GPU-Aware MPI with ROCm™

Presenter: Bob Robey
AMD @ Tsukuba University
Oct 21-23, 2025

Agenda

- Introduction
- Running GPU-Aware MPI examples
  - Point-to-Point Communication Example
  - Collective Communication Example
- Measuring GPU-Aware Communication BW and Latency
  - GPU Placement Consideration on AAC cloud
  - GPU Placement Consideration on LUMI
  - Communication Options
  - Measuring intra-node/inter-node communication bandwidth
  - Measuring collective communication performance
  - Communication performance on MI300A
- ROCm Collective Communication Library (RCCL)
- Summary
- Exercises

What is MPI?

- MPI (Message-Passing Interface) is the de facto standard for communication in High Performance Computing
- Processes in an MPI program have private address space
  - MPI program can be executed on systems with distributed memory space
- MPI standard defines message passing APIs for point-to-point and collective operations

What is GPU-Aware MPI?

- Traditionally, only pointers of the host buffers could be passed to MPI calls
- GPU-Aware MPI provides this opportunity to pass GPU buffers to MPI calls
- Without GPU-Aware MPI, GPU buffers have to be staged through host memory with hipMemcpy
- Many MPI implementations including CRAY-MPICH, MVAPICH and OpenMPI support GPU-Aware Communication

What is GPUDirect RDMA?

- GPUDirect RDMA is a technology that provides the opportunity for network adapters to directly access GPU device memory and completely bypass the host

- Note that GPU-Aware MPI refers to support for passing GPU buffers to MPI calls in MPI implementations while GPUDirect RDMA is a technology that enables direct access to GPU memory

- A GPU-Aware MPI may or may not use GPUDirect RDMA for communications between GPUs

GPU-Aware Point-to-Point Communication Example

```
//allocate memory
h_buf=(int*) malloc(sizeof(int)*bufsize);
hipMalloc(&d_buf,bufsize*sizeof(int));

//initialize
if (rank == 0)
{
    for (i=0; i<bufsize; i++)
        h_buf[i] = i;
    hipMemcpy(d_buf, h_buf, (bufsize) * sizeof(int), hipMemcpyHostToDevice);
}

if (rank == 1)
{
    for (i=0; i<bufsize; i++)
        h_buf[i] = -1;
    hipMemcpy(d_buf, h_buf, (bufsize) * sizeof(int), hipMemcpyHostToDevice);
}
```

Allocate memory on host

Allocate memory on device

Initialize device buffer

```
// communication
if (rank == 0) {
   MPI_Send(d_buf, bufsize, MPI_INT, 1, 123, MPI_COMM_WORLD); }

if (rank == 1) {
   MPI_Recv(d_buf, bufsize, MPI_INT, 0, 123, MPI_COMM_WORLD, &status); }
```

GPU-Aware P2P communication

```
// validate results
if (rank == 1)
{
    hipMemcpy(h_buf, d_buf, (bufsize) * sizeof(int), hipMemcpyDeviceToHost);
    for (i=0; i<bufsize; i++)
    {
        if (h_buf[i] != i)
            printf("Error: buffer[%d] = %d but is expected to be %d\n", i, h_buf[i], i);
    }
    fflush(stdout);
}

free(h_buf);
hipFree(d_buf);
MPI_Finalize();
```

Validate results

Free memory

What if we don’t have GPU-Aware MPI?

- Stage GPU buffers through host memory with hipMemcpy

```
if (rank == 0) {
   //copy send buffer from device to host
   hipMemcpy(h_buf, d_buf, (bufsize) * sizeof(int), hipMemcpyDeviceToHost);

   MPI_Send(h_buf, bufsize, MPI_INT, 1, 123, MPI_COMM_WORLD);
}


if (rank == 1) {
   MPI_Recv(h_buf, bufsize, MPI_INT, 0, 123, MPI_COMM_WORLD, &status);

   //copy receive buffer from host to device
   hipMemcpy(d_buf, h_buf, (bufsize) * sizeof(int), hipMemcpyHostToDevice);
}
```

GPU-Aware Collective Communication Example

```
//set device
hipSetDevice(rank%8);

//check device ID
hipGetDevice(&deviceID);
printf("rank%d running on device %d\n", rank, deviceID);

//allocate memory on host
h_buffer = (int *)malloc( count * sizeof(int) );

//allocate memory on device
hipMalloc(&d_sendbuf,count*sizeof(int));
hipMalloc(&d_recvbuf,count*sizeof(int));

//initialize send and receive buffers
for (i=0; i<count; i++) h_buffer[i] = i;
hipMemcpy(d_sendbuf, h_buffer, (count) * sizeof(int), hipMemcpyHostToDevice);

hipMemset(d_recvbuf,0,count*sizeof(int));

//GPU-Aware Reduce
MPI_Reduce( d_sendbuf, d_recvbuf, count, MPI_INT, MPI_SUM, root, comm );

//validate results
if (rank == root) {
   for (i=0; i<count; i++) h_buffer[i] = 0;
   hipMemcpy(h_buffer, d_recvbuf, (count) * sizeof(int), hipMemcpyDeviceToHost);
   for (i=0; i<count; i++) {
      if (h_buffer[i] != i * size) {
          errs++;
       }
    }
   if(errs!=0) printf("errors=%d\n", errs);
}

hipFree(d_sendbuf);
hipFree(d_recvbuf);
free( h_buffer );
```

Set device

Allocate send/recv buffers on device

Initialize send/recv buffers

GPU-Aware Collective Communication

Validate results

Free memory

GPU connectivity on AAC Cloud (MI210)

rocm-smi --showtopo

```
GPU[0]          : (Topology) Numa Node: 0
GPU[0]          : (Topology) Numa Affinity: 0
GPU[1]          : (Topology) Numa Node: 0
GPU[1]          : (Topology) Numa Affinity: 0
GPU[2]          : (Topology) Numa Node: 0
GPU[2]          : (Topology) Numa Affinity: 0
GPU[3]          : (Topology) Numa Node: 0
GPU[3]          : (Topology) Numa Affinity: 0
GPU[4]          : (Topology) Numa Node: 1
GPU[4]          : (Topology) Numa Affinity: 1
GPU[5]          : (Topology) Numa Node: 1
GPU[5]          : (Topology) Numa Affinity: 1
GPU[6]          : (Topology) Numa Node: 1
GPU[6]          : (Topology) Numa Affinity: 1
GPU[7]          : (Topology) Numa Node: 1
GPU[7]          : (Topology) Numa Affinity: 1
```

```
============================ Link Type between two GPUs ============================
       GPU0       GPU1       GPU2       GPU3       GPU4       GPU5       GPU6       GPU7
GPU0   0          XGMI       XGMI       XGMI       PCIE       PCIE       PCIE       PCIE
GPU1   XGMI       0          XGMI       XGMI       PCIE       PCIE       PCIE       PCIE
GPU2   XGMI       XGMI       0          XGMI       PCIE       PCIE       PCIE       PCIE
GPU3   XGMI       XGMI       XGMI       0          PCIE       PCIE       PCIE       PCIE
GPU4   PCIE       PCIE       PCIE       PCIE       0          XGMI       XGMI       XGMI
GPU5   PCIE       PCIE       PCIE       PCIE       XGMI       0          XGMI       XGMI
GPU6   PCIE       PCIE       PCIE       PCIE       XGMI       XGMI       0          XGMI
GPU7   PCIE       PCIE       PCIE       PCIE       XGMI       XGMI       XGMI       0
```

```
============================== Hops between two GPUs ==============================
       GPU0       GPU1       GPU2       GPU3       GPU4       GPU5       GPU6       GPU7
GPU0   0          1          1          1          3          3          3          3
GPU1   1          0          1          1          3          3          3          3
GPU2   1          1          0          1          3          3          3          3
GPU3   1          1          1          0          3          3          3          3
GPU4   3          3          3          3          0          1          1          1
GPU5   3          3          3          3          1          0          1          1
GPU6   3          3          3          3          1          1          0          1
GPU7   3          3          3          3          1          1          1          0
```


These tests have been executed as a demo (not performance claim) on AAC CLOUD with MI210 GPUs, rocm 5.6.0 and openmpi/4.1.5 on Sept/26/2023.

Demo: Intra-node GPU-to-GPU Communication Bandwidth on AAC Cloud (MI210)

Use UCX for communication

```
$export HIP_VISIBLE_DEVICES=0,1
$mpirun -n 2 -mca pml ucx ../build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw  -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.2
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
# Datatype: MPI_CHAR.
16777216          41454.31

$export HIP_VISIBLE_DEVICES=0,4
$mpirun -n 2 -mca pml ucx ../build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw  -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.2
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
# Datatype: MPI_CHAR.
16777216          25172.16
```

GPU 0 & 1 → 41 GB/s

Device to device communication

GPU 0 & 4 → 25 GB/s

OSU Microbenchmark (OMB): Feature a series of MPI benchmarks that measure the performances of various MPI operations including point-to-point, collective, host-based and device-based communications

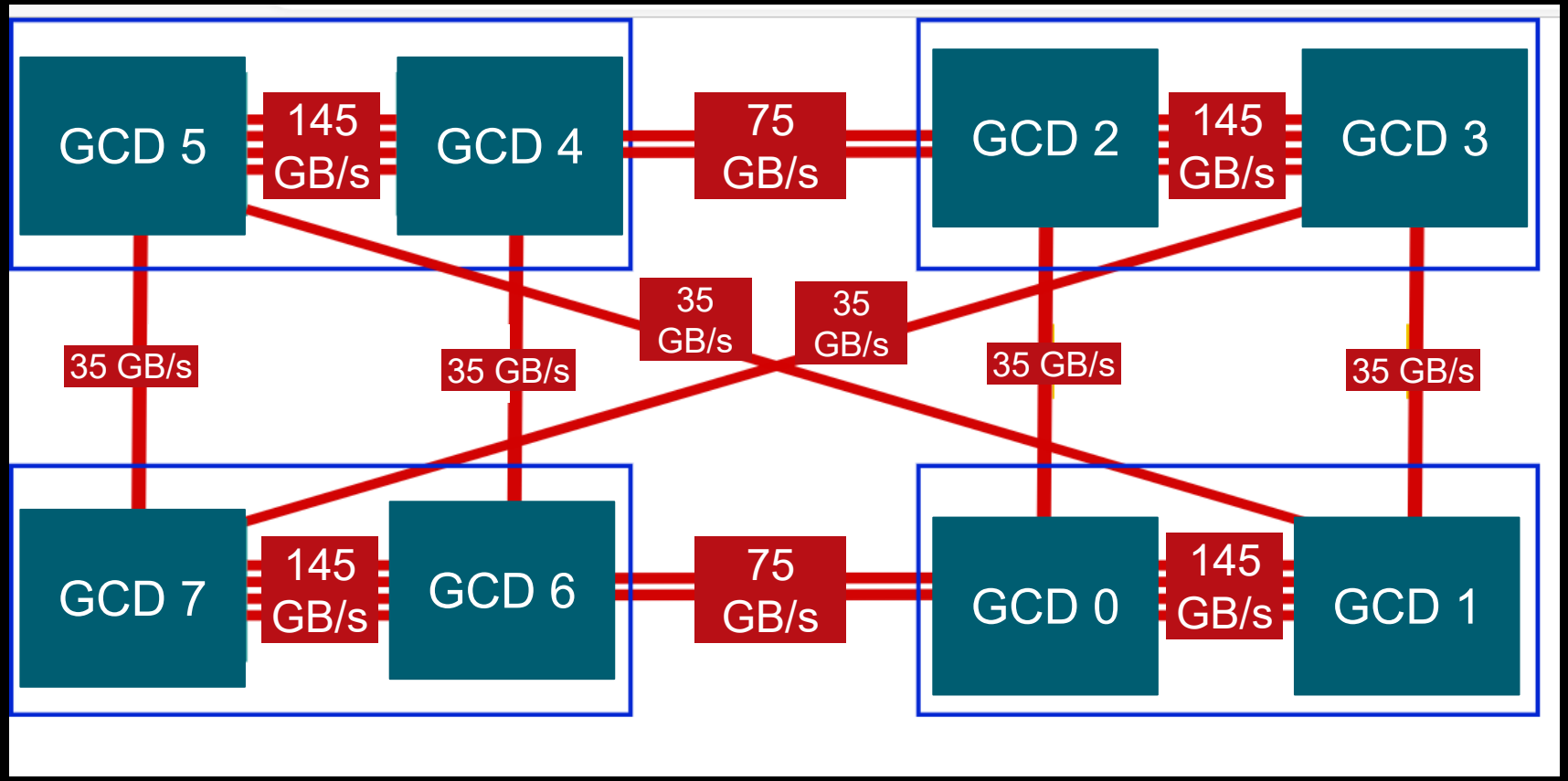
These tests have been executed as a demo (not performance claim) on AAC CLOUD with MI210 GPUs, rocm 6.1.0 and openmpi/5.0.3 on Sept/26/2023.

GPU-to-GPU Communication Options

- There are two options for GPU-to-GPU communication
  - SDMA engine
    - Provides the opportunity to overlap communication with computation
    - Each SDMA engine can provide maximum communication BW of 49 GB/s between GCDs
  - blit kernels
    - Launch kernel to handle communication
    - Pros: higher communication bandwidth
    - Cons: cannot overlap communication with computation

GPU connectivity

- Achievable GPU-to-GPU Communication Bandwidth using blit kernel
- Different number of Infinity Fabric™ links between GCDs
  - GCDs of the same GPU are connected with 4 Infinity Fabric™ links
- Different number of hops between GCDs



The numbers in this figure are calculated with (achievable bw ~ Theoretical bw * 0.7). Theoretical bw is obtained from https://docs.olcf.ornl.gov/systems/frontier_user_guide.html#frontier-compute-nodes

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,1
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          142341.39
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,2
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216           38963.39
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,3
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216           36903.69
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,4
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216           36908.40
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,5
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216           34986.18
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,6
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216           76276.50
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,7
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216           68778.59
```

GCD 0 & 1 → 142 GB/s

GCD 0 & 2 → 38 GB/s

GCD 0 & 3 → 36 GB/s

GCD 0 & 4 → 36 GB/s

GCD 0 & 5 → 34 GB/s

GCD 0 & 6 → 76 GB/s

GCD 0 & 7 → 68 GB/s

Device to device communication

Demo: Intra-node GPU-to-GPU Communication Bandwidth on LUMI Using blit Kernels

```
$module load rocm

$module load cray-mpich/8.1.18

$export MPICH_GPU_SUPPORT_ENABLED=1

$export HSA_ENABLE_SDMA=0
```

Enable blit kernel

These tests have been executed as a demo (not performance claim) on LUMI pilot partition with MI250x GPUs, rocm 5.3.0 and cray-mpich/8.1.18 on May/03/2023.

Demo: Intra-node GPU-to-GPU Communication Bandwidth on LUMI using SDMA

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,1
mghazimi@uan02:~/OMB/osu_benchmark> srun --jobid=2057636 -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          49955.50
```

GCD 0 & 1 → 49 GB/s

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,2
mghazimi@uan02:~/OMB/osu_benchmark> srun --jobid=2057636 -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          36377.30
```

GCD 0 & 2 → 36 GB/s

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,3
mghazimi@uan02:~/OMB/osu_benchmark> srun --jobid=2057636 -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          36940.74
```

GCD 0 & 3 → 36 GB/s

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,4
mghazimi@uan02:~/OMB/osu_benchmark> srun --jobid=2057636 -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          36955.43
```

GCD 0 & 4 → 36 GB/s

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,5
mghazimi@uan02:~/OMB/osu_benchmark> srun --jobid=2057636 -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          36359.46
```

GCD 0 & 5 → 36 GB/s

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,6
mghazimi@uan02:~/OMB/osu_benchmark> srun --jobid=2057636 -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          49971.79
```

GCD 0 & 6 → 49 GB/s

```
mghazimi@uan02:~/OMB/osu_benchmark> export HIP_VISIBLE_DEVICES=0,7
mghazimi@uan02:~/OMB/osu_benchmark> srun --jobid=2057636 -N 1 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m $((16*1024*1024)):$((16*1024*1024)) D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
16777216          49945.63
```

GCD 0 & 7 → 49 GB/s

```
$module load rocm

$module load cray-mpich/8.1.18

$export
MPICH_GPU_SUPPORT_ENABLED=1

$export HSA_ENABLE_SDMA=1
```

Enable SDMA

```
** Recent update **

$export HSA_ENABLE_SDMA_GANG=1
```

Utilize all links to make SDMA transfers….but *will* impact bidirectional traffic

These tests have been executed as a demo (not performance claim) on LUMI pilot partition with MI250x GPUs, rocm 5.3.0 and cray-mpich/8.1.18 on May/03/2023

Summary of the Achievable Bandwidth with blit kernel vs SDMA

- Achieve up to 49 GB/s using SDMA
- Achieve up to 142 GB/s using blit kernel
- The communication bandwidth between GCDs depends on
  - SDMA vs blit kernel
  - Number of Infinity Fabric™ links between GCDs
  - Number of hops between GCDs

Achieved Bandwidth on LUMI with blit kernel (GB/s)

	GCD1	GCD2	GCD3	GCD4	GCD5	GCD6	GCD7
GCD0	142	38	36	36	34	76	68

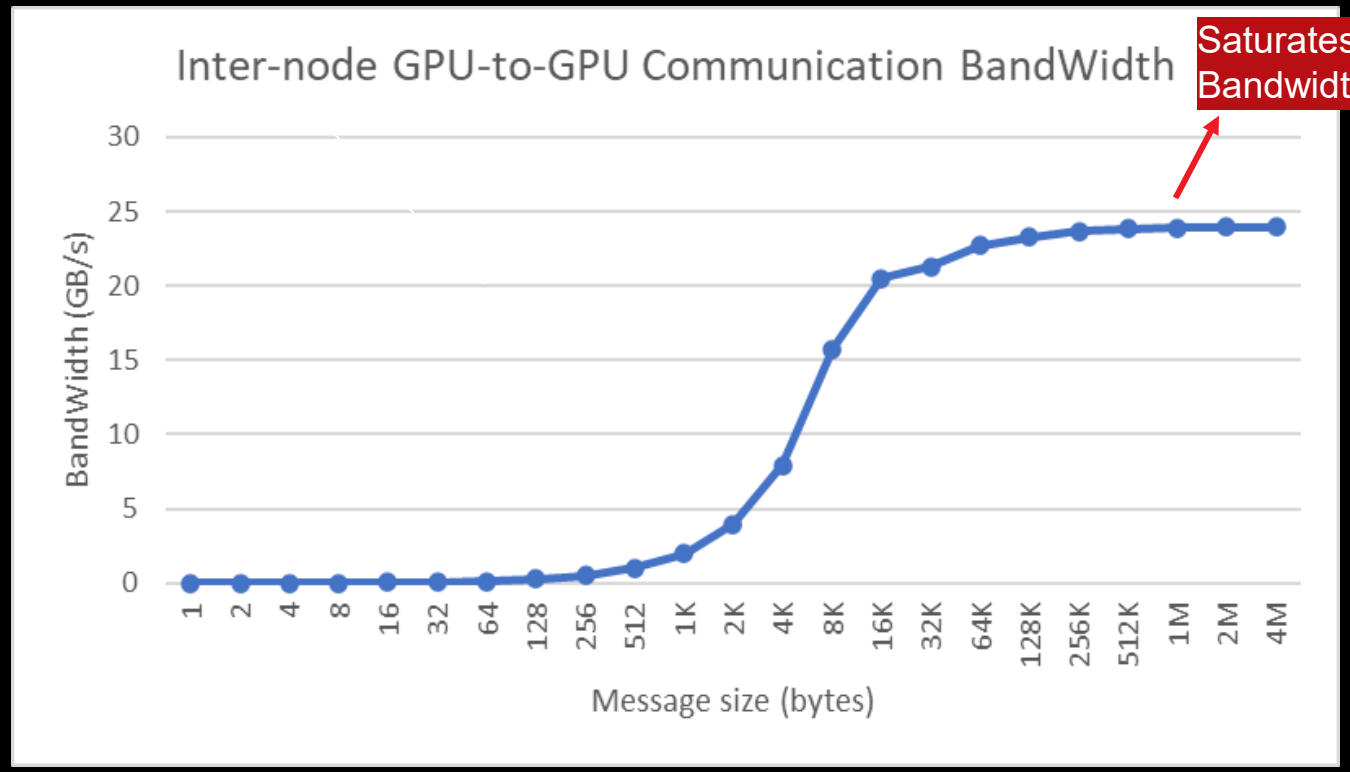
Achieved Bandwidth on LUMI with SDMA (GB/s)

	GCD1	GCD2	GCD3	GCD4	GCD5	GCD6	GCD7
GCD0	49	36	36	36	34	49	49

These numbers have been taken as a demo (not performance claim) on LUMI pilot partition with MI250x GPUs , rocm 5.3.0 and cray-mpich/8.1.18 on May/03/2023

Demo: Inter-node GPU-to-GPU Communication Bandwidth on LUMI

```
mghazimi@uan02:~/OMB/osu_benchmark> srun -N 2 -n 2 ./build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw D D
# OSU MPI-ROCM Bandwidth Test v7.0
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
1                       2.07
2                       4.13
4                       8.28
8                      16.60
16                     33.19
32                     66.45
64                    132.14
128                   264.68
256                   498.90
512                   996.77
1024                 1987.55
2048                 3975.71
4096                 7921.45
8192                15705.86
16384               20549.96
32768               21298.89
65536               22707.28
131072              23268.52
262144              23647.31
524288              23827.88
1048576             23903.00
2097152             23947.73
4194304             23968.83
```



This test has been executed as a demo (not performance claim) on LUMI pilot partition with MI250x GPUs, rocm 5.2.3 and cray-mpich/8.1.18 on May/03/2023

Demo: GPU-Aware Collective Communication

```
$srun -N 2 -n 8 --ntasks-per-node=4 ./build/libexec/osu-micro-benchmarks/mpi/collective/osu_allreduce –m 128 -d rocm
# OSU MPI-ROCM Allreduce Latency Test v7.0
# Size       Avg Latency(us)
4                    5.23
8                    5.22
16                   5.23
32                   5.22
64                   5.26
128                  5.57
```

4 ranks on node 0
4 ranks on node 1

```
srun -N 1 -n 8 --ntasks-per-node=8 ./build/libexec/osu-micro-benchmarks/mpi/collective/osu_allreduce –m 128 -d rocm
# OSU MPI-ROCM Allreduce Latency Test v7.0
# Size       Avg Latency(us)
4                    1.27
8                    1.24
16                   1.27
32                   1.27
64                   1.32
128                  1.39
```

8 ranks on node 0

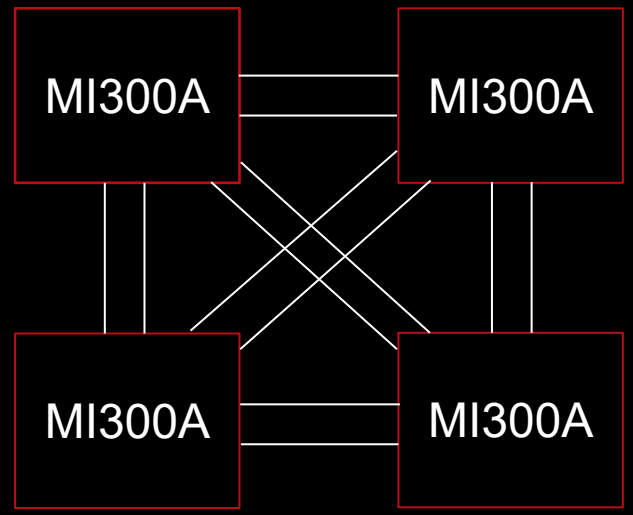
These tests have been executed as a demo (not performance claim) on LUMI pilot partition with MI250x GPUs , rocm 5.2.3 and cray-mpich/8.1.18 on May/03/2023

Network Topology on AAC Nodes with MI300A

rocm-smi --showtopo

```
============================== Link Type between two GPUs ==============================
       GPU0         GPU1         GPU2         GPU3
GPU0   0            XGMI         XGMI         XGMI
GPU1   XGMI         0            XGMI         XGMI
GPU2   XGMI         XGMI         0            XGMI
GPU3   XGMI         XGMI         XGMI         0

================================== Numa Nodes ==================================
GPU[0]          : (Topology) Numa Node: 0
GPU[0]          : (Topology) Numa Affinity: 0
GPU[1]          : (Topology) Numa Node: 1
GPU[1]          : (Topology) Numa Affinity: 1
GPU[2]          : (Topology) Numa Node: 2
GPU[2]          : (Topology) Numa Affinity: 2
GPU[3]          : (Topology) Numa Node: 3
GPU[3]          : (Topology) Numa Affinity: 3
```



Demo: Communication Example on MI300A System

```
$mpirun -n 2 --map-by package --mca pml ucx -x HIP_VISIBLE_DEVICES=1,2 ./install/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m
$((4194304*16)):$((4194304*16)) D D

# OSU MPI-ROCM Bandwidth Test v7.2
# Send Buffer on DEVICE (D) and Receive Buffer on DEVICE (D)
# Size      Bandwidth (MB/s)
# Datatype: MPI_CHAR.
67108864          90628.39
```

hipMalloc'ed buffer

```
$mpirun -n 2 --map-by package --mca pml ucx -x HIP_VISIBLE_DEVICES=1,2 ./install/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m
4194304:4194304

# OSU MPI Bandwidth Test v7.2
# Size      Bandwidth (MB/s)
# Datatype: MPI_CHAR.
4194304           22986.99
```

System allocated buffer e.g., malloc

These tests have been executed as a demo (not performance claim) on PP nodes with MI300A APUs , rocm 6.2.0 and OpenMPI 5.0.2 /UCX 1.16.0 on April/17/2024. Contact HPE for detailed performance specs

ROCm Collective Communication Library (RCCL)

- Library of standard communication routines for GPUs
- Implementing collectives e.g., all-reduce, all-gather, reduce, broadcast, reduce-scatter, gather, scatter, and all-to-all
- Initial support for direct GPU-to-GPU send and receive operations
- It is used as backend for collective communication for AI applications e.g., in PyTorch
- RCCL-test is a benchmark suite to evaluate the performance and correctness of RCCL operations

Demo: RCCL test on MI300A

git clone https://github.com/ROCm/rccl-tests.git
cd rccl-tests/
make MPI=1 MPI_HOME=/opt/rocmplus-6.1.0/openmpi/ HIP_HOME=/opt/rocm/

#After successful build, you should be able to see the executables in ./build directory. You can run the collectives with:

./build/all_reduce_perf -b 4M -e 128M -f 2 -g 4

Run with 4 GPUs

multiplication factor between sizes

Run for 4M to 128M messages

```
sghazimi@d9dbb2d52d84:~/rccl/rccl-tests$ ./build/all_reduce_perf -b 4M -e 128M -f 2 -g 4
# nThread 1 nGpus 4 minBytes 4194304 maxBytes 134217728 step: 2(factor) warmup iters: 5 iters: 20 agg iters: 1 validation: 1 graph: 0
#
rccl-tests: Version develop:990f88c
# Using devices
#   Rank  0 Pid 1004519 on d9dbb2d52d84 device  0 [0000:01:00.0] AMD Instinct MI300A
#   Rank  1 Pid 1004519 on d9dbb2d52d84 device  1 [0001:01:00.0] AMD Instinct MI300A
#   Rank  2 Pid 1004519 on d9dbb2d52d84 device  2 [0002:01:00.0] AMD Instinct MI300A
#   Rank  3 Pid 1004519 on d9dbb2d52d84 device  3 [0003:01:00.0] AMD Instinct MI300A
#
#                                                     out-of-place                       in-place
#       size         count      type   redop    root     time   algbw   busbw #wrong     time   algbw   busbw #wrong
#        (B)    (elements)                               (us)  (GB/s)  (GB/s)            (us)  (GB/s)  (GB/s)
     4194304       1048576     float     sum      -1    91.43   45.88   68.81      0    85.67   48.96   73.44      0
     8388608       2097152     float     sum      -1    128.0   65.53   98.30      0    135.4   61.97   92.95      0
    16777216       4194304     float     sum      -1    226.2   74.16  111.25      0    246.0   68.19  102.29      0
    33554432       8388608     float     sum      -1    409.7   81.91  122.86      0    441.4   76.01  114.01      0
    67108864      16777216     float     sum      -1    789.2   85.04  127.56      0    819.6   81.88  122.83      0
   134217728      33554432     float     sum      -1   1567.0   85.65  128.48      0   1612.5   83.24  124.85      0
```


These tests have been executed as a demo (not performance claim) on AAC nodes with MI300A APUs , rocm 6.1.0 and OpenMPI 5.0.3/UCX 1.16.0 on April/22/2024

Summary

- Many MPI implementations including Cray-MPICH and OpenMPI support GPU-Aware communication with ROCm™
- Using OSU microbenchmark to measure communication bandwidth and latency between GPUs
- Measured communication bandwidth on LUMI (MI250) and AAC (MI210)
  - The communication bandwidth between GPUs depend on
    - Infinity Fabric™ connected vs PCIe® connected
    - Using SDMA vs blit kernel
    - Number of Infinity Fabric™ links
    - Number of hops
- Measured communication bandwidth on systems with MI300A
- Measured collective communication performance With RCCL

Exercises

Instructions to Run GPU-Aware MPI Examples on AAC

- Two MPI implementations are available in the AAC Training System – OpenMPI and MVAPICH. We'll work with OpenMPI, but MVAPICH is similar
  - Get example code
    - git clone https://github.com/AMD/HPCTrainingExamples
    - cd ~/HPCTrainingExamples/MPI-examples
  - Setup the environment
    - `module load openmpi rocm`
  - Build code – first we override the underlying compiler for the OpenMPI compiler wrapper and then compile with mpicxx
    - export OMPI_CXX=hipcc
    - mpicxx -o ./pt2pt ./pt2pt.cpp
    - `mpirun -n 2 ./pt2pt`

OSU Micro-Benchmarks (OMB)

- Retrieving OSU Benchmark code
  - `mkdir OMB`
  - `cd OMB`
  - `wget https://mvapich.cse.ohio-state.edu/download/mvapich/osu-micro-benchmarks-7.3.tar.gz`
  - `tar -xvf osu-micro-benchmarks-7.3.tar.gz`
  - `cd osu-micro-benchmarks-7.3`
  - `module load rocm openmpi`
- Building OMB with OpenMPI (AAC Cloud)
  - `./configure --prefix=`pwd`/../build/ CC=`which mpicc` CXX=`which mpicxx` \`
  - `CPPFLAGS=-D__HIP_PLATFORM_AMD__=1 --enable-rocm --with-rocm=${ROCM_PATH}`
  - `make -j12`
  - `make install`

Enable rocm extension

- If you get the error "cannot include hip/hip_runtime_api.h", search for `__HIP_PLATFORM_HCC__` and replace it with `__HIP_PLATFORM_AMD__` in configure.ac and configure files.
- Running benchmarks
  - `ls -l ../build/libexec/osu-micro-benchmarks/mpi`
  - `export HIP_VISIBLE_DEVICES=0,1`
  - `mpirun -N 2 -n 2 -mca pml ucx ../build/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw -m 10240000`

Disclaimer

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED ‘AS IS.” AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Third-party content is licensed to you directly by the third party that owns the content and is not licensed to you by AMD. ALL LINKED THIRD-PARTY CONTENT IS PROVIDED “AS IS” WITHOUT A WARRANTY OF ANY KIND. USE OF SUCH THIRD-PARTY CONTENT IS DONE AT YOUR SOLE DISCRETION AND UNDER NO CIRCUMSTANCES WILL AMD BE LIABLE TO YOU FOR ANY THIRD-PARTY CONTENT. YOU ASSUME ALL RISK AND ARE SOLELY RESPONSIBLE FOR ANY DAMAGES THAT MAY ARISE FROM YOUR USE OF THIRD-PARTY CONTENT.