Science & Technology
Facilities Council

Technical Report
DL-TR-2010-001

# Working Notes of Distributed Computing Group: Evaluation of Workstation Specialists' WS-Tesla x8 GPU Server

I Kozin, DJ Cable

June 2010

Enquiries about copyright, reproduction and requests for additional copies of this report should be addressed to:

Chadwick Library
Science and Technology Facilities Council
Daresbury Laboratory
Daresbury Science and Innovation Campus
Warrington
WA4 4AD

Tel: +44(0)1925 603397
Fax: +44(0)1925 603779
email: library@dl.ac.uk

Science and Technology Facilities Council reports are available online at: http://epubs.stfc.ac.uk/

**ISSN 1362-0207**

Neither the Council nor the Laboratory accept any responsibility for loss or damage arising from the use of information contained in any of their reports or in any communication about their tests or investigations.

# Working Notes of Distributed Computing Group

# Evaluation of Workstation Specialists' WS-Tesla x8 GPU Server

Recently we have published a report[1] where we evaluated our Tesla based GPU cluster. Our general conclusion was that GPU based solutions should now belong to the arsenal of available HPC technologies. For appropriate types of workload, the technology developed by NVIDIA offers high speedups, high density of compute power as well as considerable savings in terms of money and power. The momentum achieved by the success of Tesla C1060 continues with the release of Tesla C2050 (code name Fermi). People generally agree that general purpose GPU (GPGPU) is an interesting technology but once the decision to deploy a GPU cluster has been reached, most are confronted with the question: what is the most optimal configuration for GPU deployment in an HPC solution. As usual, the answer is "it depends". In some situations, e.g. visualisation server, a single card per server might be sufficient; for others, as high number of GPUs per server as possible; yet for others, highest bandwidth to GPUs is a must. Two other conclusions were drawn in the report: i) the importance of the power required by the host, ii) relatively poor scaling across GPU-enabled servers. These two points prompt to question the efficiency of a solution based around 1U or 2U host servers which typically cannot accommodate GPUs inside, especially if more than two GPUs are needed, and therefore require a Tesla server. One alternative is to use a 4U server with two Tylersburg Intel 5520 chipsets and eight PCI Express Gen2 16x-wide slots. The effective bandwidth per GPU will remain the same as in a server with one chipset, two PCI Express slots and PCI Express bridge in the Tesla server splitting each slot into two but the advantages are lower host power and the ability to drive eight GPUs in a single box. It is exactly this type of server which is being offered by WS-Tesla x8 GPU server from Workstation Specialists[2] and we were very delighted when we were allowed to evaluate it.

## Setup

The majority of Intel servers on the market today feature a single Chipset I/O Hub (IOH). Depending on the chipset, configurations with 36 and 24 PCI Express lanes are possible using Intel 5520 and Intel 5500 chipsets respectively[3]. This enables building servers with dual and single 16-lane PCI Express slots although in practice 8-lane wide slots are used more frequently in servers. The simplest topology assumes that the CPU sockets and IOH are directly connected using Intel's Quick Path Interconnect (QPI), see Figure 1A. For example, the host servers of our GPU cluster employ single Intel 5520 chipset and therefore support two 16-lane PCI Express slots plus a 4-lane slot used by Infiniband card. Using two GPUs in such a setup would be the most optimal solution from the bandwidth point of view. However it is also possible to increase the GPU density and make pairs of GPUs share the bandwidth of a 16-lane PCI Express slot (dual GPU setup). In the Tesla servers used in our cluster, this is done through PCI Express bridges and PCI Express adapter cards which plug into the hosts.

However Intel also envisaged a reference configuration with two Tylersburg chipsets, see Figure 1B. It allows increasing the number of full 16-lane PCI Express slots to four and using up to eight GPUs if the bandwidth is shared. This is the topology used by WS-Tesla x8 server. It is a dual socket 4U server with two chipsets supporting eight PCI Express Gen2 16-lane wide slots. Notice however that in contrast to the first topology which requires only one hop to pass the data between CPU and IOH, the second topology may need two hops.

The WS-Tesla x8 server used in our evaluation featured Intel Xeon E5520 processor and 24 GB of DDR3 RAM. Clocked at 2.27 GHz, the processor is the slowest in the Xeon E-range. However, since the heavy lifting is supposed to be done by the accelerators, 80 Watt TDP and low frequency should minimise the power budget

[1] "Comparison of traditional and GPU-based HPC solutions from Power/Performance point of view", Igor Kozin. November 2009. http://www.cse.scitech.ac.uk/disco/publications/WorkingNotes.Power.pdf

[2] http://www.workstationspecialist.com/
http://www.workstationspecialist.com/hpc/personal_super_computer/

[3] "Intel 5520 Chipset and Intel 5500 Chipset Datasheet", March 2009. http://www.intel.com/assets/pdf/datasheet/321328.pdf

required by the host. Besides, the E-range supports 1066 MHz DDR3 which delivers the bandwidth much higher than that of QPI. Therefore the host seems to be ideal in sense of minimising the power and maintaining the memory bandwidth to feed the CPUs. A bottleneck may appear in suboptimal bandwidth between IOH and GPUs due to sharing of PCI Express lanes if the maximal number of GPUs is used. However this should be no worse than the bandwidth offered by the Tesla server.

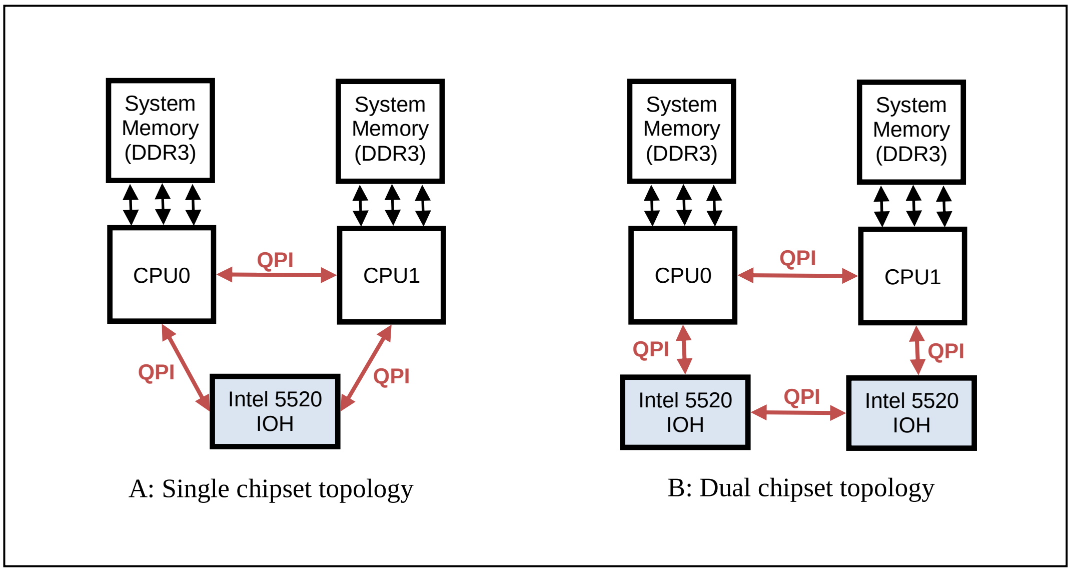


**Figure 1: Two possible topologies based on Intel 5520 Chipset I/O Hub (adapted from "Intel 5520 Chipset and Intel 5500 Chipset Datasheet").**

The picture below shows the rear view of WS-Tesla x8 server. The server came to us without GPUs and we installed eight M1060 cards taken temporary from two Tesla S1070 servers. It must be noted that M1060 is passively cooled but the blower fans seen in Figure 2 were perfectly adequate even under very heavy load. Another point worth noting is that there were no appropriate fixtures for the cards in the box since M1060 has no bracket and instead is screwed to the board of S1070 using four nuts. The cards were simply plugged into the slots of WS-Tesla x8 and not fixed to the chassis. There were no issues with this but that may be inadequate in production environment. In comparison, C1060 comes with a fixing bracket as well as with an active fan which allows using slightly higher clock and therefore takes more power. The server came with a single hard drive on which we installed Scientific Linux 5.3, the latest NVIDIA driver 195.36.08 and CUDA 2.3.

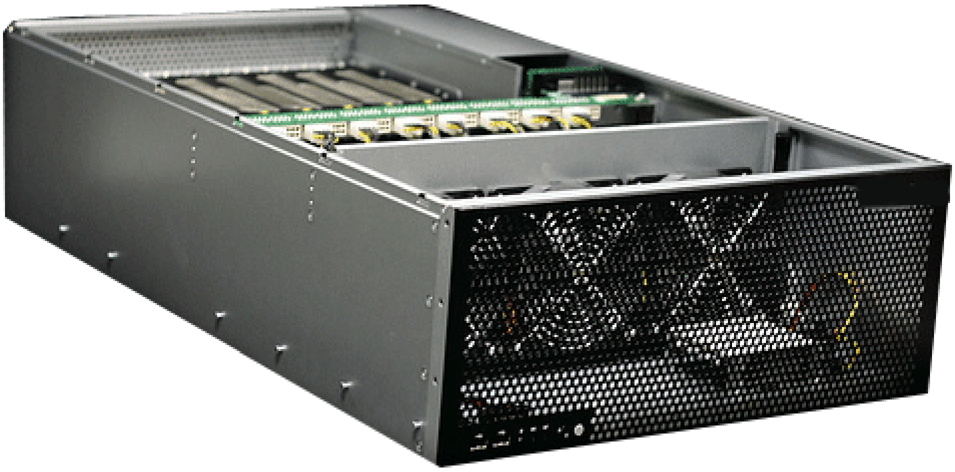
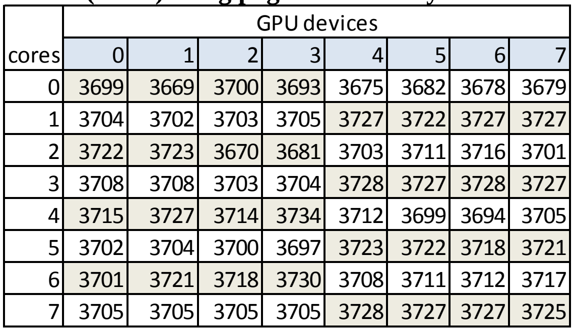


**Figure 2: Rear view of Workstation Specialists WS-Tesla x8 server with the top lid open.**

## Initial system assessment

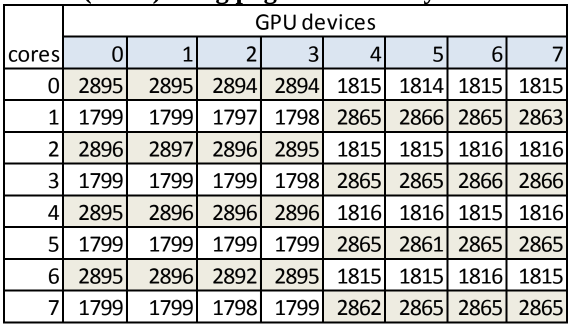
Although the topology of the system is symmetrical as follows from the Figure 1B, asymmetry may happen dynamically if a process is running on a CPU wants to access a GPU which is attached to an IOH two hops away from the CPU rather than one hop to the adjacent IOH. The latency of such an access will obviously take a bit longer but there is nothing unusual about it since if CPU0 tries to access memory attached to CPU1 it will also take a bit longer. Because GPU is meant to be a high throughput device it is more important to make sure that memory bandwidth can be sustained. Towards this end we assessed the bandwidth between host memory and GPU devices using the bandwidth test which can be found in the CUDA SDK. The test was run consecutively on across all cores and GPU devices with process to core binding using taskset tool. The results are presented in the tables below.

**Table 1: Host to device bandwidth (MB/s) using pageable memory.**



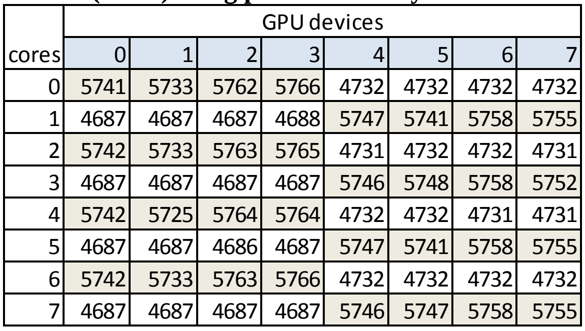
| | GPU devices | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| cores | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 3699 | 3669 | 3700 | 3693 | 3675 | 3682 | 3678 | 3679 |
| 1 | 3704 | 3702 | 3703 | 3705 | 3727 | 3722 | 3727 | 3727 |
| 2 | 3722 | 3723 | 3670 | 3681 | 3703 | 3711 | 3716 | 3701 |
| 3 | 3708 | 3708 | 3703 | 3704 | 3728 | 3727 | 3728 | 3727 |
| 4 | 3715 | 3727 | 3714 | 3734 | 3712 | 3699 | 3694 | 3705 |
| 5 | 3702 | 3704 | 3700 | 3697 | 3723 | 3722 | 3718 | 3721 |
| 6 | 3701 | 3721 | 3718 | 3730 | 3708 | 3711 | 3712 | 3717 |
| 7 | 3705 | 3705 | 3705 | 3705 | 3728 | 3727 | 3727 | 3725 |

**Table 2: Device to host bandwidth (MB/s) using pageable memory.**



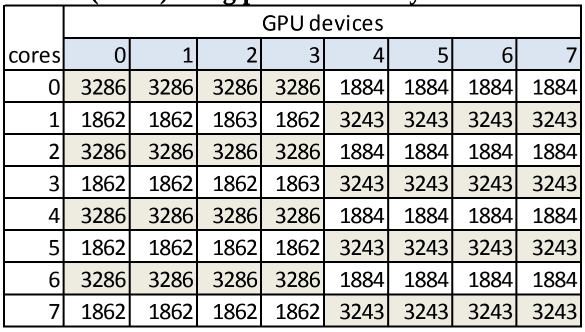
| | GPU devices | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| cores | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 2895 | 2895 | 2894 | 2894 | 1815 | 1814 | 1815 | 1815 |
| 1 | 1799 | 1799 | 1797 | 1798 | 2865 | 2866 | 2865 | 2863 |
| 2 | 2896 | 2897 | 2896 | 2895 | 1815 | 1815 | 1816 | 1816 |
| 3 | 1799 | 1799 | 1799 | 1798 | 2865 | 2865 | 2866 | 2866 |
| 4 | 2895 | 2896 | 2896 | 2896 | 1816 | 1816 | 1815 | 1816 |
| 5 | 1799 | 1799 | 1799 | 1799 | 2865 | 2861 | 2865 | 2865 |
| 6 | 2895 | 2896 | 2892 | 2895 | 1815 | 1815 | 1816 | 1815 |
| 7 | 1799 | 1799 | 1798 | 1799 | 2862 | 2865 | 2865 | 2865 |

**Table 3: Host to device bandwidth (MB/s) using pinned memory.**



| | GPU devices | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| cores | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 5741 | 5733 | 5762 | 5766 | 4732 | 4732 | 4732 | 4732 |
| 1 | 4687 | 4687 | 4687 | 4688 | 5747 | 5741 | 5758 | 5755 |
| 2 | 5742 | 5733 | 5763 | 5765 | 4731 | 4732 | 4732 | 4731 |
| 3 | 4687 | 4687 | 4687 | 4687 | 5746 | 5748 | 5758 | 5752 |
| 4 | 5742 | 5725 | 5764 | 5764 | 4732 | 4732 | 4731 | 4731 |
| 5 | 4687 | 4687 | 4686 | 4687 | 5747 | 5741 | 5758 | 5755 |
| 6 | 5742 | 5733 | 5763 | 5766 | 4732 | 4732 | 4732 | 4732 |
| 7 | 4687 | 4687 | 4687 | 4687 | 5746 | 5747 | 5758 | 5755 |

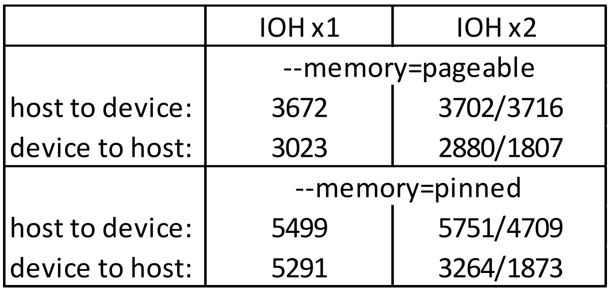
**Table 4: Device to host bandwidth (MB/s) using pinned memory.**



| | GPU devices | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| cores | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 3286 | 3286 | 3286 | 3286 | 1884 | 1884 | 1884 | 1884 |
| 1 | 1862 | 1862 | 1863 | 1862 | 3243 | 3243 | 3243 | 3243 |
| 2 | 3286 | 3286 | 3286 | 3286 | 1884 | 1884 | 1884 | 1884 |
| 3 | 1862 | 1862 | 1862 | 1863 | 3243 | 3243 | 3243 | 3243 |
| 4 | 3286 | 3286 | 3286 | 3286 | 1884 | 1884 | 1884 | 1884 |
| 5 | 1862 | 1862 | 1862 | 1862 | 3243 | 3243 | 3243 | 3243 |
| 6 | 3286 | 3286 | 3286 | 3286 | 1884 | 1884 | 1884 | 1884 |
| 7 | 1862 | 1862 | 1862 | 1862 | 3243 | 3243 | 3243 | 3243 |

The tables clearly demonstrate that GPU devices 0–3 have affinity to even cores which belong to CPU0 whereas GPU devices 4–7 have affinity to odd cores which belong to CPU1. Furthermore the numbers split into high and low depending on affinity. The only table which shows no splitting is Table 1 where host to device memory bandwidth is reported and the memory option was set to “pageable”. It is instructive to compare these results with the bandwidths we obtained on our GPU cluster[1] which is based on Supermicro single chipset motherboards. Table 5 shows that with the right affinity the bandwidths are comparable for all but device to host pinned memory transfers. This turns out to be a problem of dual IOH configurations. It seems that in a dual IOH setup device to host transfers cannot take the full advantage of pinned memory. The same is true for device to host bandwidths if there is no affinity. Overall we observe that device to host transfers are particularly penalised in the dual IOH configuration and the bandwidths of host to device transfers are roughly comparable. At this time we do not know when the dual IOH setup problem is going to be fixed but it is very clear that maintaining proper core to GPU binding is crucially important.

**Table 5: Comparison of bandwidths (MB/s) of data transfers between CPU and GPU in single and dual IOH configurations.**

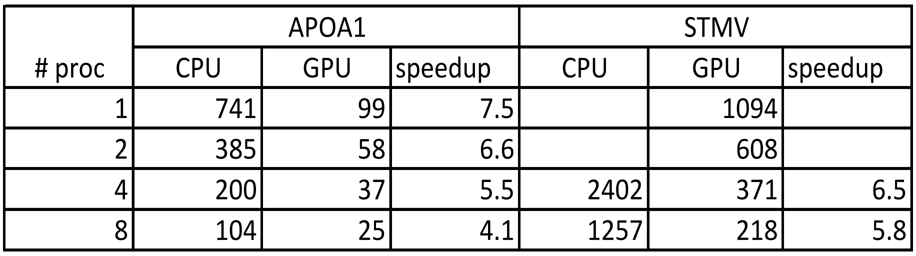


| | IOH x1 | IOH x2 |
|---|---|---|
| | --memory=pageable | |
| host to device: | 3672 | 3702/3716 |
| device to host: | 3023 | 2880/1807 |
| | --memory=pinned | |
| host to device: | 5499 | 5751/4709 |
| device to host: | 5291 | 3264/1873 |

## Application performance

NAMD[4] is known to run very efficiently on both traditional processors and GPUs. That is why we selected this molecular dynamics code as our test application. We run two tests: APOA1, the standard NAMD benchmark (model of a lipoprotein particle found in the bloodstream) and STMV (satellite tobacco mosaic virus). The former test comprises 92K atoms including lipid, protein and water. APOA1 benchmark is a moderately sized simulation suitable for long timescale studies. STMV test case is a larger benchmark comprising 1M atoms. Both tests were setup so that they do 500 simulation steps and compute energies only each 100th step. The latter is very important since energy calculations are done on the CPU in double precision and if the energies are compute every step no acceleration is observed. Presented below in the table are the best elapsed times of our runs.

**Table 6: Elapsed times in seconds of APOA1 and STMV benchmarks running on CPU and GPU.**



| | APOA1 | | | STMV | | |
|---|---|---|---|---|---|---|
| # proc | CPU | GPU | speedup | CPU | GPU | speedup |
| 1 | 741 | 99 | 7.5 | | 1094 | |
| 2 | 385 | 58 | 6.6 | | 608 | |
| 4 | 200 | 37 | 5.5 | 2402 | 371 | 6.5 |
| 8 | 104 | 25 | 4.1 | 1257 | 218 | 5.8 |

We observe an impressive speedup when running APOA1 and STMV benchmarks on GPUs. The speedup is slightly lower for APOA1 on large number of processes because it is a smaller test and start up time becomes an important factor. In order to get more adequate assessment of performance on the APOA1 test, the number of steps was increased from 500 to 5000 steps. These took 218 seconds of wall clock time using 8 GPUs which works out as 0.044 s/step (the program reported 0.041 s/step or 0.47 days/ns). This means 500 steps could be

[4] James C. Phillips, Rosemary Braun, Wei Wang, James Gumbart, Emad Tajkhorshid, Elizabeth Villa, Christophe Chipot, Robert D. Skeel, Laxmikant Kale, and Klaus Schulten. Scalable molecular dynamics with NAMD. Journal of Computational Chemistry, 26:1781-1802, 2005. http://www.ks.uiuc.edu/Research/namd/

done in 22 seconds and the speed up might have been 4.7 which is still lower than on STMV. For the record, the time per step for STMV benchmark as reported by NAMD was 0.37 s/step or 4.27 days/ns.

The results in Table 6 can be contrasted with the ones obtained on our Tesla S1070 cluster with Infiniband interconnect. The host processors used in the cluster are Intel Xeon E5540 which is about 10% quicker than E5520 used in the WS-Tesla x8 server. The elapsed time of running APOA1 test for 500 steps on four GPUs took 39 seconds on our cluster versus 37 seconds on the server which means the timing is roughly comparable. However STMV test took only 338 seconds on four GPUs of our cluster versus 371 seconds on WS-Tesla x8. The difference remained on eight GPUs (ie on two Infiniband connected servers versus fully loaded WS-Tesla x8): 180 seconds versus 218 seconds. Without GPUs, the benchmark times were always quicker on the cluster in line with the 10% clock difference. Thus the difference might be due to poorer bandwidth in dual IOH systems.

Another benchmark we were able to run was Linpack. The details about running Linpack on NVIDIA GPUs can be found elsewhere[5]. It suffices to say that for optimal performance we used matrix size N = 38400, block size NB = 1920 and environmental variables splitting the workload between CPU and GPU on DTRSM and DGEMM were set to 0.9. We achieved ~ 350 GFlops performance which corresponds to ~50% efficiency, ie half of the theoretical peak performance.

## Power measurements

Following our power measurements on the GPU cluster, we used the same methodology to measure power consumption of the WS-Tesla x8 server. Again we use an energy profile logger SP Max 512. Voltage was read directly from the mains by directly connecting it to the power outlet; current was measured through induction clamps. The device was placed inside of the rack together with a laptop through which the power logger was controlled remotely. The readings were taken in the increments of 1 second. SP Max 512 offers true RMS measurements with the accuracy of +/– 0.25% of the range plus CT (Current Transformer) error. A 10 A CT input lead was used which therefore determined the accuracy. We estimated that at 600W load our error should be around 1%.

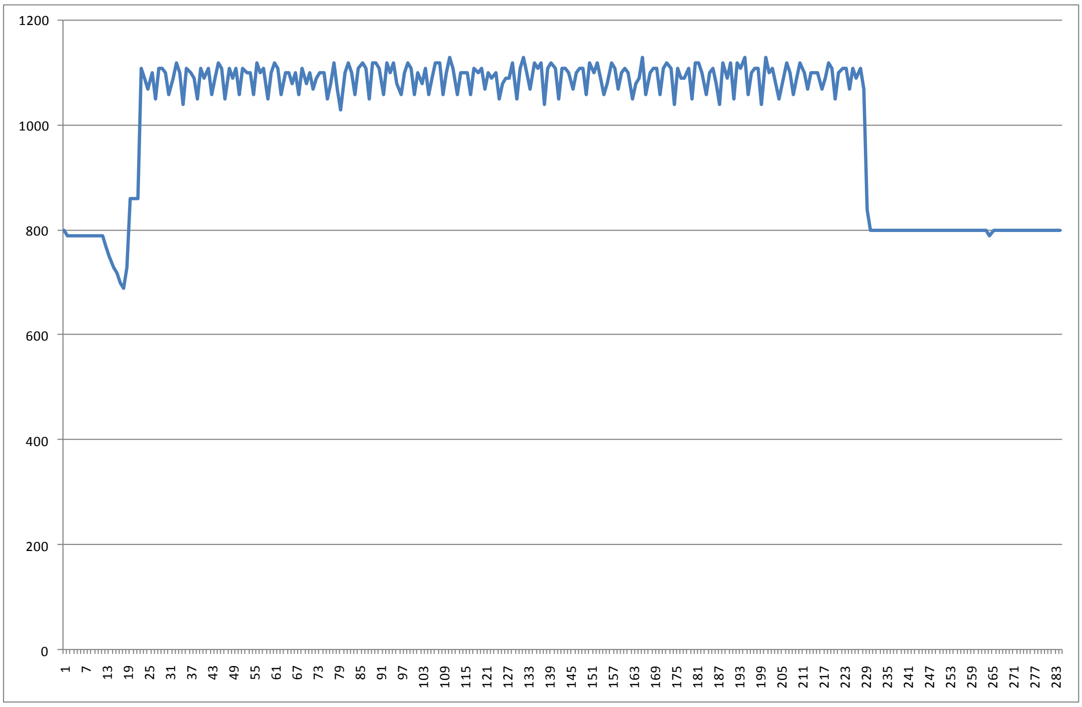


**Figure 3: Power profile of running APOA1 benchmark on WS-Tesla x8 server equipped with eight GPUs (note, that the number of steps was increased from 500 steps to 5000).**

The server consumed about 300 W in idle regime without GPUs and about 470 W when they were placed inside but not initialised. Once the GPUs were initialised we observed the same phenomenon as previously[1] when the power goes up and never comes back in the idle regime even though we were using the latest

[5] Massimiliano Fatica, “Accelerating linpack with CUDA on heterogenous clusters” in ACM International Conference Proceeding Series, vol. 383, pp. 46–51 (2009).

NVIDIA drivers. The power of the server with eight initialised GPUs nearly doubled to nearly 800 W. This seems like a lot of excessive power being wasted while the server is doing nothing. However this is still favourable compared to the idle power of our GPU cluster where the idle power of two Tesla servers and two hosts required to drive eight GPUs amounts to 1140 W.

While running NAMD, the power was typically slightly over 1 kW. It took about 350 kJ (~ 380 W average power) to accomplish 5000 steps of APOA1 benchmark on 8 Xeon cores and 235 kJ (~1080 W average power) on 8 GPUs which is ~ 1.5 energy saving on top of getting the results quicker. The power profile of running APOA1 test on 8 GPUs is presented in Figure 3.

For STMV, the difference in energy was nearly a factor of two. The average power while running the benchmark on eight GPUs was ~ 1060 W totalling in 230 kJ of energy. In comparison, in order to run the test on the host itself it required ~ 370 W and 465 kJ. Table 7 compares the present measurements with some of our previous results. The second column reports the performance and power metrics for two Infiniband connected Xeon E5520 servers. They finished STMV test in about 600 seconds and consumed slightly fewer than 600 W. This system took the maximum energy and the longest time. The same Xeon host and a Tesla S1070 server (second column) needs more energy but takes less time. Finally the WS-Tesla x8 server (forth column) runs nearly three times quicker than two Xeon servers and saves 50% energy.

**Table 7: Comparison of running STMV benchmark on different setups.**

| NAMD, STMV benchmark, 500 steps | 4x Xeon E5520 (2 servers) | 2x Xeon E5520 & S1070 | ratio to 4x Xeon | WS-tesla x8 server | ratio to 4x Xeon |
|---|---|---|---|---|---|
| Elapsed Time /sec | 599 | 338 | 1.8 | 218 | 2.7 |
| Avg Power /W | 581 | 740 | 0.8 | 1060 | 0.5 |
| Total Energy /kJ | 353 | 254 | 1.4 | 230 | 1.5 |

Unlike NAMD, Linpack requires high double precision performance which is not too impressive on M1060. Still the WS-Tesla x8 server improved on power performance efficiency scoring 0.3 GFlops/W which is better than 0.27 GFlops/W we obtained on our Tesla cluster and 0.24 GFlops/W on Intel Nehalem.

## Conclusion

It is quite clear that the aggregation of eight GPUs into a single host provides a clear advantage in terms of power efficiency. Running NAMD on the host takes about 370 W and that is the approximate amount of power we can save if we use it to drive eight GPUs instead of only four. We can also work out that average amount of power per GPU while running NAMD is about 86 W. This appears to be comparable to the power consumed by a powerful CPU.

The performance of the WS-Tesla x8 server is comparable to the one observed on our Tesla cluster. For example, the elapsed timing of APOA1 benchmark is rather close. Linpack also showed similar performance. However STMV benchmark was consistently slower by about 40 seconds. Because STMV test is larger and requires more memory, this may be related to the issue of poor bandwidth from GPU to CPU on dual IOH systems and further investigation is required. Unfortunately it is not clear yet when the bandwidth issue of dual IOH systems is going to be fixed. Nevertheless the use of servers which can accommodate eight GPUs inside 4U form factor presents a solution offering very high compute density and power efficiency. It is especially advantageous for applications which do not require high bandwidth to GPU devices and/or substantial part of the workload being computed on the CPU host.

## Acknowledgements

We are very grateful to Workstation Specialists and in particular to Mike Leach and Zia Hussan for letting us to evaluate their server.

04/06/2010 version 1.0

*I. N. Kozin and D. Cable, Distributed Computing Group, Computational Science and Engineering Department, STFC Daresbury Laboratory, Daresbury, Warrington, Cheshire, WA4 4AD, UK*