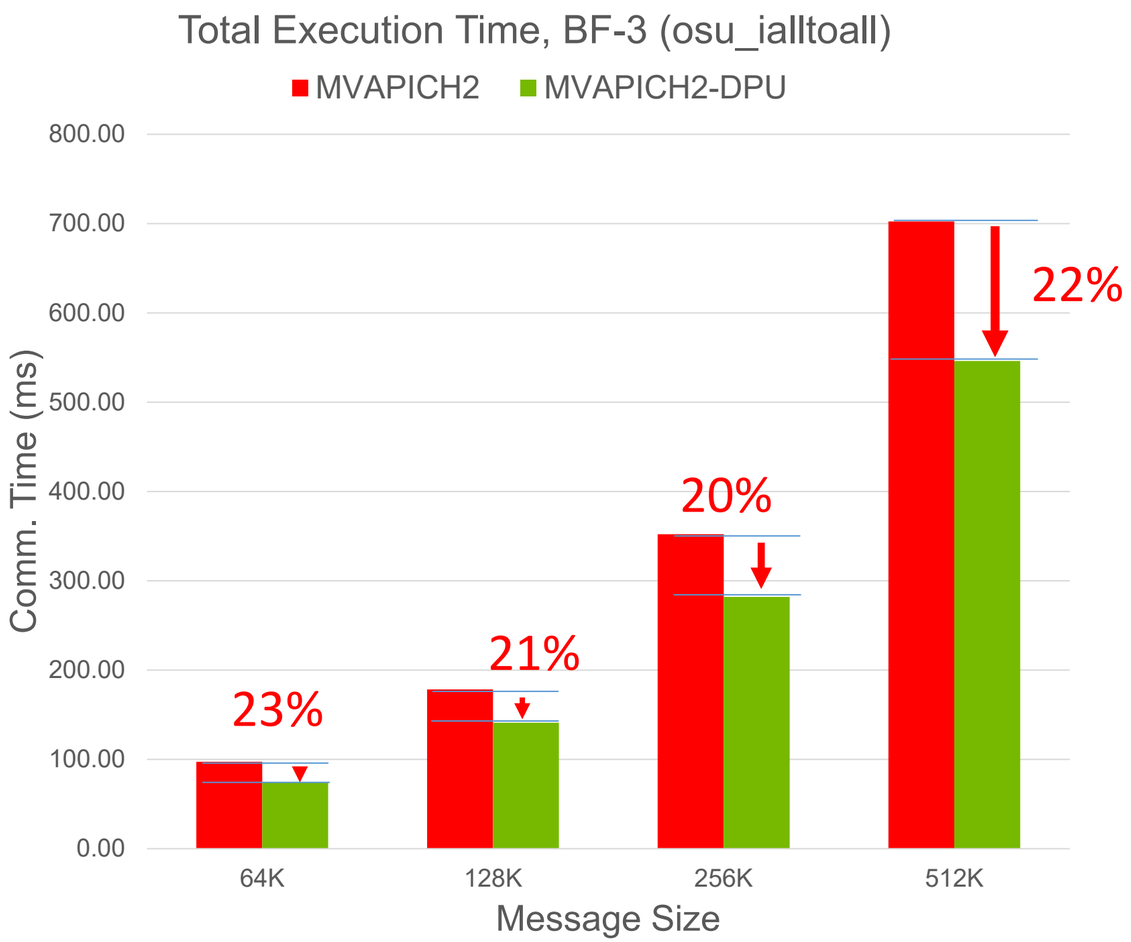
Step 5a. Run `mpirun_rsh` with DPUs enabled using the hostfile and dpufile:

```
mpirun_rsh -np <n host processes, not including DPUs>      \
    -hostfile ./hostfile                                   \
    -dpufile  ./dpufile                                    \
    MV2_USE_DPU=1                                          \
    <path to application executable>
```

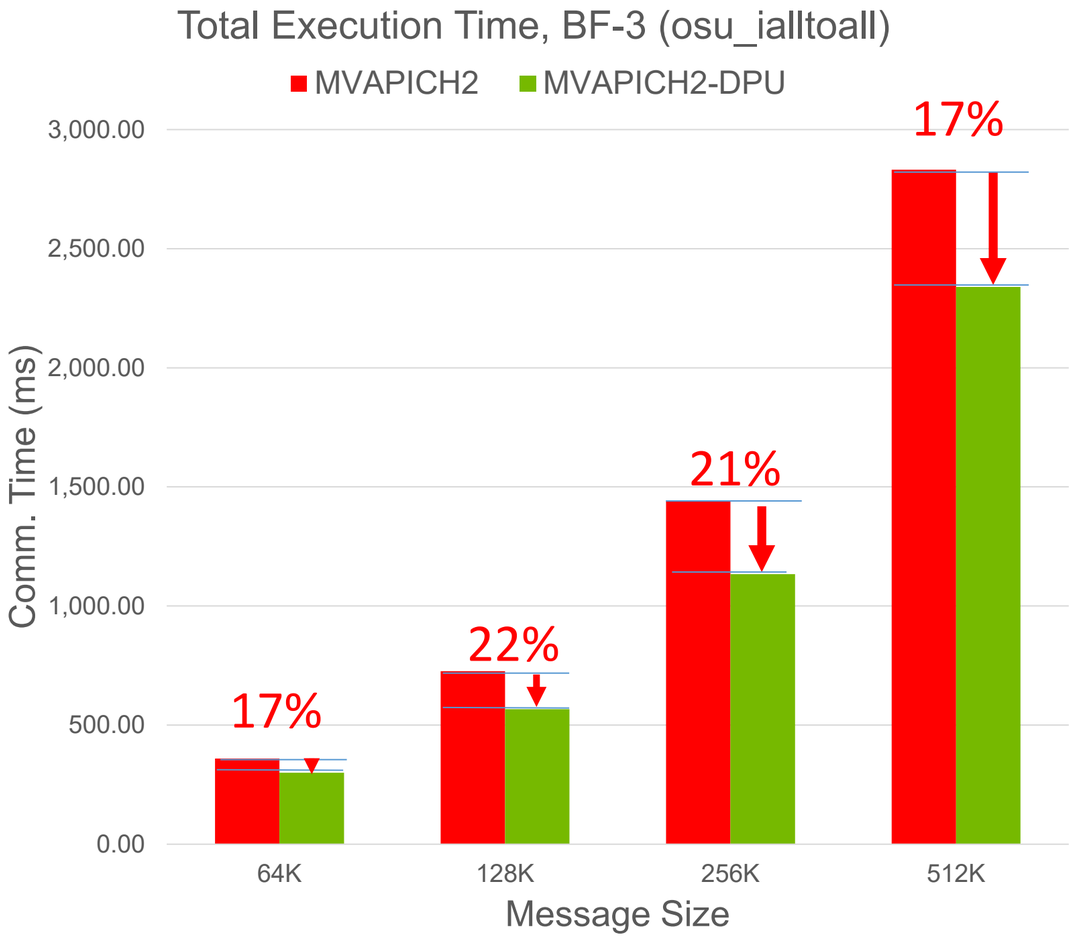
Step 5b. Run `mpirun_rsh` with DPUs disabled using just the hostfile:

```
mpirun_rsh -np <n host processes, not including DPUs>      \
    -hostfile ./hostfile                                   \
    MV2_USE_DPU=0                                          \
    <path to application executable>
```

Total Execution Time with osu_Ialltoall (32 nodes)

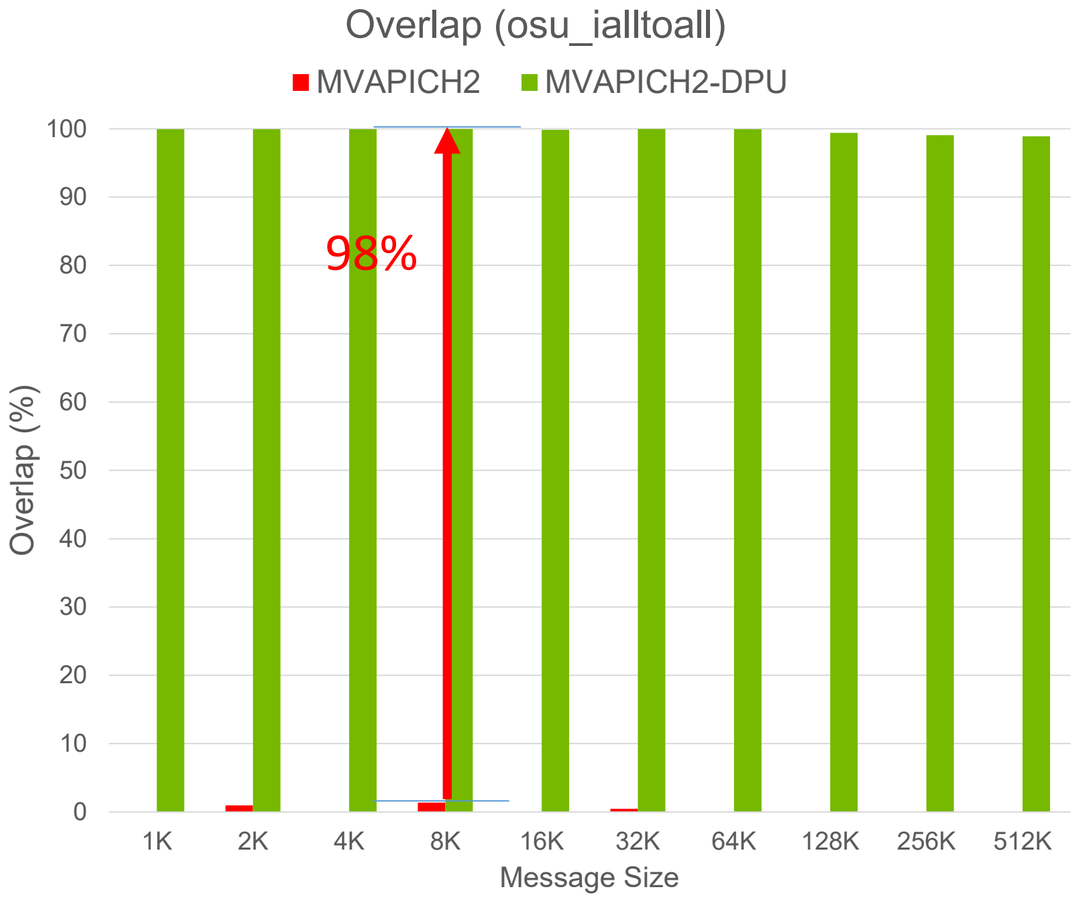


32 Nodes, 16 PPN

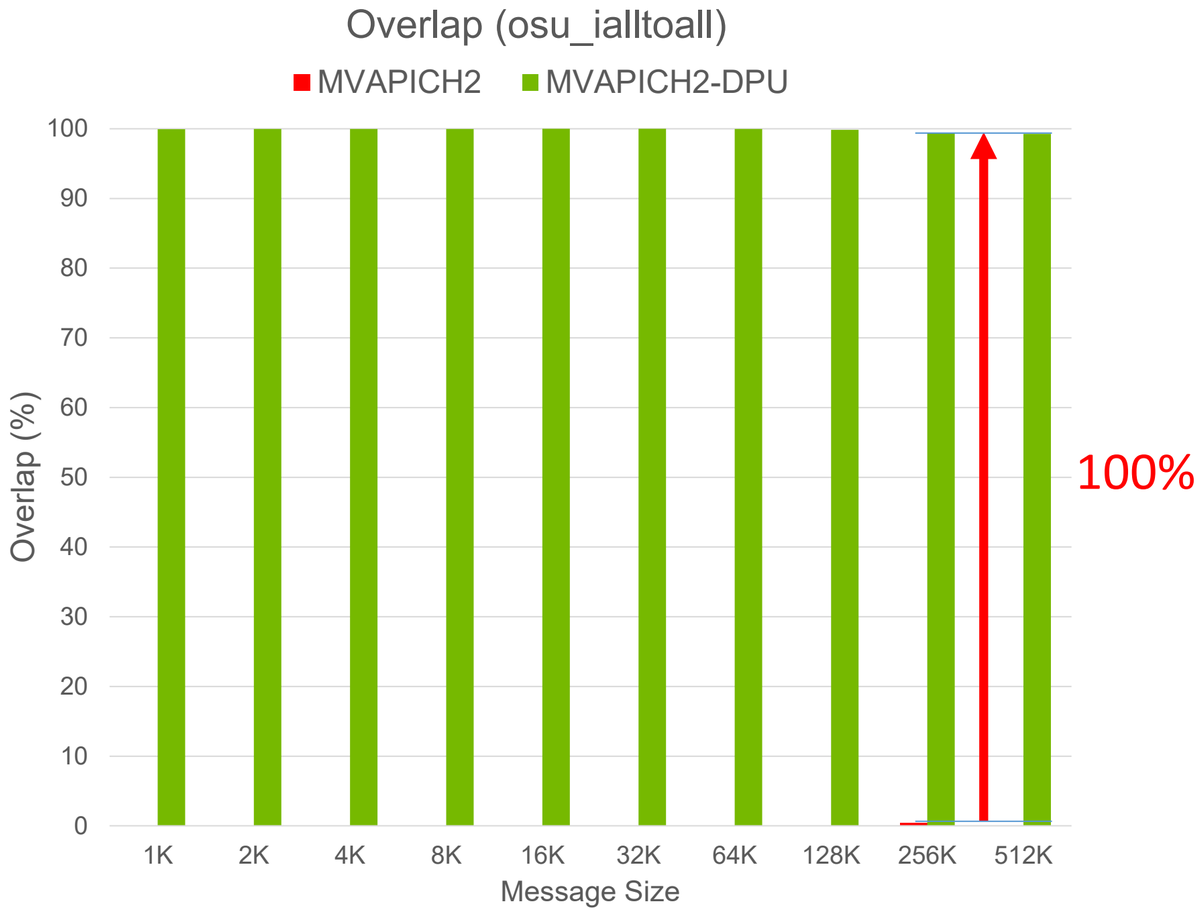


32 Nodes, 32 PPN

Overlap Between Computation & Communication with osu_Ialltoall (32 nodes)



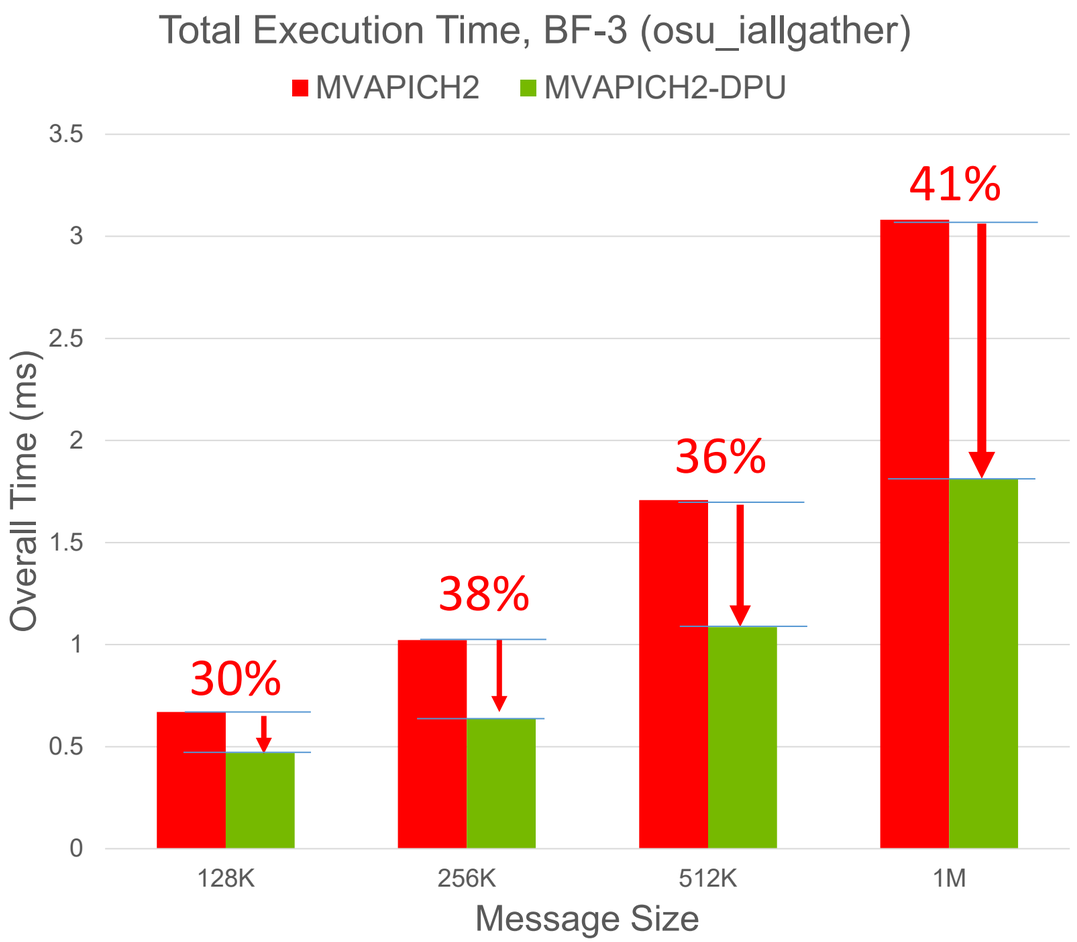
32 Nodes, 16 PPN



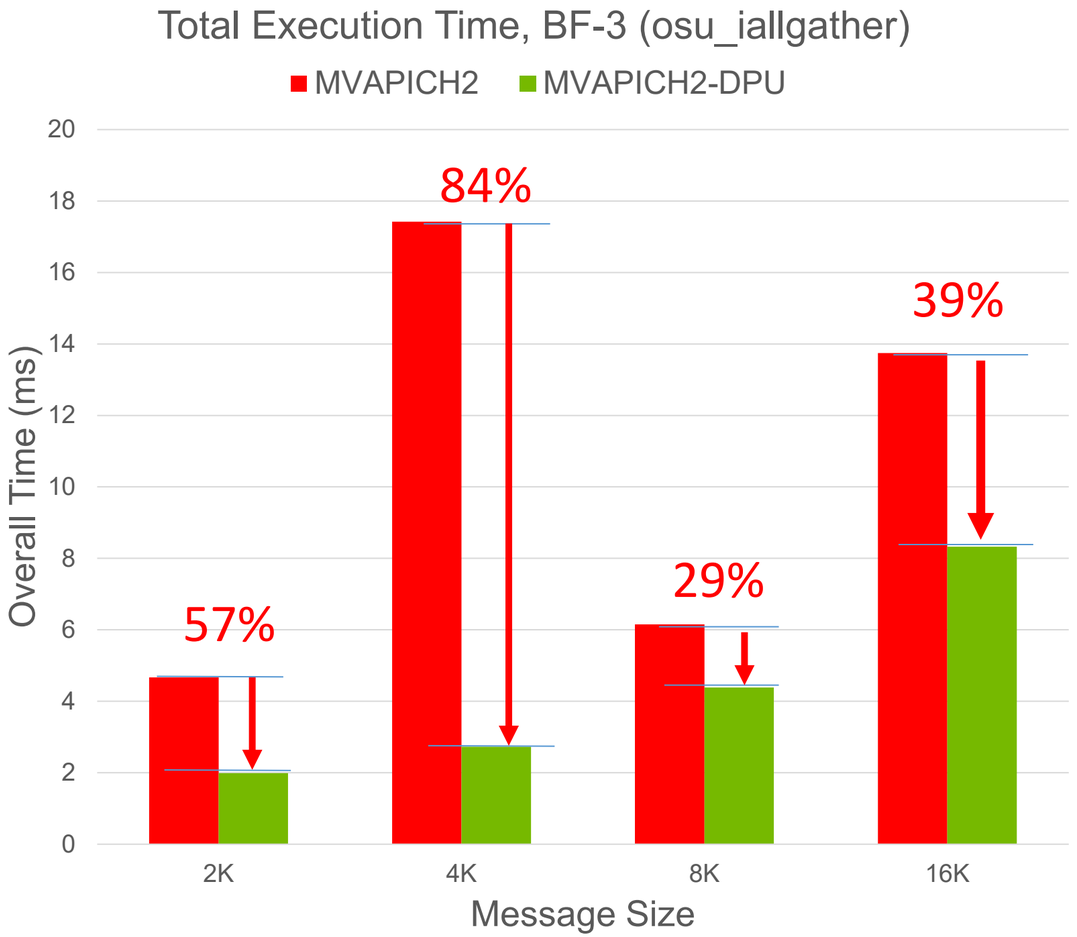
32 Nodes, 32 PPN

Delivers peak overlap

Total Execution Time with ous_Iallgather (16 nodes)

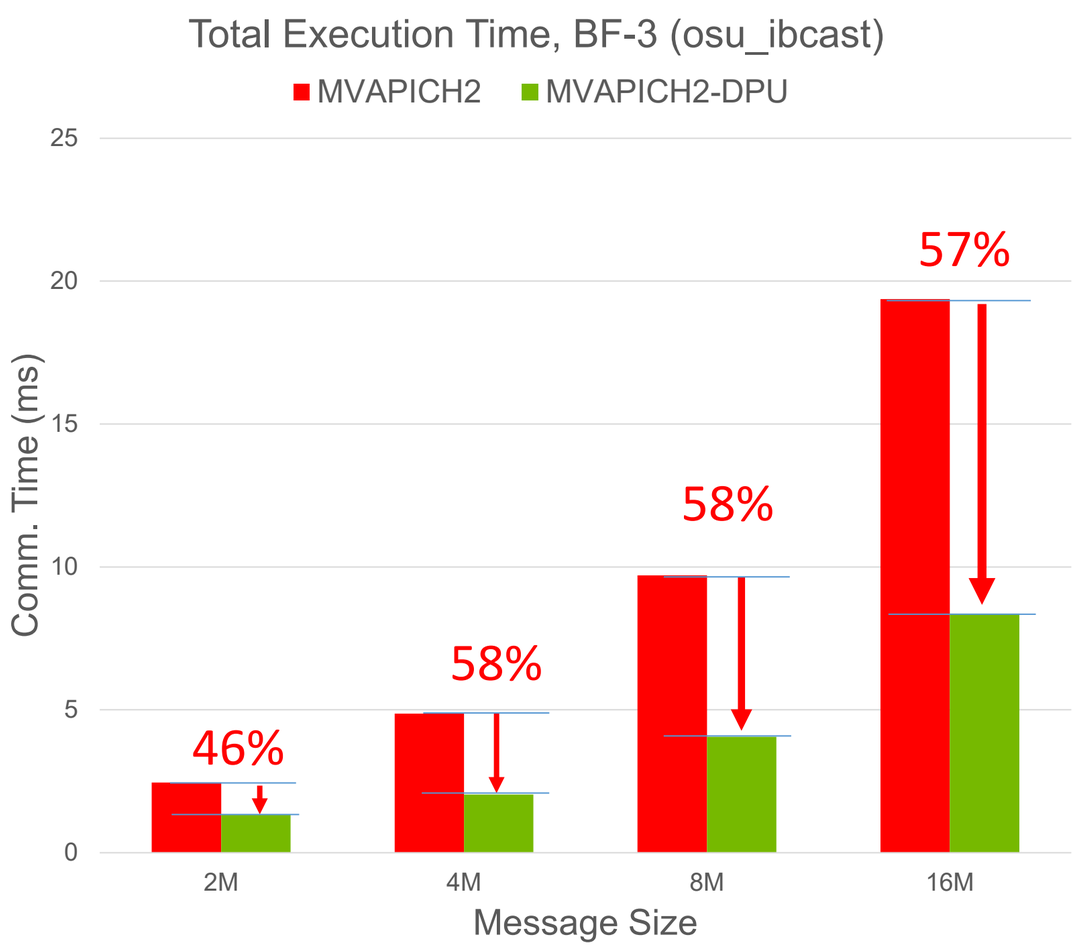


16 Nodes, 1 PPN

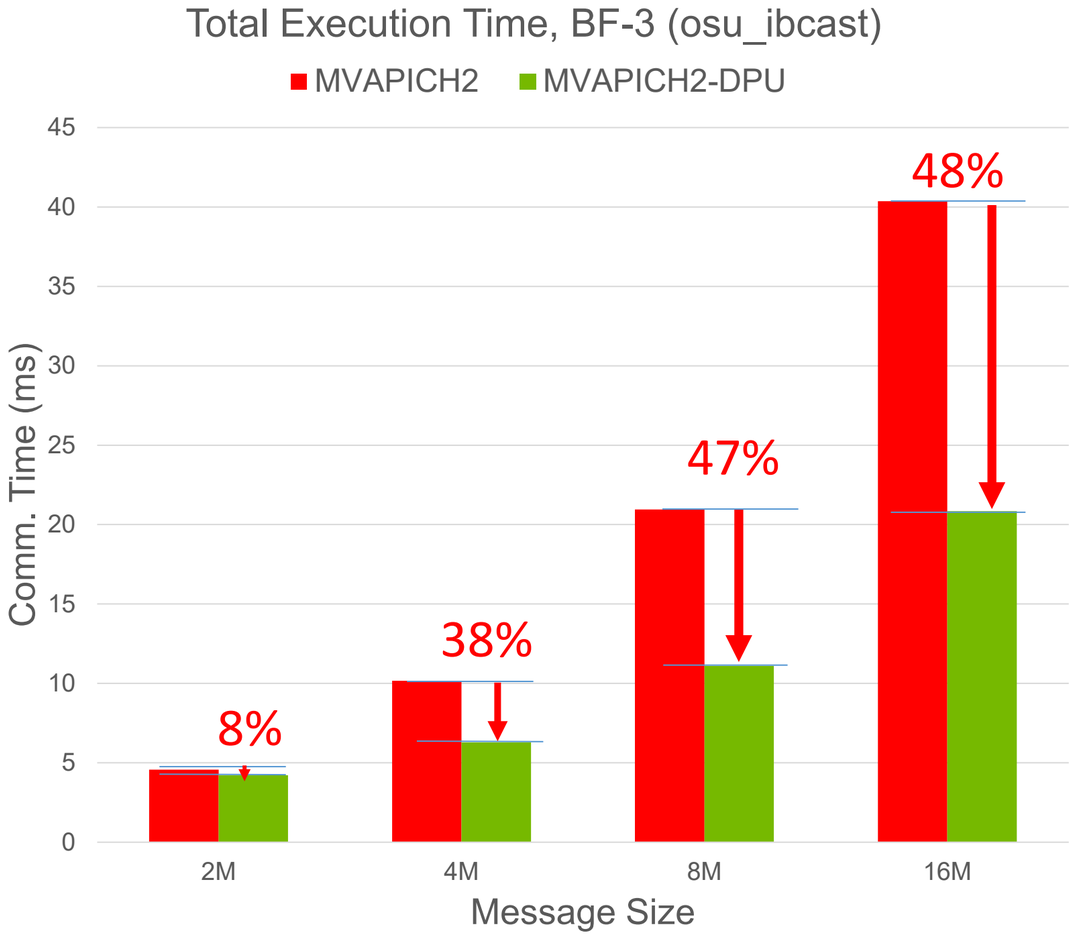


16 Nodes, 32 PPN

Total Execution Time with ous_Ibcast (32 nodes)

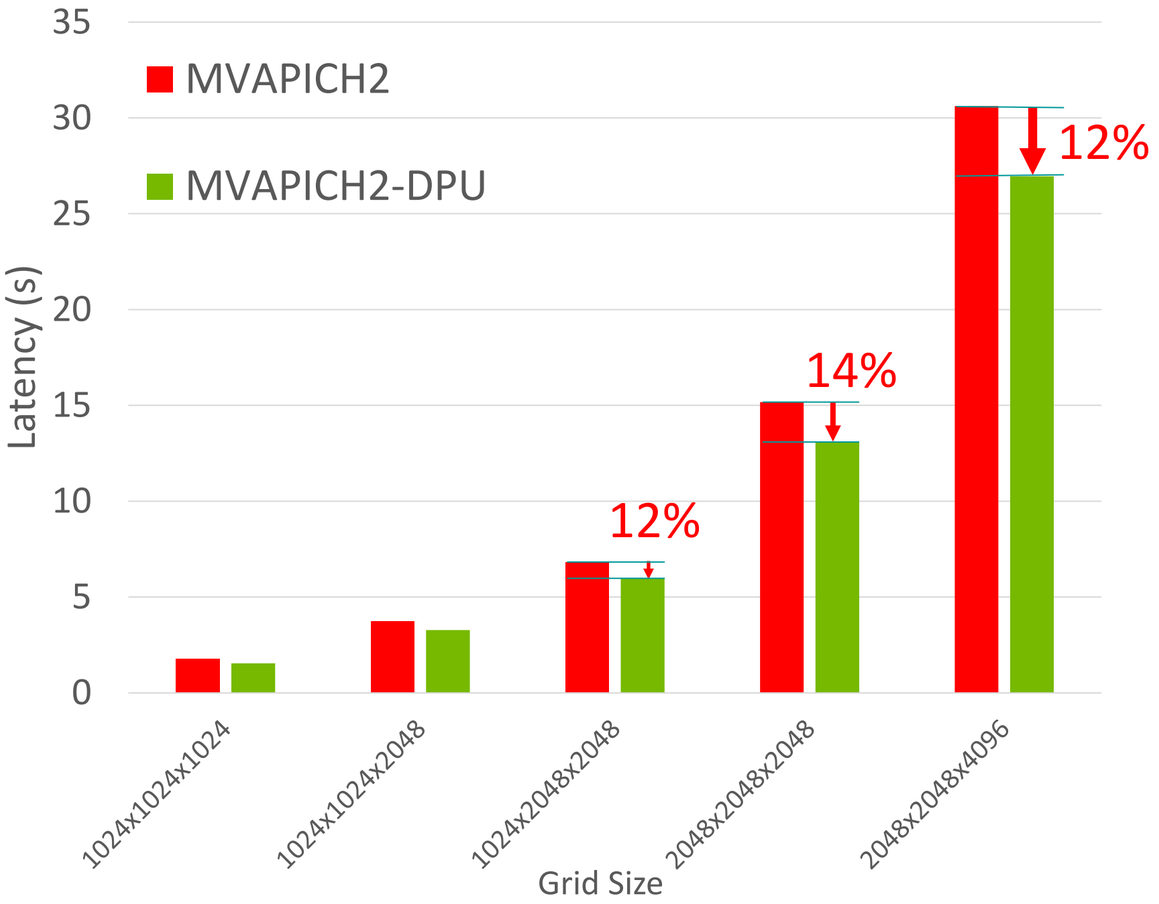


32 Nodes, 1 PPN

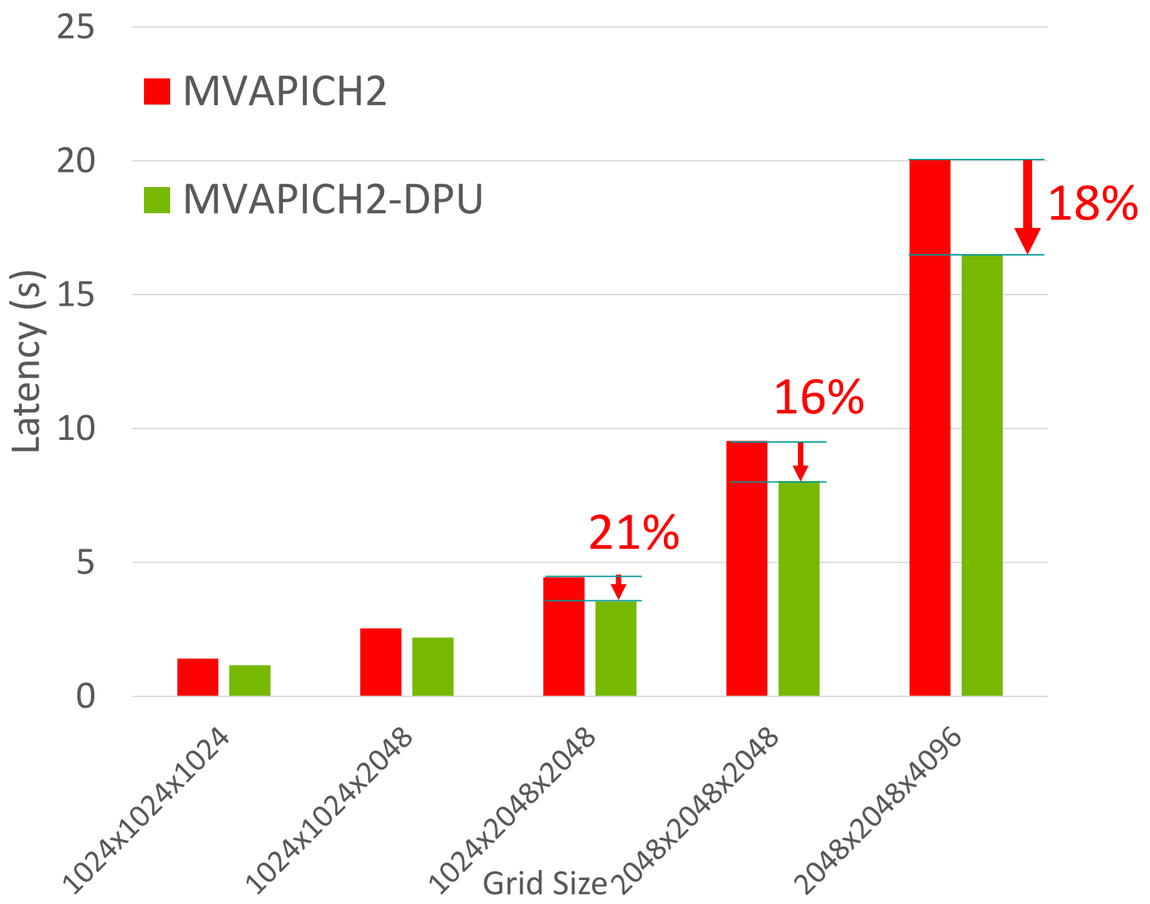


32 Nodes, 16 PPN

P3DFFT Application Execution Time (32 nodes)



32 Nodes, 16 PPN



32 Nodes, 32 PPN

Benefits in application-level execution time

Conclusion

- Efficient MVAPICH2-DPU MPI library utilizes the BlueField DPU to progress MPI non-blocking collective operations
- Provides up to 100% overlap of communication and computation for non-blocking Alltoall, Allgather, Bcast, etc
- Reduces the total execution time of P3DFFT application up to 21% on 1,024 processes
- Work in progress for MVAPICH2-DPU library to efficiently offload more types of non-blocking collective operations to DPUs

Thank You!

Dhabaleswar K. (DK) Panda, Nick Sarkauskas, Donglai Dai, Hari Subramoni

contactus@x-scalesolutions.com

X-ScaleSolutions

http://x-scalesolutions.com/