Nvidia GPUDirect Storage with IBM Spectrum Scale

Spectrum Scale User Group
June 30th, 2022
London

Dr. Ingo Meents

Disclaimer

- This information is provided on an "AS IS" basis without warranty of any kind, express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Some jurisdictions do not allow disclaimers of express or implied warranties in certain transactions; therefore, this statement may not apply to you.
- This information is provided for information purposes only as a high level overview of possible future products. PRODUCT SPECIFICATIONS, ANNOUNCE DATES, AND OTHER INOFORMATION CONTAINED HEREIN ARE SUBJECT TO CHANGE AND WITHDRAWAL WITHOUT NOTICE.
- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.
- IBM reserves the right to change product specifications and offerings at any time without notice. This publication could include technical inaccuracies or typographical errors. References herein to IBM products and services do not imply that IBM intends to make them available in all countries.

IBM Spectrum Scale

Nvidia GPUDirect Storage with IBM Spectrum Scale

Spectrum Scale User Group
June 30th, 2022
London

Dr. Ingo Meents
IT Architect
Spectrum Scale Development
IBM Systems Group

Agenda

- Introduction - Why do we want GPUDirect Storage?
- GPUDirect Storage – What is it?
- GDS READ data path in Spectrum Scale
- Performance numbers
- How to use (HW & SW Prerequisites)
- Use of cuFileRead
- References

Why GPUDirect Storage?

Short latencies & ***High throughput***

for GPU accelerated **AI** and **HPC** applications

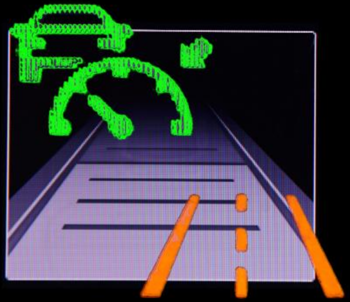
→ Up to ***2x*** improvements

Weather Forecasting
deepCAM inference

Predicting extreme
weather faster

Autonomous Driving

Data ingest
Training
Simulation

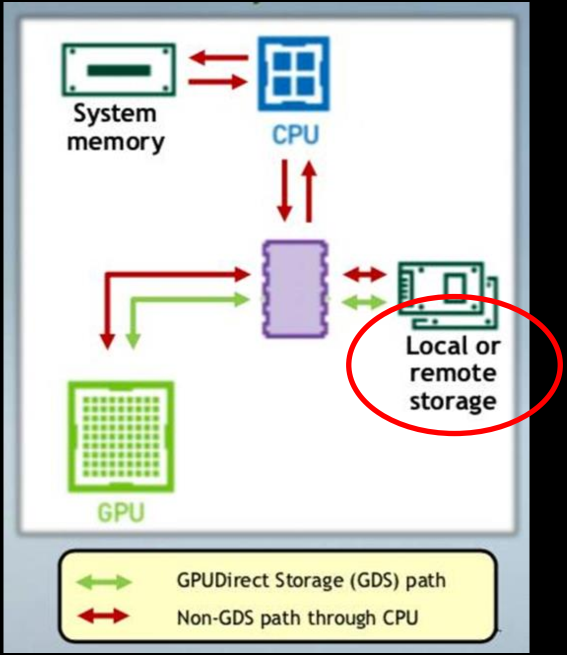


Oil and gas
exploration

4D Seismic imaging
for reservoir
mapping

What is GPU Direct Storage?

- Accelerating data movement between GPUs and storage
- Nvidia technology to keep GPUs busy
- **Direct Path Between Storage and GPU Memory**
- Based on (Remote) Direct Memory Access
  - Higher throughput
  - Lower latencies
  - Lower CPU utilization
- **API for applications: CUDA cuFile library**
- https://developer.nvidia.com/blog/gpudirect-storage/



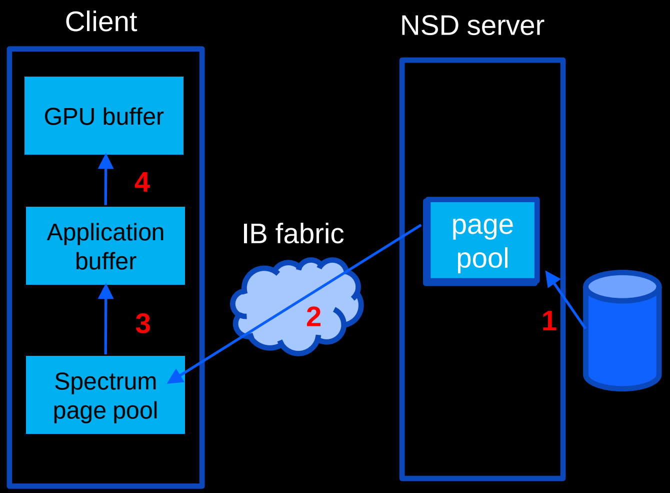
Source: Nvidia

GPUDirect Storage (GDS) for Spectrum Scale

Data path for a **READ** into a GPU buffer

Storage to GPU buffer **without** GDS:

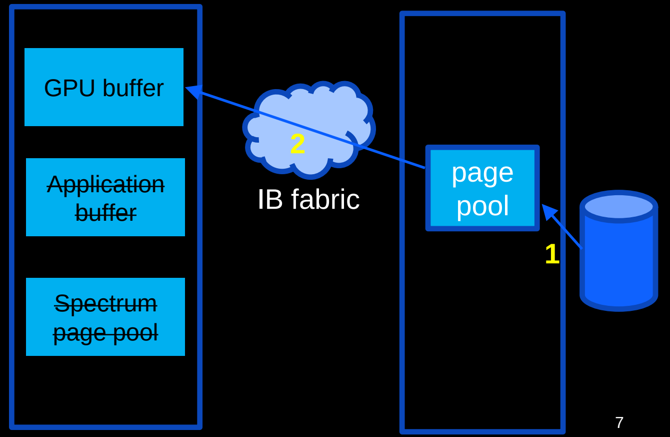
4 data transfers on path from storage media to application GPU buffer



Storage to GPU buffer **with** GDS:

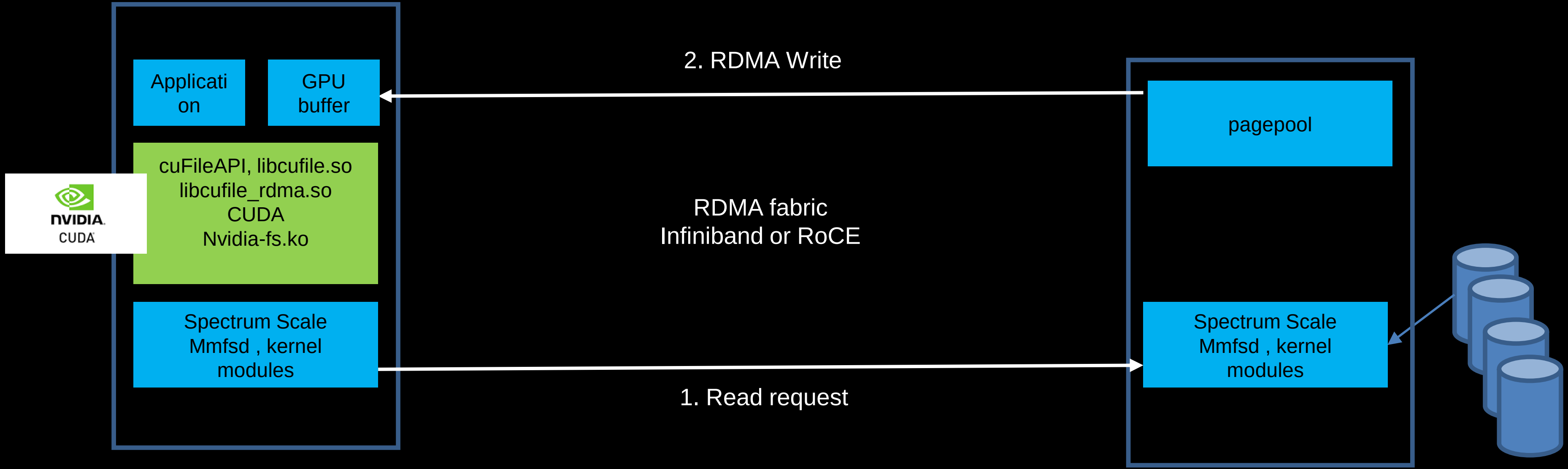
Two data transfers in path are eliminated.
increased throughput, reduced latency

Client CPU copy overhead reduced.
more CPU cycles for client application



Storage for AI and HPC

GPUDirect Storage (GDS) for Spectrum Scale

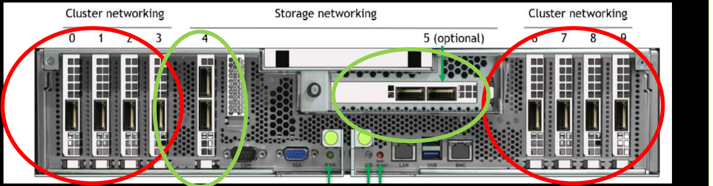


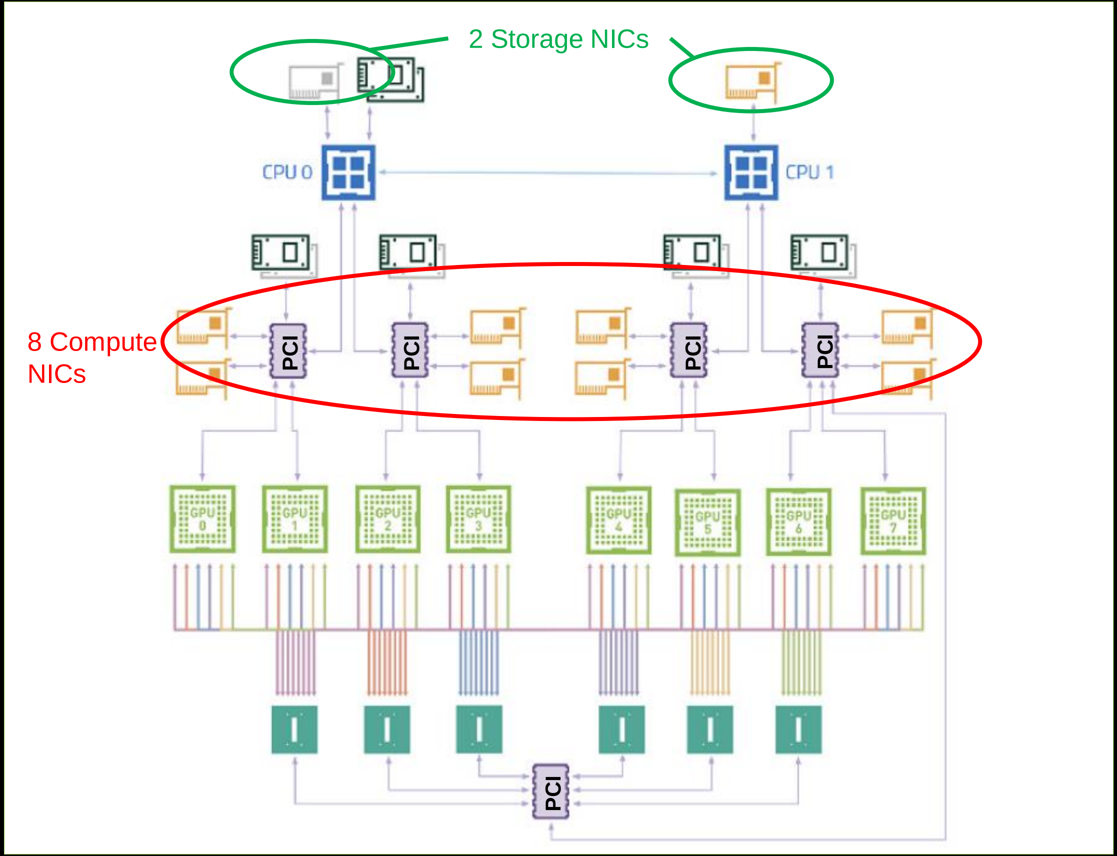
Client-side components (e.g. gpfs client, worker node)

- GPU accelerated applications with NVIDIA A100 GPUs
- infuse AI across organization

Server-side components (ESS, traditional NSD server)

Nvidia DGX A100



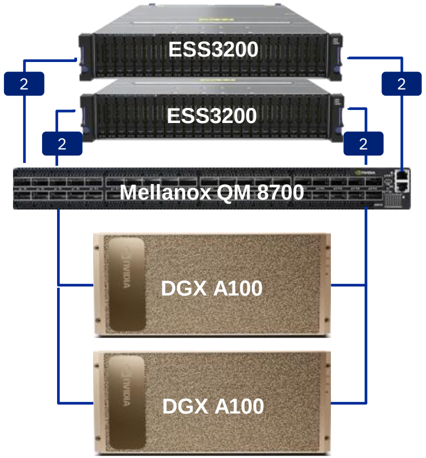


2 Storage NICs

8 Compute NICs

Pictures from DGX A100 User guide, https://docs.nvidia.com/dgx/

GDS Read Throughput – Linear scaling - IB



2 ESS 3200: 8 x HDR links
~200 GB/sec max

2 DGX A100: 4 x HDR links
~100 GB/sec max

Use of storage links

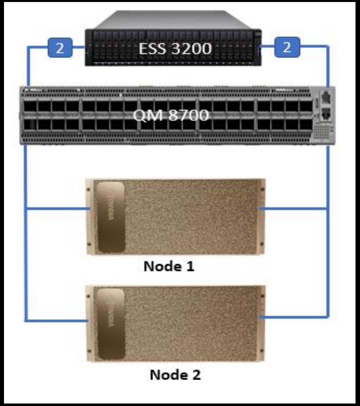
		Scenario 1	Scenario 2
		2 x ESS 3200 1 x DGX A100	2 x ESS 3200 2 x DGX A100
Throughput	Direct IO + cudaMemCopy	22 GB/s	45 GB/s
	GDS	49 GB/s	86 GB/s

Streaming Benchmark:

- NVIDIA “gdsio” utility
- 8 GPUs per DGX A100
- 2 or 4 threads per GPU
- 1M I/O size
- Data in GNR cache on ESS server

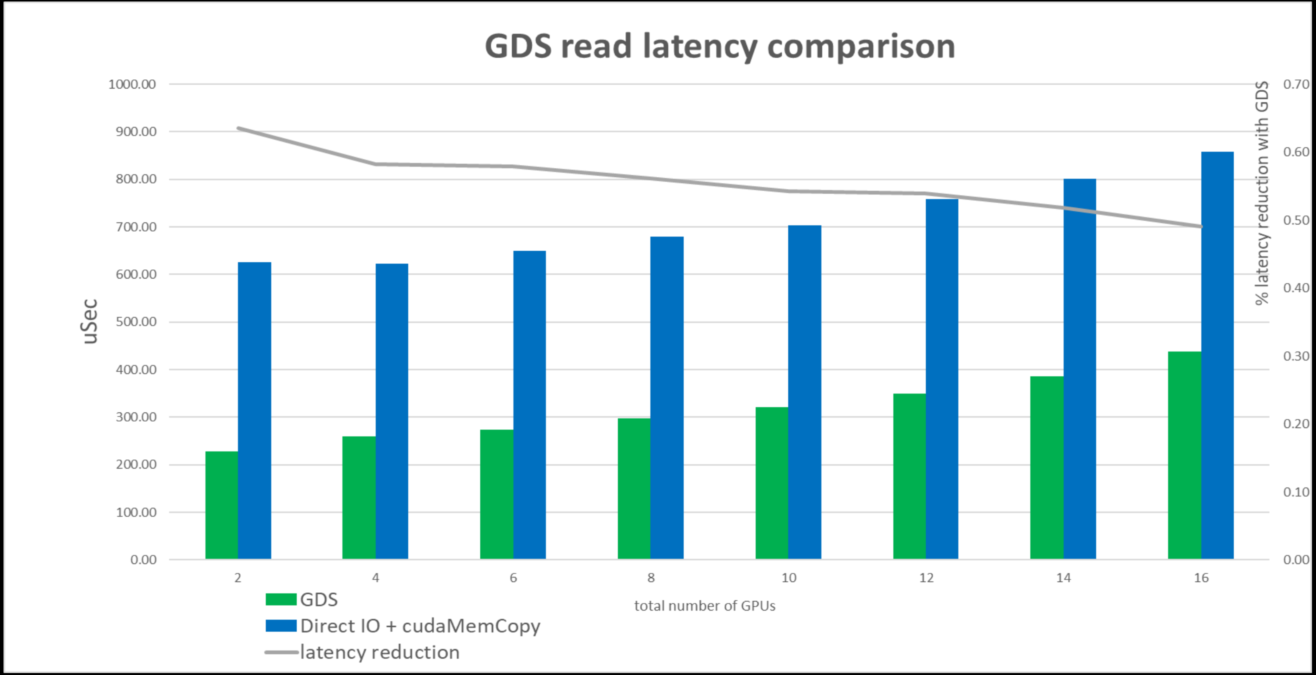
Typical throughput improvement for DGX A100 with GDS is approx. 2x when the storage and network support the throughput.

GDS Read Latency - IB



1 ESS 3200: 4 x HDR links

2 DGX A100: 4 x HDR links (storage links)



Benchmark: NVIDIA 'gdsio' benchmark with 1M I/O size and 2 threads per GPU

Typical latency reduction with GPU Direct Storage is 50%.

GDS Read Performance - IB

Experimental config using DGX-A100 compute NICs (*)

Maximum theoretical throughput:
ESS 3200: 4 x HDR = 100 GB/s max
DGX-A100 compute NICs: 8 x HDR = 200 GB/s max

	Scenario 3 (Compute NICs)
	2 x ESS 3200 2 x DGX A100
Aggregate GDS throughput	196 GB/s

> 95% of max fabric bandwidth for 2 x ESS 3200

Benchmark details:
- NVIDIA “gdsio” utility
- 8 GPUs per DGX A100
- 4 threads per GPU
- 1M I/O size
- Data in GNR cache on ESS servers

(*) *Performance numbers shown here with NVIDIA GPUDirect Storage on NVIDIA DGX A100 slots 0-3 and 6-9 (“compute NICs”) are not the officially supported NVIDIA DGX A100 network configuration and are for experimental use only. Sharing the same network adapters for both compute and storage may impact the performance of any benchmarks previously published by NVIDIA on DGX A100 systems.*

What do I need to use GPUDirect Storage with Spectrum Scale?

https://www.ibm.com/docs/en/STXKQY/gpfsclustersfaq.html#gds

Hardware

- x86_64 client with GPU
  - Data Center and Quadro (desktop) GPUs with compute capability > 6
- Storage Server (NSD server, ESS; x86_64 or ppc64le)
- RDMA Fabric
  - Mellanox CX5 / CX6
  - Switch: IB or RoCE

Spectrum Scale

- 5.1.2 (Read/IB)
- 5.1.2.1 (Write in compatibility mode/IB)
- 5.1.3 RoCE (Read, Write in compatibility mode)

Operating system

- RHEL 8.6
- Ubtuntu 20.04

MOFED

- Mellanox OFED stack
- Currently recommended:

  MLNX_OFED_LINUX-5.4-1.0.3.0

CUDA

- CUDA 11.4.2, 11.5.1, 11.6.2
- Please look at FAQ for issues and recommendations
- CUDA C/C++ program
- Nvidia DALI (data loading library)

How to exploit – cuFileRead – CUDA Application

```
// open driver
status = cuFileDriverOpen();
```

Triggers registration with GPFS

```
// register filehandle with CUDA
cf_descr.handle.fd = fd;
cf_descr.type = CU_FILE_HANDLE_TYPE_OPAQUE_FD;
status = cuFileHandleRegister(&cf_handle, &cf_descr);
```

POSIX file handle

Registers file handle with CUDA for use in cuFileRead

```
// reading data from file into device memory
ret = cuFileRead(cf_handle, devPtr, size, 0, 0);
```

Do GDS IO

```
// deregister the handle from cuFile
(void) cuFileHandleDeregister(cf_handle);
```

Triggers de-registration with GPFS

Documentation

Spectrum Scale Knowledge Center

https://www.ibm.com/docs/en/spectrum-scale/5.1.3?topic=summary-changes

https://www.ibm.com/docs/en/spectrum-scale/5.1.4?topic=architecture-gpudirect-storage-support-spectrum-scale

NVIDIA GDS Documentation:

https://docs.nvidia.com/gpudirect-storage/index.html

https://developer.nvidia.com/gpudirect-storage

Thanks for your attendance!

Contact:

Ingo Meents
IT Architect
IBM Research & Development, Germany

meents@de.ibm.com

Special thanks to: Swen Schillig, Ralph Würthner, Meng Lu Wang, Felipe Knop, John Divirgilio

Trademarks

Thank you for using IBM Spectrum Scale!