HP ProLiant SL270s Gen8 Server

Evaluation Report

Thomas Schoenemeyer, Hussein Harake and Daniel Peter
Swiss National Supercomputing Centre (CSCS), Lugano
Institute of Geophysics, ETH Zürich

schoenemeyer@cscs.ch hussein@cscs.ch daniel.peter@erdw.ethz.ch

Abstract – We evaluated the HP ProLiant SL270s Gen8 server with Intel Xeon E5-2600 processor and fully populated with eight Nvidia K20x cards.

The evaluation is based on examples provided with the Nvidia SDK as well as benchmarks from the SHOC benchmark suite.

The server is an interesting option for scientists whose applications fit onto the memory of eight GPU devices and can use multiple devices simultaneously.

1 Introduction

This report summarizes the performance measurements carried out with one HP ProLiant SL270s Gen8 server with 4U half width trays, which are designed for high GPU density and supports up to 8 GPUs.
The HP ProLiant SL270s Gen8 Server is part of the HP ProLiant SL6500 Scalable System of server solutions optimized for the High Performance Computing market segment. The 4U half-width tray is designed and optimized for extreme GPU configurations and is ideally suited for applications ranging from quantum chemistry, molecular dynamics, weather forecasting, seismic processing and data analytics.

2 Server description

2.1 Hardware

The server shown in Figure 1 is a two socket Intel server: 8 core Intel® Xeon® CPU E5-2670 @ 2.60GHz and with 64GB DDR3 Memory (8x8GB DIMMs), two 1 Gb Ethernet ports, one CX-2 based 10 Gb Ethernet port (SFP+) and on board Infiniband CX-2 port (QSFP). The system offers 8 PCIe 2.0 expansion slots with x16 bus width with full-length and full-height form factor. Maximum number of NVIDIA M2070Q, K2, K10, K20, and K20x modules and Intel
Xeon PhiTM 5110p coprocessors supported is 8 [1]. The maximum number of Intel Xeon PhiTM 7120p coprocessors is 4-6 depending on configuration. The certification for the K40 modules is on-going.
The SL270s server in our test environment was fully populated with 8 K20x modules. The board architecture is outlined in Figure 2.

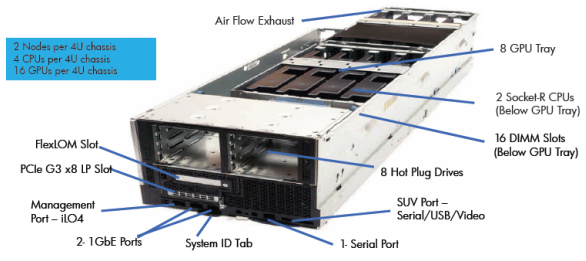

Figure 1: HP ProLiant SL270s Gen8 server

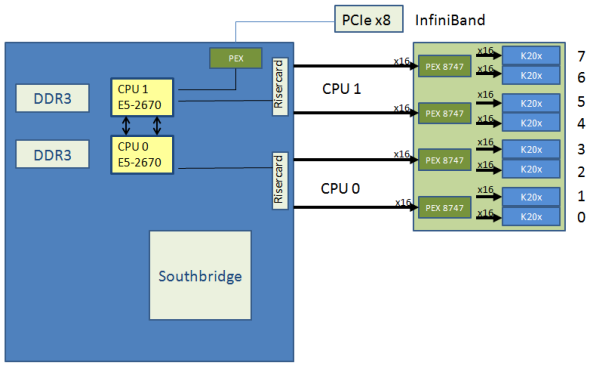

Figure 2: Schematic board architecture

2.2 Software

All our tests were carried out using the following software stack:

- SL 6.4 Linux Kernel 2.6.32
- Intel Compiler Version 13
- gcc 4.8.1
- Nvidia SDK 5.5.22
- mvapich2 1.9
- openmpi 1.6.5
- Nvidia Driver 319.60
- OFED 2.0-3.0.0

2.3 Nvidia K20X

The K20x has the [2] following specifications

- 1.31 GFlops Peak DP
- 6 GB memory size (ECC off)
- 250 GB/sec memory bandwidth (ECC off)
- 2688 CUDA cores

As one of the key advantages, this HP-server allows the deployment of standard K20x or K40 GPU cards available from any system integrator.

3 Peer-to-Peer Communication

The functionality of peer-to-peer (P2P) communication allows GPUs to exchange messages directly with each other without copying data to the host. Since the QPI link is incompatible with PCIe P2P specifications, this functionality cannot be used among all GPUs on this server. The full list is available with

```
./deviceQuery
export CUDA_VISIBLE_DEVICES="0,1,2,3,4,5,6,7"

> Peer access from Tesla K20Xm (GPU0) -> Tesla K20Xm (GPU1) : Yes
> Peer access from Tesla K20Xm (GPU0) -> Tesla K20Xm (GPU2) : Yes
> Peer access from Tesla K20Xm (GPU0) -> Tesla K20Xm (GPU3) : Yes
> Peer access from Tesla K20Xm (GPU0) -> Tesla K20Xm (GPU4) : No
...
```

According to the board architecture shown in Figure 2, four GPUs are attached to CPU 0 and to CPU 1, respectively. This allows an efficient intra-node point-to-point communication for multiple GPUs.

The result of this test is as expected, due to the architecture of the server P2P is only possible among GPUs 0, 1, 2, 3 and among GPUs 4, 5, 6 and 7. Communication among

GPUs of the first group and GPUs of the second group will still work, but will get staged via host memory.

GPUs attached to the same PCI switch show a bandwidth of 6.2 GB/s unidirectional and 12.2 GB/s bidirectional. As shown in the log file below (Figure 3), the data bandwidth between GPUs attached to the same riser card give (e.g. 4 and 6) falls off to 5.4 GB/s unidirectional and 10.2 GB/s bidirectional.

```
Testing devices 4 and 5
  Unidirectional bandwidth 4->5: 6.219 Gbytes/sec
  Unidirectional bandwidth 5->4: 6.221 Gbytes/sec
  Bidirectional bandwidth 5<->4: 12.257 Gbytes/sec
Testing devices 4 and 6
  Unidirectional bandwidth 4->6: 5.372 Gbytes/sec
  Unidirectional bandwidth 6->4: 5.370 Gbytes/sec
  Bidirectional bandwidth 6<->4: 10.196 Gbytes/sec
Testing devices 4 and 7
  Unidirectional bandwidth 4->7: 5.371 Gbytes/sec
  Unidirectional bandwidth 7->4: 5.374 Gbytes/sec
  Bidirectional bandwidth 7<->4: 10.194 Gbytes/sec
Testing devices 5 and 6
  Unidirectional bandwidth 5->6: 5.371 Gbytes/sec
  Unidirectional bandwidth 6->5: 5.369 Gbytes/sec
  Bidirectional bandwidth 6<->5: 10.195 Gbytes/sec
Testing devices 5 and 7
  Unidirectional bandwidth 5->7: 5.373 Gbytes/sec
  Unidirectional bandwidth 7->5: 5.372 Gbytes/sec
  Bidirectional bandwidth 7<->5: 10.193 Gbytes/sec
Testing devices 6 and 7
  Unidirectional bandwidth 6->7: 6.218 Gbytes/sec
  Unidirectional bandwidth 7->6: 6.219 Gbytes/sec
  Bidirectional bandwidth 7<->6: 12.242 Gbytes/sec
```

Figure 3: Logfile of P2P test

Inter-node GPUDirect via RDMA is therefore possible only with GPUs 4,5,6,7 connected to the same riser card as the PCIe slot reserved for the IB host adapter.

4 Benchmark results

4.1 Bandwidth Measurements

The bandwidth test included in the Nvidia SDK measures host to device bandwidth, device to host bandwidth and device to device bandwidth. We launched the ./bandwidthTest --device=all

With increasing the number of GPUs with export CUDA_VISIBLE_DEVICES="0,1,..7" We used pinned memory. According to the PCIe Gen2 speed of the K20x devices the bandwidth per device is roughly 6.3 GB/s and the aggregated throughput scales with the number of devices (Figure 4). The P2P among GPUs attached to CPU 0 and among GPUs attached to CPU1 allows a very high bandwidth of about 180 GB/s and scales ideally with the number of GPUs (Figure 5), which is close to the maximum theoretical bandwidth when ECC is on.

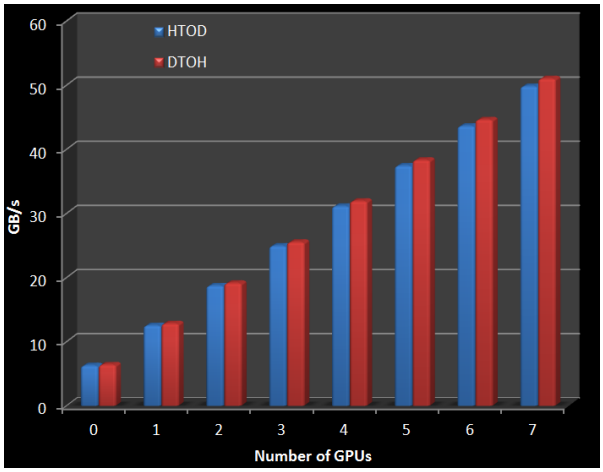

Figure 4: Host-to-device and device-to-host memory, pinned

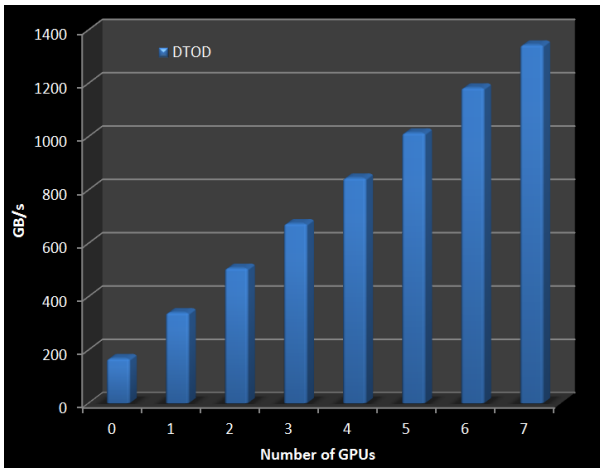

Figure 5: Aggregated device-to-device memory bandwidth

4.2 N-body

N-body is described in detail in [3]. This kind of simulation technique approximates many computational science problems such astrophysics, protein folding and turbulent fluid flows. The all-pairs approach to N-body simulation evaluates all pair-wise interactions among the N bodies and requires a substantial time to compute and is therefore an interesting target for acceleration. NVIDIA ported the all-pairs computational kernel using the CUDA programming model.

We launched the benchmark for problem sizes with 1024K particles with:

./nbody -benchmark -numbodies=1024000 -fp64 -numdevices=n

A patch was supplied by Nvidia [4], because the current version of the SDK gives wrong performance results for more than 3 GPUs.

The scalability is excellent for up to 5 GPUs, above that no further performance increase is observed (Figure 6).
Possibly a higher number of particles will result in a better scalability, however the main memory of the server did not allow to run larger cases.

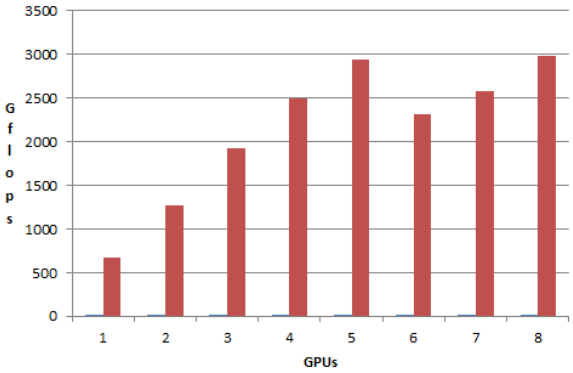

Figure 6: Nbody with 1024000 particles

4.3 MonteCarlo multi-GPU implementation

This code is the implementation of the Monte Carlo approach to option pricing, written in CUDA.

We used the streamed version that uses one CPU thread and handles all GPUs.

Because the Monte Carlo pricing of each option is independent of all others, the computation can be distributed across multiple CUDA-capable GPUs present in the system. The input options are divided into contiguous subsets (the number of subsets equals the number of CUDA-capable GPUs installed in the system).

The code uses all CUDA devices available on the server, the only way to control that is to set the environment variable accordingly with

export CUDA_VISIBLE_DEVICES="0,1,2,3,4,5,6,7"

for all GPUs on the system.

Number of simulation paths was set to 1048576 in MonteCarloMultiGPU.cpp.

./MonteCarloMultiGPU --method=streamed --type=double

The result is shown in Figure 7. The performance measured in options per second increases nearly linearly with the number of GPUs.

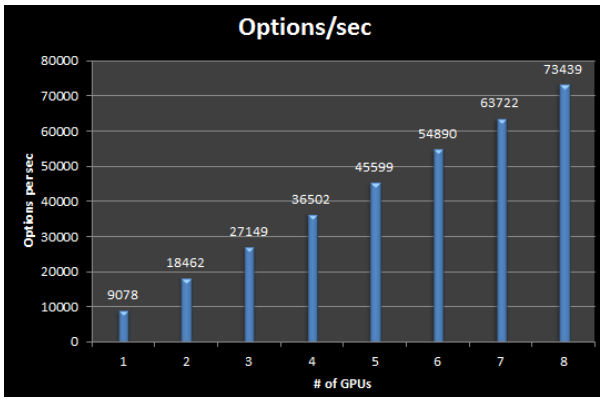

Figure 7: Options per second using 1048576 simulation paths

4.4 SPECFEM3D_Cartesian

SPECFEM3D_Cartesian [5] simulates 3D seismic wave propagation based on the spectral-element method. It can handle topography and elastic, viscoelastic, acoustic, poroelastic, anisotropic models, and has adjoint capabilities embedded to compute sensitivity kernels for gradient-based inversions. The simulation run 1000 time steps and calculated 20480 elements per MPI process. Each MPI Process used one GPU. This code shows an excellent scalability, the elapsed time per timestep decreases nearly ideally with the number of GPUs (Figure 8).

Figure 9 shows the scalability for the CPU version that has a similar behavior. The difference in the performance is about a factor 4.9 between a CPU socket and one GPU.

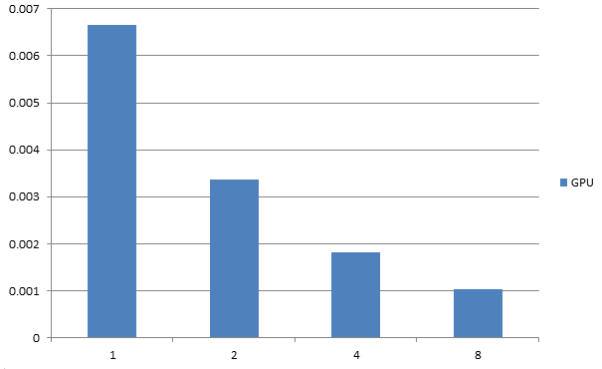

Figure 8: Elapsed time per timestep for 1, 2, 4 and 8 GPUs

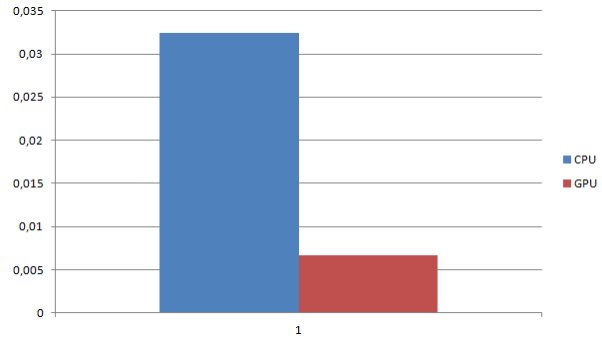

Figure9: Comparison between 1 CPU socket and 1 GPU

4.5 SHOC Benchmark for multi-GPU implementation

We used the GNU Compiler with Intel MPI 1.9.

Due to the server design, we are most interested in the multiple GPU benchmarks. We measured the server performance and scalability with two so-called True Parallel (TP) benchmarks taken from the latest SHOC source [7].

For each GPU one MPI-rank is used. The server allows us to measure the performance with up to 8 devices per node, including all communication. These algorithms are decomposed among the parallel tasks and may include CPU computation time.

4.5.1 Reduction

This benchmark measures the performance of a sum reduction operation using the double precision floating point data [7]. The kernel performs first a partial reduction on a global input data array saving the partial results to the local shared memory. Finally, a reduction is computed over the local data array and the result is saved into a global output memory array. The result for up to 8 GPUs is shown in Figure 10 for the mean value of Allreduce-DP-Kernel.

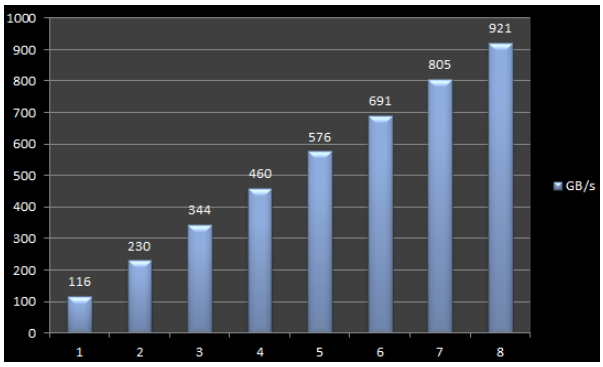

Figure 10: Bandwidth for reduction Allreduce-DP (mean)

4.5.2 Scan_DP

This benchmark measures the performance of the parallel prefix sum algorithm (also known as Scan) on a large array of floating point data and based on the method described in [10]. The results are shown in Figure 11, we picked the mean value of TPScan-DP-Kernel.

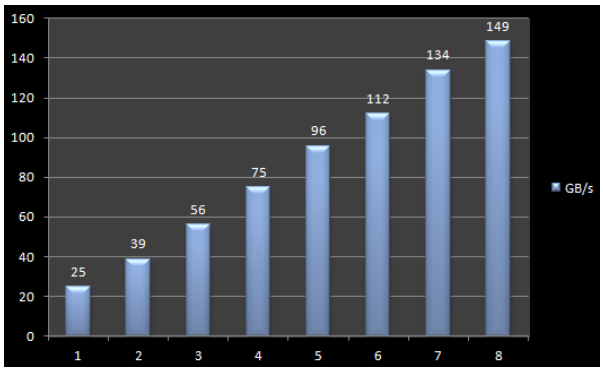

Figure 11: Bandwidth for Scan_DP

4.5.3 MD

MD measures the speed of a simple pairwise calculation of the Lennard-Jones potential from molecular dynamics. We run 8 copies of this benchmark to simulated an extreme throughput environment. We used the option –s 4 for the largest problem size. The Nvidia-smi output during the run is shown in Figure 12. The maximum power during the execution never exceeded 60W. The aggregated performance is 1684 Gflops and is almost similar to the single GPU performance multiplied by 8 (8 x 211 Gflops).

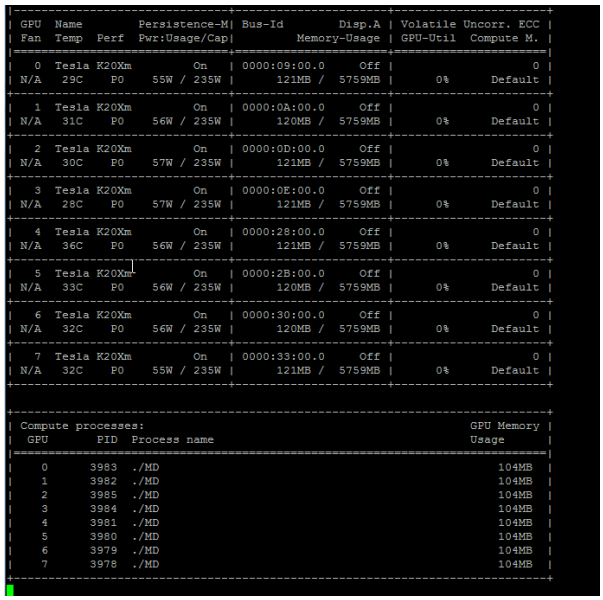

Figure 4: Nvidia-smi output during the MD throughput run

5 Conclusion

The overall design is extremely dense, since the 4U SL270s half-width server needs a SL6500 chassis of 4U. This chassis can accommodate two SL270s servers, all individually serviceable. In other words, one could house 16 GPUs in one 4U chassis. Despite the dense design, we did not observe any problems with stability or thermal issues. However we used only one half of the SL6500 chassis and all benchmarks run in less than 30 minutes. We also run the multi-GPU stress test [8] for one hour, the temperature of individual GPUs were constant after 10 min. The observed temperature range for all GPUs was between 47 and 60C.

The server is certainly an interesting option for applications, where an increase of the GPU:CPU ratio improves the sustained Gflops/Watt. Remarkable is the outstanding performance of the SPECFEM3D_Cartesian benchmark compared to the CPU version.

The aggregated memory bandwidth, scalability and throughput performance is excellent. P2P communication is only possible for GPUs attached to one CPU, since GPU P2P across QPI is not supported.

Since we had only one server, RMDA GPU P2P could not be tested, however inter-node P2P communication should be possible at least between 4 GPUs of each server.

6 Literature

1. HP SL270, Quick Specs, 2012.
2. Nvidia Tesla, Datasheet, November 2012
3. Lars Nyland, Mark Harris, Jan Prins, Fast N-Body Simulation with CUDA, distributed with the SDK4.0 and SDK4.1, 2011
4. Peter Messmer, Nvidia, personal communication, 2013
5. SPECFEMD 3D Cartesian, User Manual, September 2013
6. Anthony Danalis, Scalable Heterogeneous Computing Benchmark Suite (SHOC), Tech. Report, Oak Ridge National Laboratory, University of Tennessee, 2010.
7. Future Technologies Group, SHOC, The Scalable Heterogeneous computing benchmark suite, Manual Version 1.0.2 : March 2011
8. Gpuburn Multi-GPU CUDA Stress Test, 2013