Evaluation of the Intel Westmere-EX server processor

Sverre Jarp, Alfio Lazzaro, Julien Leduc, Andrzej Nowak
CERN openlab, July 2011 – version 1.0

Executive Summary

One year after the arrival of the Intel Xeon 7500 systems ("*Nehalem-EX*"), CERN openlab is presenting a set of benchmark results obtained when running on the new Xeon E7-4870 Processors, representing the "*Westmere-EX*" family. A modern 4-socket, 40-core system is confronted with the previous generation of expandable ("EX") platforms, represented by a 4-socket, 32-core Intel Xeon X7560 based system – both being "top of the line" systems.

Benchmarking of modern processors is a very complex affair. One has to control (at least) the following features: processor frequency, overclocking via Turbo mode, the number of physical cores in use, the use of logical cores via Symmetric Multi-Threading (SMT), the cache sizes available, the configured memory topology, as well as the power configuration if throughput per watt is to be measured. As in previous activities, we have tried to do a good job of comparing like with like.

In a "top of the line" comparison based on the HEPSPEC06 benchmark, the "Westmere-EX" platform provides a 39% advantage over the "Nehalem-EX" one. In this comparison the power consumption remained constant, yielding an appreciable 39% of throughput per Watt improvement. Other benchmarks had similar scores, between 14% and 46%, depending on the configuration of SMT, Turbo and frequency scaling. The benefits of SMT remained constant at around 25% for throughput based applications, but were far lower (7%) for an OpenMP based latency bound program. In addition, Turbo mode efficiency was explored and compared in depth for the first time. While before Turbo mode was found to provide large boosts for low active core counts and no boosts for high active core counts, the situation with "Westmere-EX" was the opposite. Turbo mode seemed to provide small but consistent improvements across all active core counts.

Table of Contents

Introduction

Description of the processor

The codename for Intel's current Xeon 7000 Family microarchitecture is "Nehalem". The initial expandable "EX" flavor of Nehalem processors used the same 45 nm manufacturing process as the previous "Dunnington" processors. According to Intel's well-known "tick-tock" model, the following microarchitecture evolution is a "tick", which corresponds to a shrink of the manufacturing process. "Westmere" is the name given to this 32 nm die shrink of Nehalem. This shrink is now offered to the expandable server market with the Westmere-EX coming out along with a new naming scheme: the Intel Xeon Processor E7 family.

As for the Westmere-EP, the first new properties of the Xeon E7 family come directly from the shrink, allowing Intel to pack 25% more cores in each die (reaching 10 cores in total), and also 25% more of the shared L3 cache (reaching 30MB), compared to Xeon 7500 processors. Moreover, Intel managed to add these 2 cores, and the 6MB bigger L3 cache, while increasing the frequency and keeping power consumption within the same thermal design envelope. In this paper we compare the high end Westmere-EX E7-4870, which has 10 cores (20 threads) running at 2.40 GHz with a Thermal Design Power (TDP) of 130W, with the corresponding Nehalem-EX X7560, clocked at 2.27 GHz with the same 130W TDP and 8 cores (16 threads). These two processors have a fair number of features in common: they are both equipped with four Quick Path Interconnect (QPI) links at 6.4 GT/s, four Scalable Memory Interconnect (SMI) links and they both fit in the same LGA-1567 socket. Both support Simultaneous Multithreading (SMT), which allows two hardware threads to be scheduled on each core, and both support Turbo-boost mode, which allows the processor to increase its frequency when only few of the cores are in use, while maintaining power within the designed envelope. It should be noted, however, that the Turbo boost bins for the "Westmere-EX" part have been upgraded from 2.67GHz for the "Nehalem-EX" to 2.8GHz in the latter case.

Hardware configuration

As stated in the processor description, both processors fit in the same socket and have the same TDP. This allowed us to use exactly the same platform, after an extensive firmware upgrade session, for all our measurements, switching only the processors between the two series of experiments.

Our test system is a QSSC-S4R server jointly developed by Intel and Quanta. It provides four LGA-1567 sockets to connect up to four Xeon 7500/E7-4000 series processors and two Boxboro-EX IOH chipsets to handle the IO as illustrated on Figure 1.

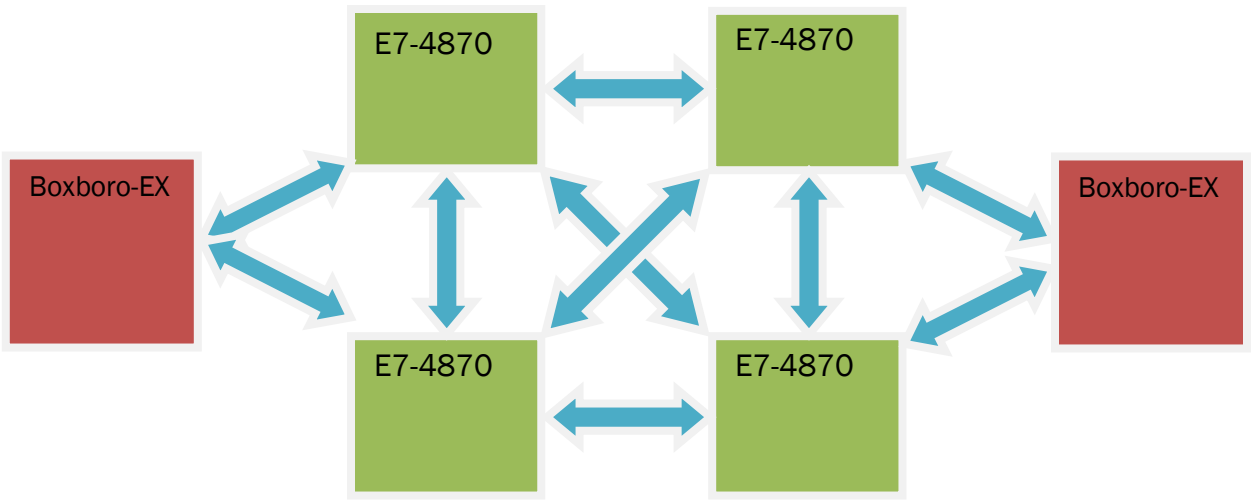

Figure 1: QPI topology of the full system

Up to 10 PCIe expansion boards can be plugged in this 4U server, and one of those slots is occupied by an Intel RS2BL080 SAS/SATA RAID card for front slot hard drive connectivity. This card supervises two 146.8GB SAS hard drives in a RAID0 configuration.

Since the memory configuration is crucial for good performance of this class of servers, and the Xeon E7-4000 series is no exception, it consists of 32 x 4 GB memory DIMMs. The system embeds eight memory boards; each of those boards is connected to two SMI links of a single processor. With this configuration all the SMI links of the four processors can be exploited. According to the previous processor description, each SMI link can handle four memory DIMMs through an SMB, but the available memory allowed only to accommodate two slots out of four for each SMI link. Thus the chosen last level topology balances the DIMMs on the SMBs, allowing the test system to maximize the memory bandwidth of the underlying processors configuration.

To be able to exploit all 128GB of RAM, on a system that is fully populated with 8 memory boards, the system is powered by three Power Supply Units (PSUs), out of a maximum of four.

Software configuration

The system was running 64-bit Scientific Linux CERN 5.6 (SLC5), based on Red Hat Enterprise Linux 5 (Server). The default SLC5 Linux kernel (version 2.6.18-238.9.1.el5) was used for all the measurements.

Standard energy measurements

Power meter

The standard energy measurement procedure is well-established in the CERN IT department for measuring the power consumption of any system that might be operated in the CERN Computing Center. Since this procedure was already thoroughly described in a previous openlab paper, “Evaluation of energy consumption and performance of Intel’s Nehalem architecture” [OPL09], it is now included as an appendix.

Results

The system is equipped with three power supplies, and the resulting total power consumption is the sum of the three power consumption measurements on those PSUs.

When conducting the tests without SMT, the system exposes 40 cores in total. Thus, according to the standard energy measurement procedure, the load stress consists of running 20 instances of CPUBurn along with 20 instances of LAPACK (using 6GB of memory each).

In the second phase, now with SMT enabled, the system was considered as an 80 core server, meaning that the Load stress test should be conducted by running 40 instances of CPUBurn along with 40 instances of LAPACK (using 3GB of memory each).

Active Power		*Idle*	*Load*	*Standard measurement*
128 GB	SMT-off	688 W	1211 W	1106 W
	SMT-on	688 W	1246 W	1134 W

Table 1: Total power consumption using three PSUs

As we can observe, these power consumption measurements reach some sizeable figures, even when the server is idle. The power consumption measurements for the ”Westmere-EX” system are the same as the ones for the “Nehalem-EX” based server. This sounds reasonable considering that those two processors are rated at the same 130W TDP, and that those servers are using exactly the same hardware components except for the processor SKUs.

Standard performance measurements

HEPSPEC2006

One of the important performance benchmarks in the IT industry is the SPEC[1] CPU2006 benchmark from the SPEC Corporation. This benchmark can be used to measure both individual CPU performance and the throughput rate of servers.

It is an industry standard benchmark suite, designed to stress a system's processor, the caches and the memory subsystem. The benchmark suite is based on real user applications, and the source code is commercially available. A High Energy Physics (HEP) working group has demonstrated good correlation between the SPEC results and High Energy Physics applications when using the C++ subset of the tests from the SPEC CPU2006 benchmark suite [WLCG09]. As a result the HEP community has decided to use the C++ subset of SPEC2006, "HEPSPEC06" rather than internal benchmarks because SPEC2006 is readily available, and its results can be directly generated by computer manufacturers to evaluate the performance of a system aimed at running HEP applications.

In this set of benchmarks it was compiled with GCC 4.1.2 in 64-bit mode, the standard compiler available with SLC5 and the performance measurements were carried out using with SMT disabled or enabled, and with Turbo mode on.

Since SPEC CPU2006 benchmark execution flow consists of serially launching several single threaded applications, several independent instances have to be launched simultaneously to evaluate the system scalability. This means that the HEPSPEC06 benchmark is indeed a rate measurement.

Number of processes	*HEPSPEC 06*
1	14.3
8	107
16	197
32	360
64	471

Table 2: HEPSPEC 06 measurements for X7560 ("Nehalem-EX") with Turbo mode disabled (not scaled)

Number of processes	*HEPSPEC 06*
1	16.4
8	123
20	281
40	501
80	654

Table 3: HEPSPEC 06 measurements for E7-4870 ("Westmere-EX") with Turbo mode disabled (not scaled)

[1] Standard Performance Evaluation Corporation (http://www.spec.org)

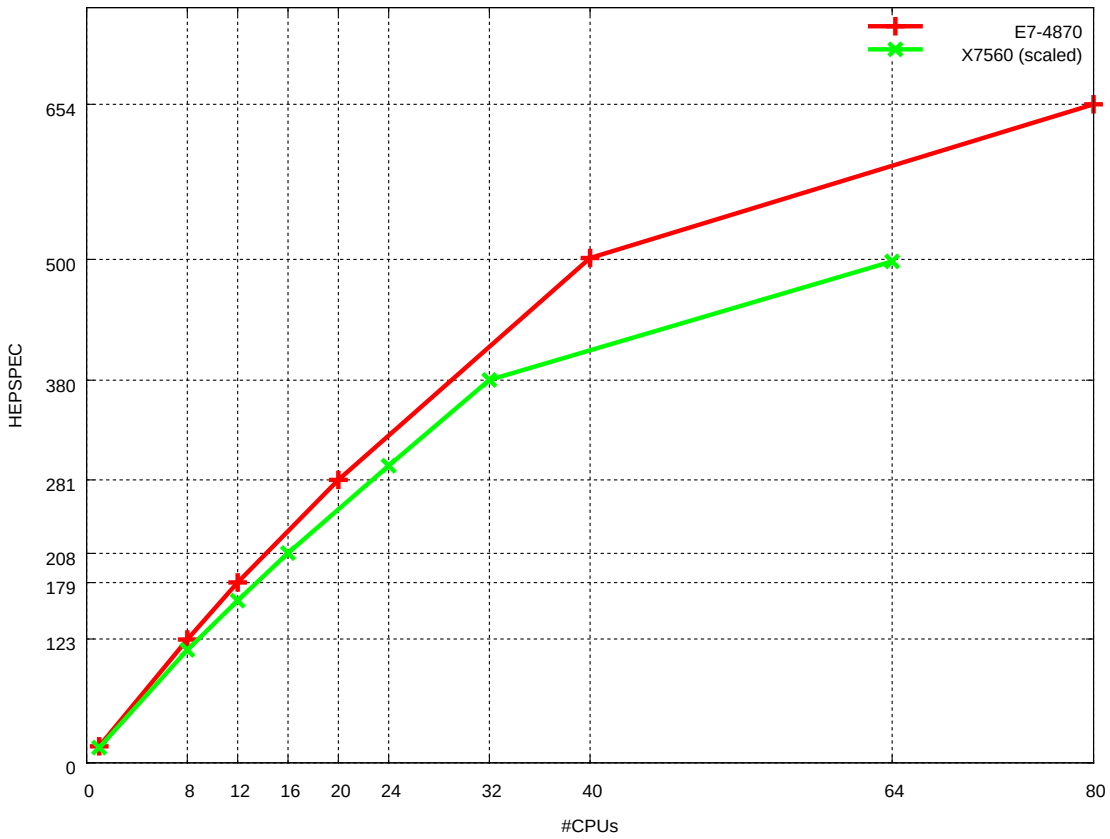

Figure 2: HEPSPEC2006 performance comparison E7-4870: SMT-on Turbo-off vs. X7560: SMT-on Turbo-off frequency scaled

X7560 based "Nehalem-EX" comparison (with Turbo disabled)

To compare the two systems, the X7560 results were frequency scaled, from the initial 2.27GHz clock rate up to 2.4GHz, to match the E7-4870 frequency. Both systems are four socket systems, aimed at the "expandable" server market.

As mentioned above concerning the core count, the older X7560 system counts a total of 32 cores (4x8 cores), but the E7-4870 offers 25% more, in the same 130W TDP per processor. SMT implementation on those two processors is similar, doubling the hyperthreaded core count for the two platforms. Therefore, SMT allows the X7560 CPU to count 64 hyperthreaded cores and the newer E7-4870 CPU reaches the colossal 80 hyperthreaded core figure.

As previously stated in the hardware description, the E7-4000 series processors are an evolution of the same micro architecture rather than a radical change from the previous 7500 series processors. This evolutionary progression is reflected on Figure 2: the scalability trends of the HEPSPEC plots are very similar. Of course, the almost linear initial portion stops at 32 cores, when the older contender has to start using hyperthreaded cores, but it underlines that, at the same frequency, an E7-4870 core provides an averaged 6% performance increase over an X7560 core.

If a direct comparison to the X7560 based platform is considered (SMT off), the new system yields 39% more throughput. Frequency scaled, the E7-4870 yields about 31% more throughput, the core performance improvements accounting for 6% and the core count increase for 25%.

X7560 based "Nehalem-EX" comparison Turbo enabled

The same frequency scaled comparison is then conducted with the Turbo feature enabled.

Here the evolutionary progression is depicted in Figure 3, similarly as in the Turbo disabled comparison. But here, surprisingly, from one to 16 cores, the two platforms offer approximately the same performance, and the "Westmere-EX" continues its close to linear progression until 40 cores, when all its physical cores are fully occupied. For the "Nehalem-EX" processor, after 16 cores, the HEPSPEC06 measurements are increasing at a slower pace, showing that the turbo effect is not able to sustain higher core occupancies.

Number of processes	*HEPSPEC 06*
1	16.2
8	120
16	227
24	299
32	372
64	478

Table 4: HEPSPEC 06 measurements for X7560 ("Nehalem-EX") with Turbo mode enabled

Number of processes	*HEPSPEC 06*
1	17.9
8	130
24	345
40	521
80	654

Table 5: HEPSPEC 06 measurements for E7-4870 ("Westmere-EX") with Turbo mode enabled

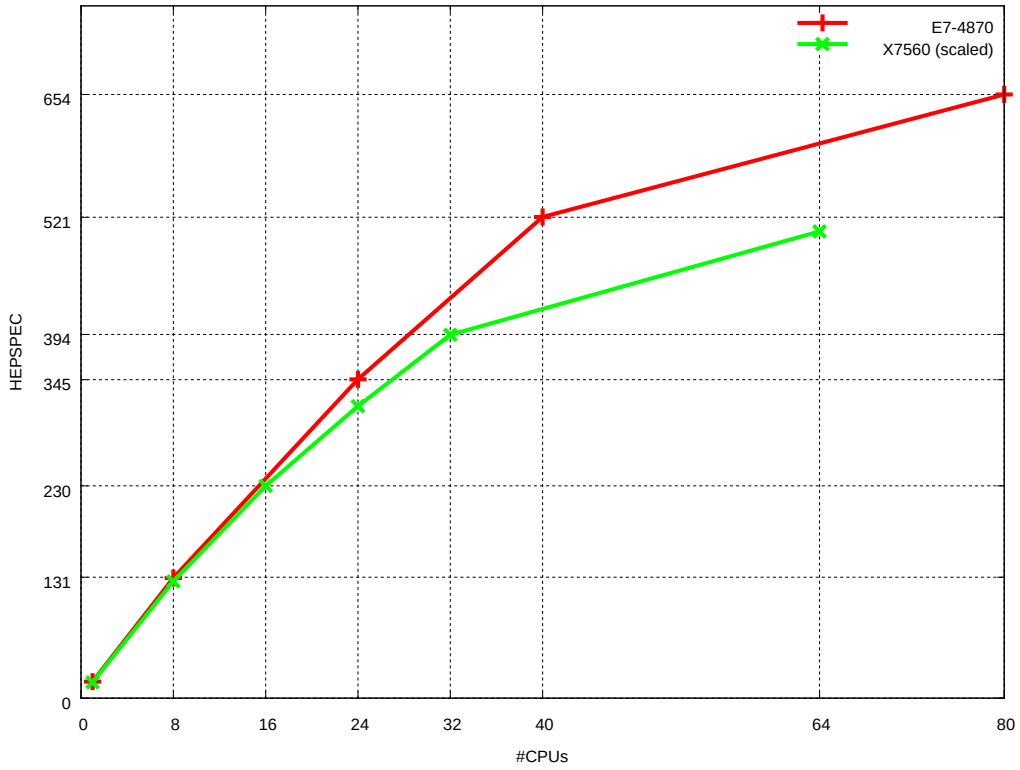

Figure 3: HEPSPEC2006 performance comparison E7-4870: SMT-on Turbo-on, X7560: SMT-on Turbo-on frequency scaled

As previously stated the newer core offers about 6% more performance than an X7560 ("Nehalem-EX") core, therefore from 1 to 16 cores the turbo gain is higher for the X7560, allowing it to close the performance gap with its younger peer.

Number of processes	*HEPSPEC 06 Turbo gain*
1	+13%
8	+13%
16	+10%
24	+7%
32	+3%

Table 6: HEPSPEC06 Turbo gains for X7560 (Nehalem-EX)

Number of processes	*HEPSPEC 06 Turbo gain*
1	+9%
8	+6%
24	+5%
40	+4%

Table 7: HEPSPEC06 Turbo gains for E7-4870 (Westmere-EX)

This phenomenon is confirmed in Table 6: using more than 4 cores per processor, the Turbo performance gain drops for the "Nehalem-EX" processor, while Table 7 shows a really linear decrease of the Turbo gain when inversely increasing the processor occupancies.

The low values for the Turbo gain with low core occupancies indicate that our "Westmere-EX" family CPUs may suffer from a firmware issue[2]: when performing a direct measurement of the core frequency stressing one core on a single socket, the system reports 2513MHz while on the processor specifications page [Intl4870], Intel indicates that the E7-4870 is able to reach 2800MHz. This Intel provided maximum frequency is more in line with our expectations. Indeed with 2.8GHz, the observed Turbo boost should be close to 17% for HEPSPEC runs on a single core.

If a direct comparison to the X7560 is considered, the new system allows for 37% more throughput turbo on. Frequency scaled the E7-4870 yields 30% more throughput.

SMT advantage

The SMT feature is present across all the lines of Intel processors since the Nehalem microarchitecture, providing interesting additional performance when the system has to execute more threads than its actual core count.

The gain produced by SMT can be deduced by comparing the HEPSPEC06 results for 40 and 80 processes for the "Westmere-EX", and for 32 and 64 processes for the "Nehalem-EX": in the case of both processors the SMT gain is the same at 26%. This additional gain shows the remarkable scalability potential of the four socket "Westmere-EX" system, increasing significantly its performance up to its maximum 80 SMT cores.

Additionally, the consistent SMT boost provided by the "Westmere-EX" system shows that the common Boxboro-EX platform provides enough bandwidth for the CPU upgrade, with respect to HEPSPEC06 multiprocessor performance.

Platform efficiency

Given that the platform power consumption and HEPSPEC06 measurements are available, the power efficiency in terms of HEPSPEC per Watt can be deduced.
Efficiency can be evaluated for two cases:

1. SMT off: taking the standard power measurement without SMT and the HEPSPEC measurement for 40 cores
2. SMT on: taking the standard power consumption with SMT and the HEPSPEC measurement for 80 cores

The efficiency comparison between the X7560 and the E7-4870 based system is straightforward, as the power consumption is the same for the two servers, the only difference is the HEPSPEC performance measurement.
Therefore, the same improvement as in HEPSPEC performance is observed on the efficiency side, the transition to the E7 family CPU allowing a 31% efficiency boost over the previous generation.

[2] In early test phases, several BIOS releases are often required to reach nominal performance

Another interesting point can be underlined when plotting the normalized platform efficiency histogram, using 2GB of memory per core.

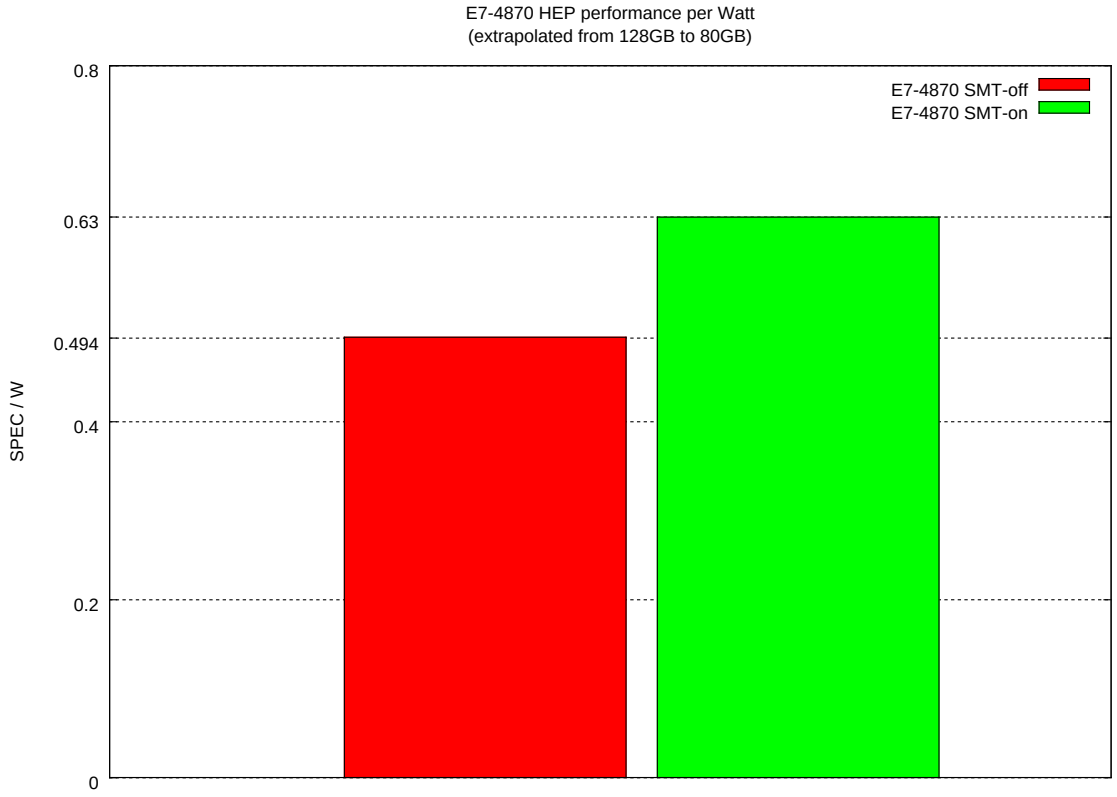

Figure 4: Efficiency of the E7-4870 platform with 3 PSUs

According to openlab previous evaluation of the Xeon X5670 [OPL10], a quad socket E7-4870 server with 3 PSUs offers similar efficiency as a dual socket Xeon X5670 server with a single PSU. The quad socket server is 2.4% less efficient that the dual socket server SMT-off, but 3.1% more efficient when SMT is enabled.

Multi-threaded Geant 4 prototype

Geant4 is one of the principal toolkits used in Large Hadron Collider (LHC) simulation. Its primary purpose is to simulate the passage of particles through matter. This type of simulation is a CPU-intensive part of a bigger pipeline used to process the events coming from the detectors. Since HEP has always been blessed with parallelism inherent in the processing model, it is natural to try to utilize modern multi-core systems by converging towards multi-threaded event processing. The Geant4 prototype discussed here is one of the key steps in that direction.

Based around Geant4, this suite has been updated to support multi-threading by two Northeastern University researchers: Xin Dong and Gene Cooperman. The example used in this case is “ParFullCMSmt”, a parallelized version of the “ParFullCMS” program, which represents a simulation close in properties to what the CERN CMS

experiment is using in production. Thus, this is an example of a real-world application in use at CERN.

One of the key metrics of a parallel application and the underlying parallel platform is scalability. The tests described in this chapter focus on the scalability of the multi-threaded Geant4 prototype, which is defined as throughput. In principle, a single-threaded process has a specified average time it takes to process 100 events. Thus we measure the influence of the number of processes (and implicitly the number of processing units engaged) on the processing time of 100 events. In an ideal case, as more processes with more work are added, one would expect the throughput to grow proportionally to the added resources, and so the processing time of 100 events would remain unchanged (per thread). Another key metric considered in this case is "efficiency", which is defined as the scaling of the software relative to the serial runtime, confronted with ideal scaling determined by the core count. In cases where multiple hardware threads are being used, perfect scaling is defined by the maximum core count of the system (40).

Technical test setup

The threads were pinned to the cores running them, and the throughput defining factor was the average time it takes to process one 300 GeV pi- event in a predefined geometry. The system was SMT-enabled, which means that the hardware threading feature was activated and used during the tests. Thus, if there were no more physical cores available, the jobs would be pinned to hardware threads, still maximizing the amount of physical cores used. In addition, the pinning system minimized the amount of sockets engaged. The tested framework was based on Geant4 4.9.2p01, CLHEP 2.0.4.2 and XercesC 2.8.0, and was compiled using the GCC 4.3.3 compiler.

Based on samples obtained from this and previous measurements, the measurement error for this benchmark is considered to be approximately +/- 0.5%.

Scalability testing

The application tested very well up to 40 physical cores. The efficiency under full physical core load was 100%, which corresponds to a perfect scaling factor of 40x. Detailed scalability data does not show efficiency penalties in scaling between 1 and 40 cores. Key scaling data points:

- 2x for 2 processes (101% efficiency)
- 8x for 8 processes (101% efficiency)
- 20x for 20 processes (101% efficiency)
- 32x for 32 processes (101% efficiency)

This data shows clearly that this benchmark exhibits excellent scalability on the tested system. One reason for the unusually high efficiency rating can possibly be the fact that some common data remains in the large caches of the CPU as different threads are trying to access it. Figure 4 demonstrates the gathered data while running on physical cores only, without the use of SMT. The simulation time is scaled on the left (blue) x-axis, while the efficiency is scaled on the right (green) x-axis.

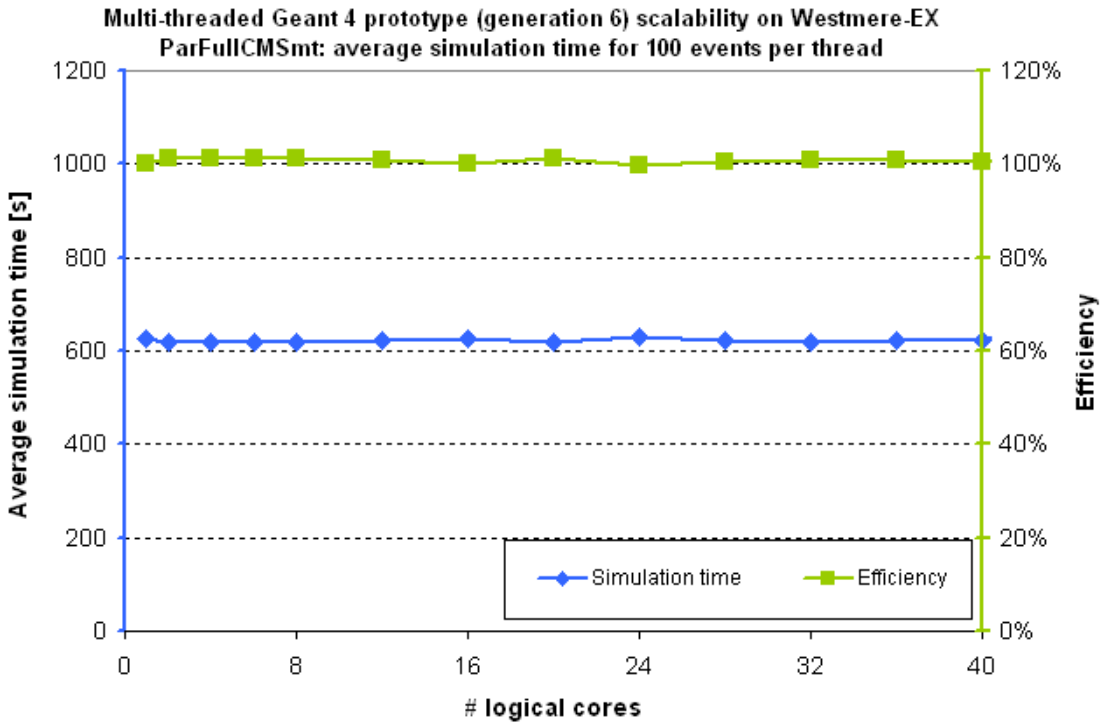

Figure 5: ParFullCMSmt scalability on Westmere-EX (without SMT)

In contrast, the following graph (Figure 5) shows the data for points between 1 and 80 threads. The efficiency curve grows past 40 cores to surpass 100%, since for thread counts higher than 40, expected scalability is fixed to 40x. Thus a final value of 123% indicates that the system loaded with 40 threads of the benchmark yields 23% more throughput than a perfectly scaled serial version on 40 physical cores.

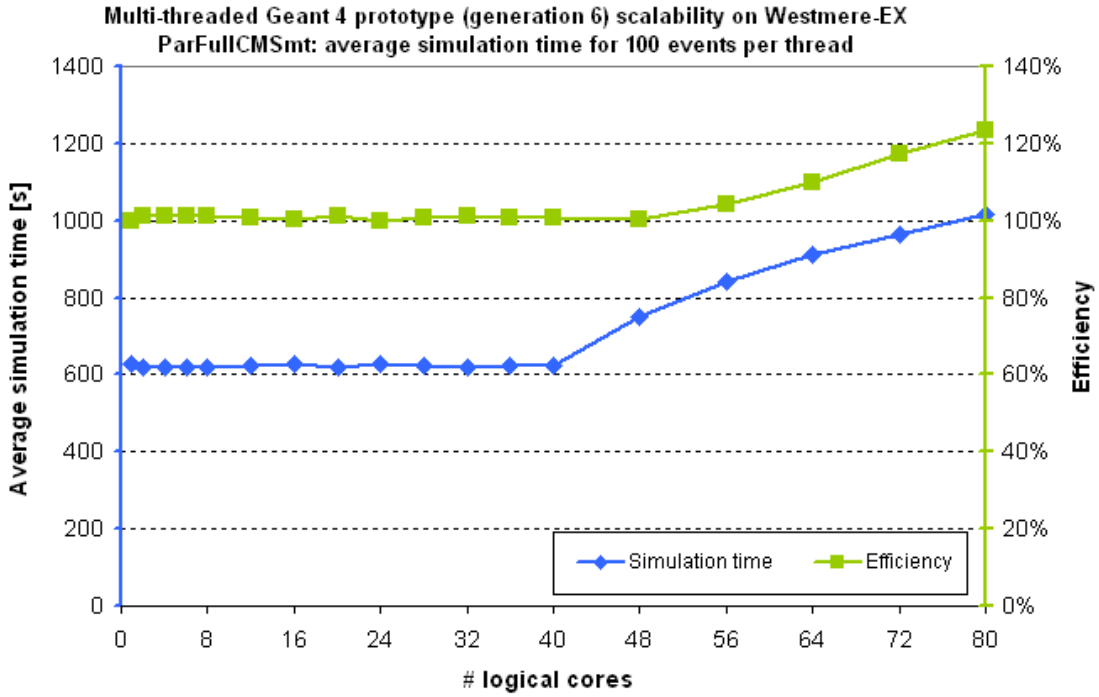

Figure 6: ParFullCMSmt scalability on Westmere-EX (with SMT)

Hyper-threading advantage

The advantage of running with SMT was:

- 4% with 56 hardware threads,
- 10% with 64 hardware threads,
- 17% with 72 hardware threads
- and finally 23% with all hardware threads engaged.

One should note that this extra performance is traded in for a penalty in memory usage, as the number of software threads is twice that of the case of 40 cores. Nevertheless, this advantage is in line with the figures that CERN openlab has seen for many other processors since the first Nehalem family (about 25%).

X7560 based "Nehalem-EX" comparison

For this benchmark, the frequency scaled figures for the X7560 and the E7-4870 are the same or within a very small percentage of each other. This indicates that no noticeable advances beneficial to this benchmark were made on the microarchitectural level. However, a more pragmatic comparison will confront the new top of the line 4-socket system (4 x E7-4870) and the previous top of the line 4-socket system (4 x X7560). In this comparison, the new "Westmere-EX" system wins significantly.

Let us first consider the additional theoretical performance that is to be obtained when replacing the old system with the new one. There are 25% more cores on each "Westmere-EX" chip, so that should give an ideal scaling factor of 1.25x. In addition,

the new "EX" parts are clocked at a higher frequency than their predecessors: 2.4 GHz instead of 2.27 GHz, within the same thermal envelope. That introduces an ideal scaling factor of 1.057x. Multiplying the two we obtain 32% of additional expected throughput, which is matched perfectly by the measured 32% increase in the throughput of the whole platform. These figures certify that the E7-4870 delivers expected scalability both on the frequency and on the core count fronts.

Parallel Maximum Likelihood fit

The "Parallel Maximum Likelihood fit" is a parallel benchmark from the data analysis domain.

The analysis of data collected by the Large Hadron Collider (LHC) physics experiments is the *tool* that allows experimentalists to claim discovery of new physics phenomena. Experiments collect data in form of independent events, an event being a set of measurements of physical quantities (*observables*) recorded in a limited span of time by the detectors. The events can be classified in different species, which are generally denoted as *signal*, for the events of interest, and *background*, for all other events. Signal events can be very rare and often represent a tiny fraction of the analyzed data [Phys08]. Hence, a huge quantity of events needs to be collected and the extraction of the signal over the background events represents a major challenge. For collecting large samples of data, experiments run with high acquisition rate and for long periods. In case of experiments on accelerators, two parameters are used to represent the instantaneous and total amount of data delivered by the accelerator to the experiments: the instantaneous luminosity and the integrated luminosity, respectively. Together with the efficiency of acquisition of the experiments, these parameters can be translated in number of events that need to be analyzed [Lee04]. For example, the LHC has reached a peak value for the instantaneous luminosity of about 1.3×10^{33} cm^{-2} s^{-1}, and the ATLAS and CMS experiments have collected at the time of writing more than 1 fb^{-1} (10^{39} cm^{-2}) each. This integrated luminosity corresponds to about 70×10^{12} recorded collisions. These numbers will increase during the remaining running phase of the LHC until 2012, expecting to reach the design instantaneous luminosity of 10^{34} cm^{-2} s^{-1} with 4 times more integrated luminosity.

Data analysts can run their analyses on the partial recorded sample to check whether they have already collected sufficient data to claim a discovery or in any case to improve the previous search measurements, setting more stringent limits on discoveries. Note that a conservative approach is adopted, so that the complexity of the analyses will increase with the quantity of data. In this "game" it becomes crucial to have the best discrimination between signal and background events, so that less events are needed for reaching the possible discovery. This is the goal of data analysis techniques. Introducing the concept of probability per each event to be signal or background, several techniques used in the HEP community are based on the evaluation of the likelihood function [Cow98]. This function is the sum of different terms (related to the probabilities) calculated for each event of the given data sample. In HEP the RooFit package [Roo11], which is part of the ROOT software framework

developed at CERN, provides classes to define and evaluate the likelihood function. Since each event is independent from the others, it is possible to parallelize the calculation of the likelihood function by performing the calculation of the terms in the sum in parallel. Therefore a parallel prototype of RooFit has been developed using a shared memory paradigm implemented with OpenMP [Che11]. Furthermore, several optimizations of the code have been used, including auto-vectorization by the Intel C++ compiler. We should note that all calculations are done in double precision.

The following tests are based on a maximum likelihood fit analysis [Cow98], where the likelihood function is evaluated in parallel with the RooFit prototype. The maximum likelihood technique allows for estimating the values of some parameters on a given data sample by means of a minimization process. The package used for minimization is Minuit2 [Min72], which is the most common package used for optimization of a function in the HEP community. The main algorithm in this package, MIGRAD, searches for the minimum of the likelihood function using gradient information [Num07], performing several iterations before reaching convergence. This procedure requires several evaluations of the likelihood function. Depending on the number of iterations during the minimization, the number of free parameters, the complexity of the function, and the number of events and the number of observables, the time spent for the fit can vary from a few seconds to several days. The tests presented here are based on an analysis performed at the BaBar experiment (an experiment which was located at the SLAC National Accelerator Laboratory, California) [Aub09]. The data sample consists of 500,000 generated events with 3 observables. The model is composed of 11 Gaussian and 5 Polynomial distributions, divided into 5 species (1 signal and 4 backgrounds). Thus, this is an example of a real-world application in use in the HEP community.

In the following tests the execution time spent by the application while performing the fit with parallel execution using a certain number of threads was considered. They are optimized for latency (strong scaling). Note that time for event preparation and general initialization is not considered (it is less than 1 second). Since the final summation is executed in parallel, this can lead to slightly different results in the minimization depending on the number of threads used in the parallelization. This problem was solved by implementing a parallel reduction based on a particular summation algorithm [Red01]. This guarantees that the results are the same in all configurations. The workload is statically split on the several threads, without any significant increase in the memory footprint (data and results are shared). For the considered likelihood function about 40% of the execution time is spent in evaluating exponentials. The sequential portion, which cannot be parallelized, is 1% of the total sequential time, mainly for evaluating the normalization integrals and the calculation of an additional term of the likelihood function (the extended term). However, OpenMP synchronization overheads and access to data for increasing number of parallel threads introduce a significant penalty to the scalability. In particular the latter can be reduced on systems with larger available cache size (L3 cache). Future developments on the implementation aim to reduce these effects.

Technical setup

RooFit prototype v3.2 and Minuit2 from ROOT v5.28 are used in the tests. Code was compiled with the Intel C++ compiler version 12.0.2 20110112, supplying the following options:

"`-O2 -m64 -fPIC -funroll-loops -finline-functions -msse3 -ip -openmp`".

The compiler is able to successfully apply auto-vectorization in main loops. Tests are run on the discussed "Westmere-EX" system and for comparison on the X7560 based "Nehalem-EX". The threads are pinned to the cores running them. The systems are SMT-enabled, which means that the hardware threading feature was activated and used during the tests. Thus, if there are no more physical cores available, the jobs would be pinned to hardware threads, still maximizing the amount of physical cores used. In addition, the pinning system maximized the amount of sockets engaged. Running the sequential application on the "Westmere-EX" with Turbo OFF takes about 43 minutes.

Results

The first test presented is a comparison between Turbo mode ON and OFF. These results are shown on the plots on Figure 6. It is possible to see that on the "Westmere-EX" Turbo has less impact on the performance (around 4%-5% for 1 to 40 threads) with respect to the "Nehelem-EX" results (18% for 1 thread, with a constant decrease with the number of threads, up to no speed-up with 32 threads). We should note that core affinity has been used to maximize the number of sockets involved, i.e. N threads means N/4 thread per socket. Naively, the major impact of Turbo is expected up to 4 cores, i.e. one thread per socket. However, Turbo has a constant effect on the "Westmere-EX" when varying the number of threads, while on the "Nehalen-EX" the speed-up goes rapidly down: 18%, 14%, 13%, 7% for 1, 4, 8, 16 threads, respectively, and no significant effect over 16 threads. There is no effect when using SMT threads for both systems. This result can be correlated with the nominal frequency of the two systems. It turns out that for a single thread execution, the "Westmere-EX" is 10% faster with Turbo OFF (4% frequency scaled), but it is 3% slower with Turbo ON. This means that a single core on the "Westmere-EX" gives better performance with Turbo OFF, while it is the opposite for Turbo ON. A possible explanation is that the two systems reach the same frequency with Turbo ON. Indeed, the same behavior occurs also with 2 and 4 threads, which is reasonable since there is a maximum of 1 thread per socket. Increasing the number of threads and giving the fact that the effect of Turbo decreases on "Nehalem-EX", the "Westmere-EX" becomes faster already with 8 threads. All results of this comparison are shown in Table 8. At this point it should be reiterated that one of the main limitations to the scalability of the application is accessing data in memory. The fact that the "Westmere-EX" has more L3 cache (30 MB) with respect to the "Nehalem-EX" (24 MB) improves the overall performance of the application for higher number of threads (see the results in table).

Number of threads	*Turbo OFF*	*Turbo ON*
1	+10% (+4%)	−3% (−8%)
2	+6% (+1%)	−5% (−10%)
4	+9% (+3%)	−1% (−7%)
8	+14% (+8%)	+7% (+1%)
12	+22% (+15%)	+16% (+10%)
16	+24% (+18%)	+21% (+14%)
32	+26% (+19%)	+27% (+20%)

Table 8: Comparison of the "Westmere-EX" and "Nehalem-EX" execution times, for Turbo OFF and Turbo ON. A positive (negative) number refers to faster (slower) "Westmere-EX" execution. The numbers in parentheses are frequency scaled (considering the nominal frequencies of the two systems)

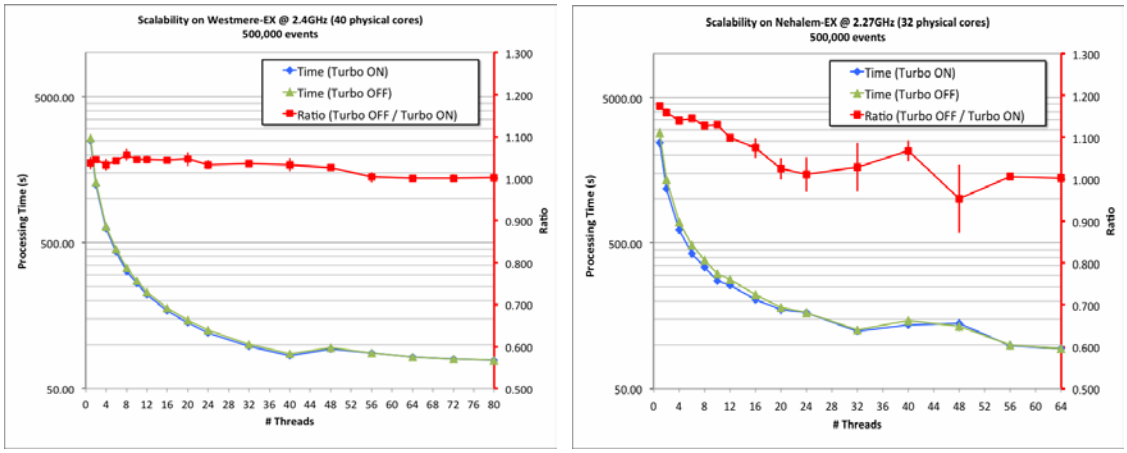

Figure 7: Comparison of the execution time with Turbo ON and Turbo OFF ("Westmere-EX" on the left, "Nehalem-EX" on the right)

From the point of view of the scalability of the application, Turbo can speed-up the execution of the portion that is not parallelizable, with a consequent improvement on the scalability. This means that on the "Westmere-EX" a better scalability for higher number of threads is expected when Turbo is ON. So the speed-up is calculated with Turbo ON taking as reference the sequential execution with Turbo OFF. The results are shown on Figure 7. With 32 threads, "Westmere-EX" reaches a speed-up of 26.7x, which becomes 30.9x and 33.1x for 40 and 80 threads, respectively. In the meantime, "Nehalem-EX" has 23.2x for 32 threads and 30.4x for 64 threads. Also in this case a key factor that improves the performance on the "Westmere-EX" is the bigger cache available on the system when using 32 threads, while there is no expected improvement when the two systems are fully loaded. Using these numbers the benefit of using SMT can be extracted, which is +7% for the "Westmere-EX" (80 versus 40 threads) and +31% for the Nehalem-EX (64 versus 32 threads). Also in this case we should consider two facts:

- Running more threads to fully load the Westmere-EX introduces more OpenMP overheads then loading the Nehalem-EX with less threads;
- Westmere-EX with 40 threads has +3% when running with Turbo ON, with no speed-up at 80 threads, while the Nehalem-EX has a negligible speed-up because of the Turbo ON for both 32 and 62 threads.

Considering Turbo OFF, the speed-up of the SMT on the Westmere-EX becomes +10%.

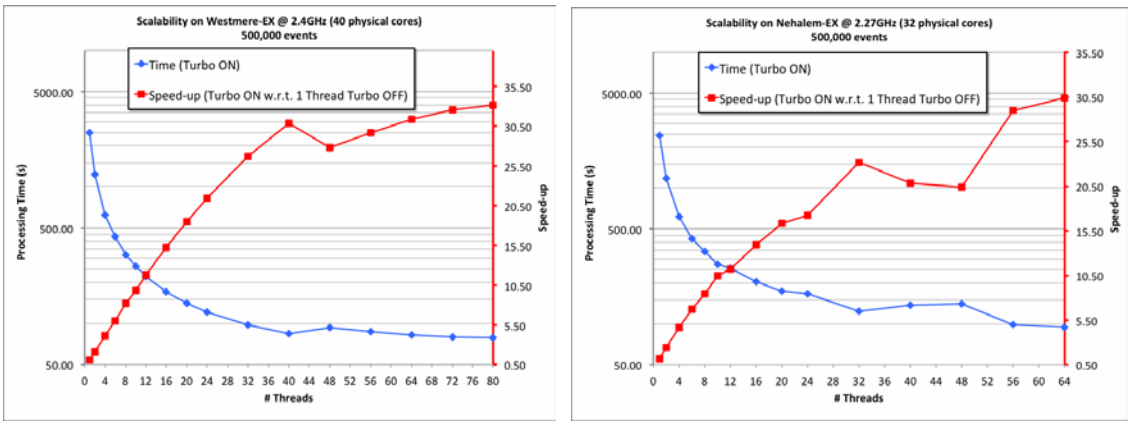

Figure 8: Execution time and speed-up, considering Turbo ON with respect to 1 single thread execution with Turbo OFF ("Westmere-EX" on the left, "Nehalem-EX" on the right)

Finally, Table 9 shows a comparison of the performance of the two systems when they are fully loaded. Two main conclusions can be drawn from these numbers:

- Without using SMT, the "Westmere-EX" has an excellent performance with respect to "Nehalem-EX". Assuming +25% due to more available cores and the higher frequency, the rest can be reasonably attributed to the bigger L3 cache size.
- With SMT the consideration of the previous point is still valid, but then we are limited by the OpenMP overheads due to running 16 more threads. With 80 threads OpenMP calls take about 50% of the execution time. This is a clear limitation of the application for the specific case under consideration when running on such a large number of threads.

Number of threads *Nehalem-EX vs. Westmere-EX*	*Turbo OFF*	*Turbo ON*
32 vs. 40	+46% (+38%)	+47% (+39%)
64 vs. 80	+20% (+14%)	+20% (+14%)

Table 9: Comparison of the "Westmere-EX" and "Nehalem-EX" execution times, for Turbo ON and Turbo OFF. The comparison is done with respect to "Westmere-EX" results, so a positive number means that "Westmere-EX" is faster. The numbers in parentheses are frequency scaled (considering the nominal frequencies of the two systems)

Conclusions and summary

Overall, the new "Westmere-EX" system provides performance that is much improved over its "Nehalem-EX" predecessor, but also one that is less predictable with respect to theoretical calculations.

Core increase and architectural changes

The new "Westmere-EX" system provides a substantial core count increase for yet another generation of Xeon processors. The additional 25% of cores is reflected in the performance measurements. The architectural changes were minor, but the HEPSPEC06 benchmark still benefited from an additional 8.5% for a single core.

Hyper Threading (SMT)

In throughput based benchmarks (HEPSPEC06 and the Multi-threaded Geant 4 prototype), SMT provided a gain comparable with previous observations, on the level of 23-26%. In the case of parallel minimization, which is a latency bound application, the benefit was only 7%, likely due to parallel overheads of the implementation.

Turbo mode

In comparison with the "Nehalem-EX", the Turbo implementation in the tested system seemed to provide boosts that were rather low for low core counts and falling short of official predictions. The reason for this behavior was not clear at the time of writing, but it is speculated that future firmware updates could stabilize this situation. Despite of that, there was noticeable benefit of turbo mode even with all cores loaded, which lead to noticeably higher scores on the HEPSPEC06 and parallel minimization benchmarks.

Overall platform performance

When comparing the systems head to head without any kind of scaling (best to best), the new "Westmere-EX" platform provided 39% more throughput in the HEPSPEC06 benchmark – within the same power envelope. Interestingly enough, the "Westmere-EP" system achieved exactly the same 39% advantage in our previous measurements comparing it to a "Nehalem-EP" predecessor (with different relative core counts). In other benchmarks, the advantage was between 20% and 47%, depending on the Turbo/SMT configuration.

Frequency scaled, the new platform provides between 14% and 39% more throughput, again depending on the Turbo/SMT configuration.

Since the power consumption has not changed, one can reasonable assume that the performance per Watt increase is comparable with the performance increase itself – 39% is a very respectable figure.

References

Intl4870	Intel® Xeon® Processor E7-4870 Specifications: http://ark.intel.com/Product.aspx?id=53579
WLCG09	Multiple authors: *Transition to a new CPU benchmarking unit for the WLCG*, (2009)
OPL09	A. Busch, J. Leduc: "Evaluation of energy consumption and performance of Intel's Nehalem architecture", CERN openlab (2009)
OPL10	S. Jarp, A. Lazzaro, J. Leduc, A. Nowak: "Evaluation of the Intel Westmere-EP server processor", CERN openlab (2010)
Phys08	G. Kane, A. Pierce, *Perspectives on LHC Physics*, World Scientific (2008)
Lee04	S. Y. Lee, *Accelerator Physics*, World Scientific (2004)
Cow98	G. Cowan, *Statistical Data Analysis*, Clarendon Press, Oxford (1998)
Roo11	W. Verkerke and D. Kirkby, *The RooFit Toolkit for Data Modeling*, http://root.cern.ch/drupal/content/roofit
Che11	S. Jarp *et al.*, *Parallelization of maximum likelihood fits with OpenMP and CUDA*, CERN-IT-2011-009 (2011)
Min72	F. James, *MINUIT - Function Minimization and Error Analysis*, CERN Program Library Long Writeup D506 (1972)
Num07	W. H. Press and S. A. Teukolsky and W. T. Vetterling and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press (2007)
Aub09	B. Aubert *et al.*, BaBar Collaboration, Phys. Rev. D 80, 112002 (2009)
Red01	Y. He and C. H. Q. Ding, *Using Accurate Arithmetic to Improve Numerical Reproducibility and Stability in Parallel Applications*, The Journal of Supercomputing", 18, 259-277 (2001)

Appendix A - standard energy measurement procedure

Measurement equipment

For the measurements a high precision power analyzer with several channels is required, since it must be able to measure the power consumption of any system from a simple UP system, with a single power supply unit (PSU) to a large server equipped with 4 PSUs.

To this extend a ZES-Zimmer LMG450 power analyzer is used. It allows the measurement of common power electronics. It has an accuracy of 0.1% and allows the measurement of four channels at the same time, and thanks to its convenient

RS232 port, it can be linked to a PC to sample the values on the 4 channels, as shown on Figure 8.

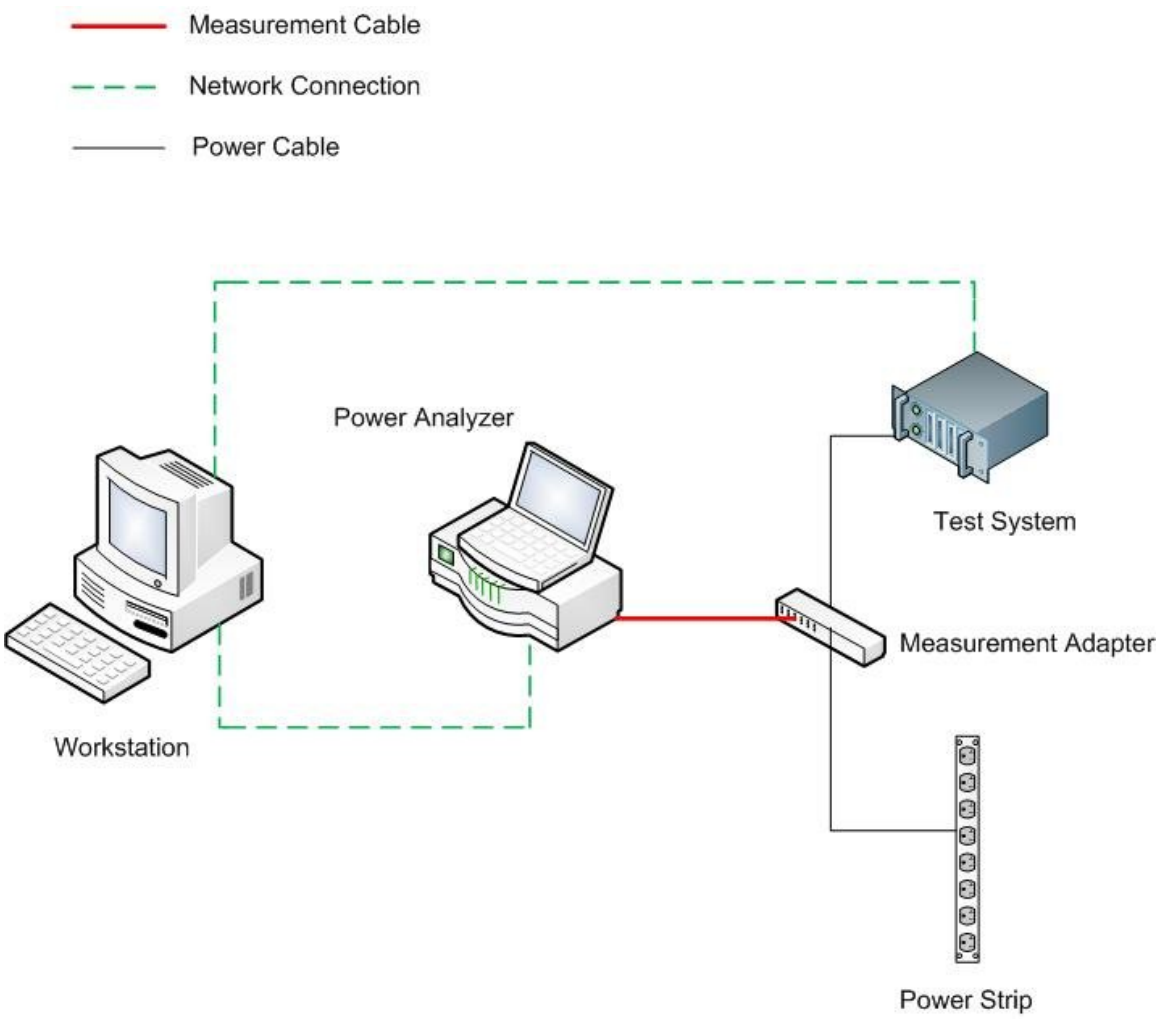

Figure 9: Power test setup

Three units are measured for each channel:

- *Active Power (P)*: The active power is also often called "real" power and is measured in Watts (W). If the active power is measured over time the energy in kilowatt hours is determined.
- *Apparent Power (S)*: Apparent power is the product of voltage (in volts) and current (in amperes) in the loop. This part describes the consumption from the electrical circuit. It is measured in VA.
- *Power Factor:* In our case, the power factor means the efficiency of the power supply. The closer the power factor is to one, the better is the efficiency: powerfactor = active power/apparent power

If the system includes several PSUs the Active Power must be summed on all the channels in use to compute the total Active Power of the system, for the two stress conditions.

LAPACK/CPUBurn

Those two tools are used to stress the evaluated systems, providing a reproducible load for any type of server:

1. *LAPACK* (Linear Algebra PACKage) was written in Fortran90 and is used to load both the memory system and the CPU. It provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems. The memory consumption depends on the size of the generated matrices and is easy to adapt to fit the needs.
2. *CPUBurn* was originally written as a tool for overclockers, so that they can stress the overclocked CPUs, and check if they are stable. It can report if an error occurs while the benchmark is running. It runs Floating Point Unit (FPU) intensive operations to get the CPUs under full load, allowing the highest power consumption to be measured from the CPU.

Standard energy measurement

The standard energy measurement is a combination of the Active power measured measured under two different stress conditions:

1. *Idle*: the system is booted with the Operating System and it does nothing.
2. *Load*: the system is running CPUBURN on half of the cores, and LAPACK on all the other cores, using all the installed memory.

An example to stress a system counting 8 cores and 12 GB of memory for the Load condition, would imply to run 4 instances of CPUBurn along with 4 instances of LAPACK each consuming 3 GB of memory.

According to that, the standard energy measurement is a mix of the active power under Idle condition, accounting for 20%, and the active power under Load condition, accounting for 80%.