Evaluation of the Intel Nehalem-EX server processor

Sverre Jarp, Alfio Lazzaro, Julien Leduc, Andrzej Nowak
CERN openlab, May 2010 – version 1.1

Executive Summary

In this paper we report on a set of benchmark results recently obtained by the CERN openlab by comparing the 4-socket, 32-core Intel Xeon X7560 server with the previous generation 4-socket server, based on the Xeon X7460 processor. The Xeon X7560 processor represents a major change in many respects, especially the memory sub-system, so it was important to make multiple comparisons. In most benchmarks the two 4-socket servers were compared. It should be underlined that both servers represent the “top of the line” in terms of frequency. However, in some cases, it was important to compare systems that integrated the latest processor features, such as QPI links, Symmetric multithreading and over-clocking via Turbo mode, and in such situations the X7560 server was compared to a dual socket L5520 based system with an identical frequency of 2.26 GHz.

Before summarizing the results we must stress the fact that 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 MultiThreading (SMT), the cache sizes available, the configured memory topology, as well as the power configuration if throughput per watt is to be measured. We have tried to do a good job of comparing like with like.

In summary, we saw a broad range of results. Our variant of the SPEC benchmark rate, “HEPSPEC”, gave a stunning 3x overall improvement on the new server, thanks to good scaling with the 32 cores and a 26% additional gain when enabling SMT. In-house data analysis and simulation benchmarks showed throughput increases in the range of 11 to 60%. Oracle database tests will follow. Finally it should be mentioned

that the 4-socket server can be equipped with 32 memory cards (DIMMs) which correspond to 512 GB total memory. This is a very impressive amount of memory that also comes with a very significant thermal load.

Table of Contents

Introduction

Description of the processor

Intel's current Intel64 "Nehalem" microarchitecture is widely deployed over the available Intel processor line. For dual processor (DP) servers, Intel proposes the Xeon 5500 series. For desktop systems, it offers the Core i7 and Core i5 series, and several low power parts gained momentum in the laptop market. The "Nehalem" microarchitecture offers several improvements to all these platforms compared to the previous "Core" microarchitecture, which in our tests is represented by the Intel Xeon X7460 "Dunnington" processor.

One of the most awaited improvements, especially considering DP systems, is the new memory management subsystem: each processor incorporates the memory controllers that handle all the memory directly attached. This means that, in the case of a well balanced DP system, each physical processor directly manages half of the server's total memory, and accesses the other half indirectly through a request to the other processor. Those communications between the two sockets are possible thanks to the new Quick Path Interconnect (QPI) links. QPI technology replaces the Front Side Bus, by offering a point to point connection between a CPU and a chipset or between two CPUs.

This new approach radically transforms the memory subsystem from a uniform architecture, or uniform memory access (UMA), to a non uniform topology, or non uniform memory access (NUMA). Now when a processor accesses the memory managed by the remote CPU, an additional inter-processor communication hop is necessary – at a cost. The NUMA aspects of the Nehalem microarchitecture are interesting, since historically NUMA has been associated with systems counting a large number of CPUs. This means that the Nehalem microarchitecture has a considerable potential for the high end market of scalable servers counting several processors.

In a single processor system, a single QPI link on the CPU is sufficient, since all the communication takes place between the processor and the chipset. In a DP system, the scheme is different since either processor requires at least one QPI link to the other processor and one with the chipset, so two QPI links are the minimum for a DP capable processor.

For the platforms where the processor count is higher, the number of connections must also be increased to accommodate the connection to the chipset and to its siblings. For instance, three QPI links per processor allow a four socket "glueless" configuration with all the memory within two hops. But Intel went further and equipped the Nehalem-EX with four QPI links and as a consequence all remote memory is now available within one hop for such a configuration.

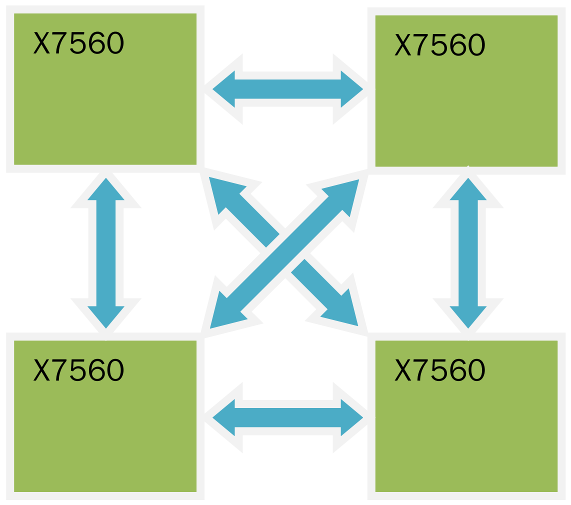

Figure 1: QPI links topology in a four sockets system

Figure 1 illustrates the layout of the four processors in a quad socket system, and the QPI connections from processor to processor. One should not forget that each processor deploys a fourth QPI link to the I/O subsystem. This layout is a fully connected mesh topology.

Not only does the Xeon 7500 series allow for a higher processor count, but also internally, each processor doubles its core count from the four cores of the Xeon 5500 series to eight[1]. This impressive number of cores is backed by a better performing memory subsystem. As with the existing Xeon 5500 series parts, each processor has a dedicated memory controller that provides access to its attached memory. In the case of the Xeon 7500 series, each processor includes four Scalable Memory Interconnect (SMI), and each of those SMI links gives access to four DDR3 DIMMs through its Scalable Memory Buffer (SMB). This memory structure enables each processor to have up to 16 attached DIMMs, allowing the four socket system to scale up to 64 DIMMs. This allows for up to 512GB of memory, when using 8GB DIMMs.

This impressive memory capacity coupled with the efficient NUMA architecture and the high core count of the Xeon 7500 series processors should transcend the scalable systems it equips to reach new levels of performance, compared to the previous generation Xeon 7400 series, dubbed Dunnington.

Hardware configuration

The processor evaluated in this paper is an X7560 running at 2.27 GHz. Our test system is a QSSC-S4R server jointly developed by Intel and Quanta. It provides four LGA-1366 sockets to connect up to four Xeon 7500 series processors and two Boxboro-EX IOH chipsets to handle the IO as illustrated on Figure 2.

[1] Later in the paper we compare to the Xeon 7460 which has six cores per processor.

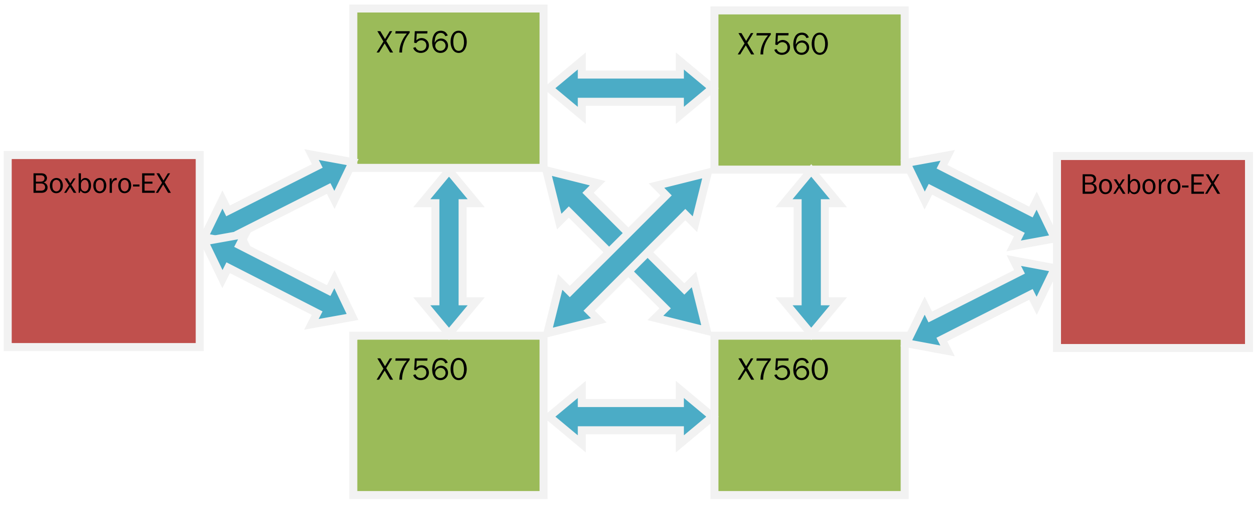

Figure 2: QPI topology of the full system

Up to 10 PCIe expansion boards can be plugged in this 4U server, and one of those slots is occupied by a 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 7500 series is no exception, it consists of 32 x 4 GB memory DIMMs. The system embeds eight emerald ridge 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 emerald ridge boards, the chassis must count at least three Power Supply Units (PSUs), with a maximum of four, to be able to sustain high loads.

Software configuration

The system was running 64-bit Scientific Linux CERN 5.4 (SLC5), based on Red Hat Enterprise Linux 5 (Server). The default SLC5 Linux kernel (version 2.6.18-164.15.1.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”, it is now included as an appendix.

Results

The system is equipped with three power supplies, offering no redundancy. As already mentioned, the described configuration requires a minimum of three PSUs to sustain the load associated to the benchmarks.

When conducting the tests without SMT, the system appears as having 32 cores in total. Thus, according to the standard energy measurement procedure, the load stress consists of running 16 instances of CPUBurn along with 16 instances of LAPACK (using 8GB of memory each).

In the second phase, now with SMT enabled, the system was considered as a 64 core server, meaning that the Load stress test should be conducted by running 32 instances of CPUBurn along with 32 instances of LAPACK (using 4GB of memory each).

Active Power		*Idle*	*Load*	*Standard measurement*
128 GB	SMT-off	715 W	1209 W	1110 W
	SMT-on	715 W	1243 W	1137 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. An additional series of measurements realized as the server was equipped with 64GB of memory and only four emerald ridge cards, showed that each emerald ridge card populated with eight 4GB memory DIMMs consumes 40W when idle and 60W when loaded, which in Standard power consumption terms is 56W. This implies that the Standard power consumption of all the memory boards and their associated DIMMs is 448W. The conclusion of those measurements shows that the memory subsystem draws almost 40% of the total power consumed by the system. The power-performance of the Dunnington system was not evaluated in this context.

Standard performance measurements

HEPSPEC2006

One of the important performance benchmarks in the IT industry is the SPEC[2] 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	*15.5*
8	124
16	227
32	379
64	478

Table 2: HEPSPEC 06 measurements for X7560 with Turbo-mode switched off

Number of processes	*HEPSPEC 06*
1	13
8	91
12	119
16	139
24	157

Table 3: HEPSPEC 06 measurements for X7460

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

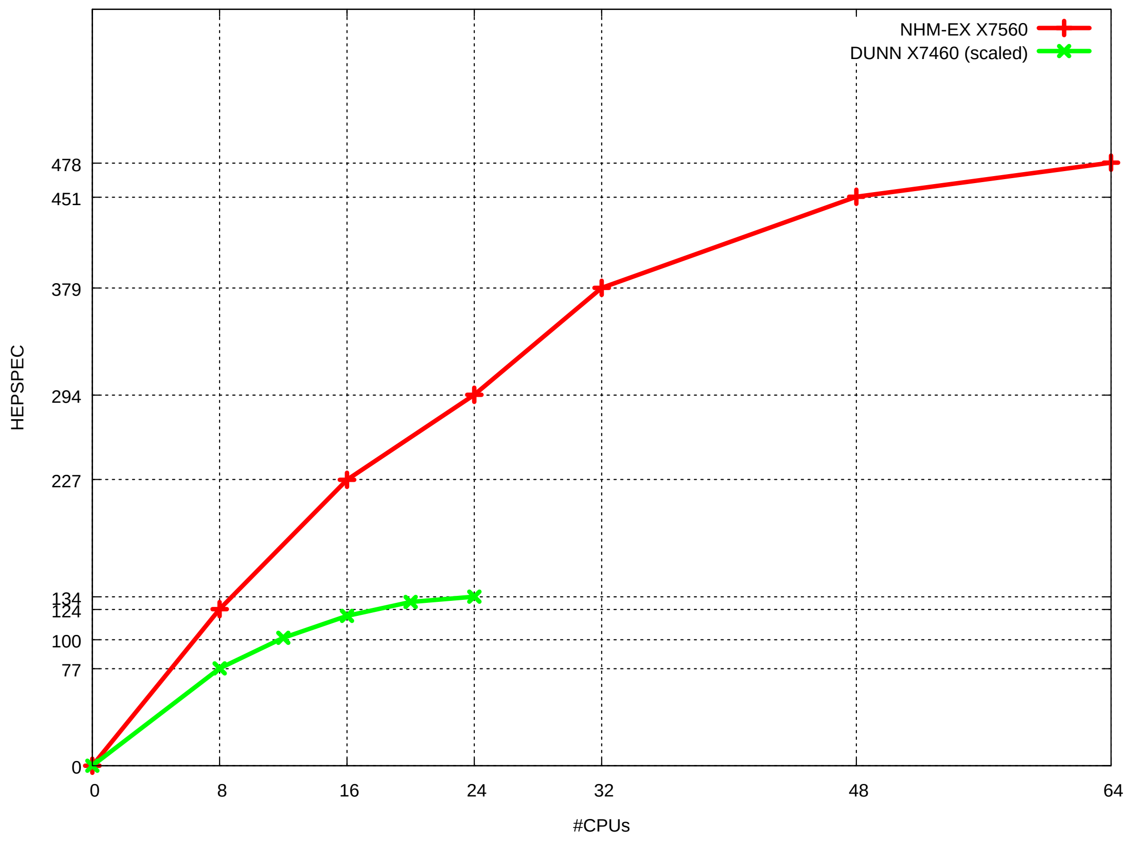

Figure 3: HEPSPEC2006 performance comparison X7560: SMT-off Turbo-on, X7460 frequency scaled

X7460 based Dunnington comparison

To compare the two systems, the X7460 results were frequency scaled, from the initial 2.66GHz clock rate down to 2.27GHz, to match the X7560 frequency. Both systems are four socket systems, aimed at the "expandable" server market, but their differences are rather huge.

As mentioned above concerning the core count, the X7460 system counts a total of 24 cores (4x6 cores), but the X7560 system offers 33% more, in the same 130W TDP per processor. SMT gives another bonus to the newer platform, offering up to 64 hyperthreaded cores.

As highlighted before, the main advantage of the "Nehalem-EX" platform is the new hierarchical memory subsystem: the Front Side Bus memory topology was already a handicap for dual processor (DP) systems, but its scalability limitation put a stop to global system scalability at a level where adding cores results in no substantial performance increase.

Number of processes	Throughput scaling	Relative efficiency
1	1.0x	100%
8	6.9x	86%
16	10.6x	66%
24	12.0x	50%

Table 4: Scalability testing of the X7460 system

Number of processes	Throughput scaling	Relative efficiency
1	1.1x	110%
8	8.0x	100%
16	14.6x	91%
32	24.4x	76%

Table 5: Scalability testing of the X7560 system Turbo-on (1 core with Turbo off used as reference)

Scalability testing shows that the behavior of the two systems running HEPSPEC benchmark is inherently different. Where the X7460 seems to reach quickly a horizontal asymptote, the X7560 system is able to sustain increasing load.

If a direct comparison to the X7460 server is considered, the tested system allows for 3x more throughput. Frequency scaled results show that the X7560 based system yields around 3.5x more throughput.

To have a better idea of the scalability capabilities of the X7560 processor, its scalability efficiency should be compared with the X5570 Nehalem-EP processor. Using a reference at one process on the system, and considering the maximum number of processes without SMT, one HEPSPEC process per core, the X5570 offers an efficiency of 79%, going from 1 to 8 processes. When considering the core increase of the EX version against the EP version, the HEPSPEC scalability of the X7560 underlines a performance balance close to the DP flavors.

SMT advantage

The SMT feature is present across all the line of the new Intel processors, providing interesting additional performance when the system has to execute more threads than its actual core count. The Dunnington cannot be used as the reference platform, because it lacks support for SMT, so the reference platform in this case will be the X5570 Nehalem-EP processor.

The gain produced by SMT can be computed by comparing the HEPSPEC06 results for 32 and 64 processes for the Nehalem-EX, and for 8 and 16 processes for the Nehalem-EP: in the case of the X5570, the SMT gain is around 24% and for the X7560 processor, the SMT gain is 26%. This again shows the rather remarkable scalability potential of the four socket Nehalem system, increasing significantly its performance up to its maximum of 64 SMT cores.

Multi-threaded Geant 4 prototype

Geant4 is one of the principal toolkits used in 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 NEU 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 (32).

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.

Scalability testing

The application tested scaled quite well up to 32 physical cores. The efficiency under full physical core load was 93%, which corresponds to a scaling factor of 29.7x. Detailed scalability data reveals a drop in efficiency when running more than one process. Continued measurements up until 32 cores revealed a modest efficiency decrease. Key scaling data points:

- 1.93x for 2 processes (96% efficiency)

- 3.87x for 4 processes (97% efficiency)
- 7.71x for 8 processes (96% efficiency)
- 15.42x for 16 processes (96% efficiency)
- 22.63x for 24 processes (94% efficiency)

The scalability data reveals a significant potential for an optimal management of a cpu-bound workload. Relative scaling up to 8 or 12 cores is comparable to the Nehalem-EP and Westmere-EP systems tested previously. A slight drop in efficiency is observed past 24 cores, the origin of which remains to be discovered. In essence, the efficiency curve is nearly flat, which means that one can expect excellent, predictable scalability with this kind of applications. The graph below (Figure 4) is limited to 32 cores only and shows the total simulation time curve in blue and the efficiency (scalability) curve in green.

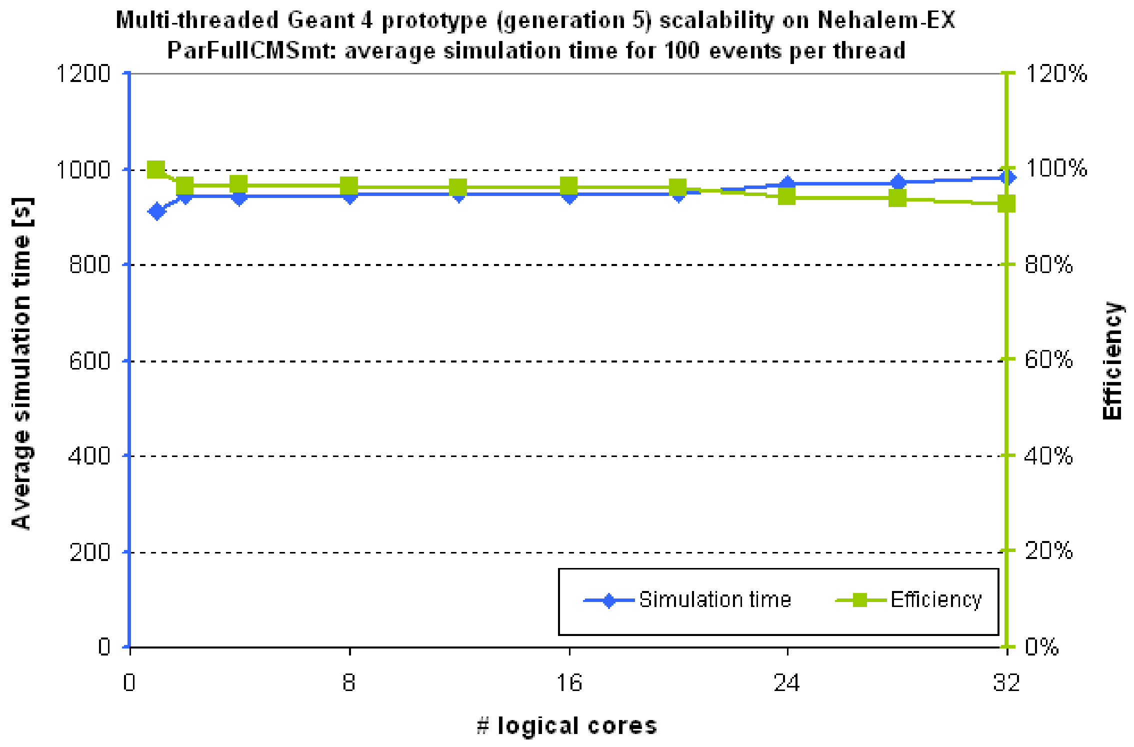

Figure 4: ParFullCMSmt scalability on Nehalem-EX (without SMT)

In contrast, the following graph (Figure 5) shows the data for points between 1 and 64 threads. The efficiency curve recovers past 32 cores and ultimately surpasses 100%, since for thread counts higher than 32, expected scalability is fixed to 32x. Thus a final value of 118% indicates that the system loaded with 64 threads of the benchmark yields 18% more throughput than a perfectly scaled serial version on 32 physical cores.

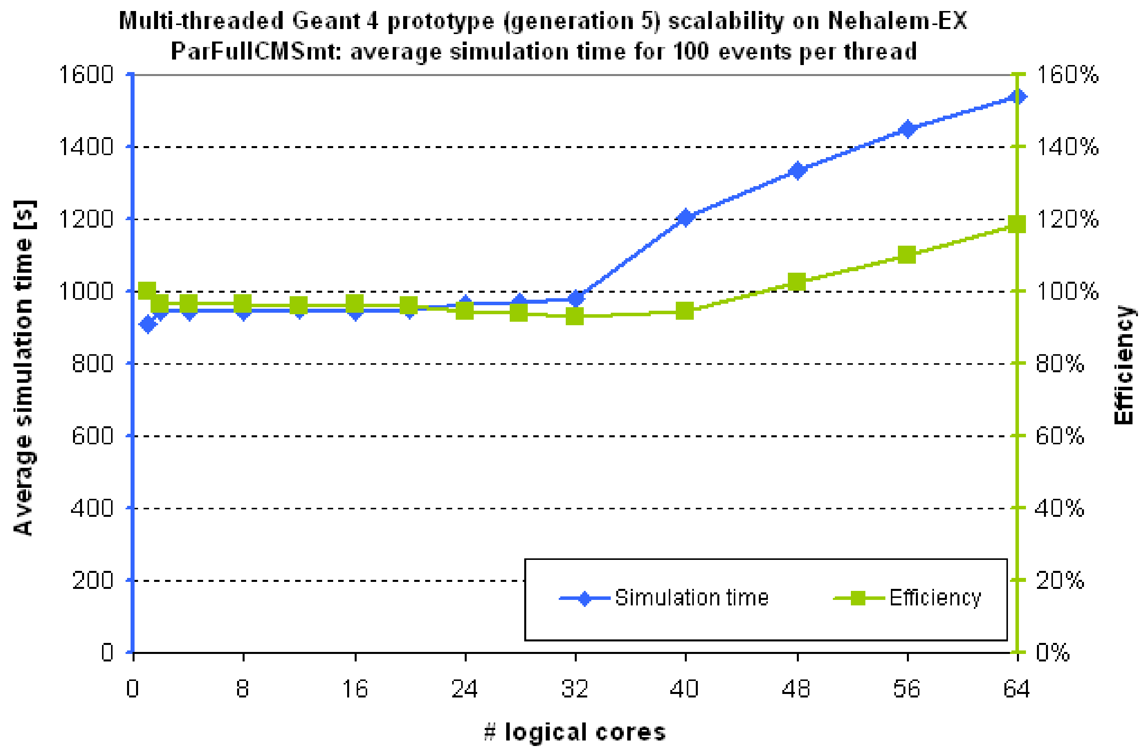

Figure 5: ParFullCMSmt scalability on Nehalem-EX (with SMT)

Hyper-threading advantage

The advantage of running with SMT was within the measurement error for 40 hardware threads, 10% with 48 hardware threads, 19% with 56 hardware threads and finally 28% with all hardware threads engaged. One should note that this extra 28% of performance is traded in for a penalty in memory usage, as the number of software threads is twice the one in the case of 32 cores.

X7460 based Dunnington comparison

The first group of results in this section is frequency scaled. When compared to a Xeon X7460 based Dunnington platform tested earlier, the Nehalem-EX performs 17% better with one thread on one core, and 12% better with 24 threads on 24 cores. Another interpretation of this figure is that in the context of a many-core x86 system, the new Nehalem microarchitecture and platform provide 17% more throughput per core than the previous Core 2 design. It should be noted that the Dunnington system provided excellent efficiency figures above 98% in the tested range between 1 and 24 cores. This result has much improved from the early Dunnington samples tested at openlab before.

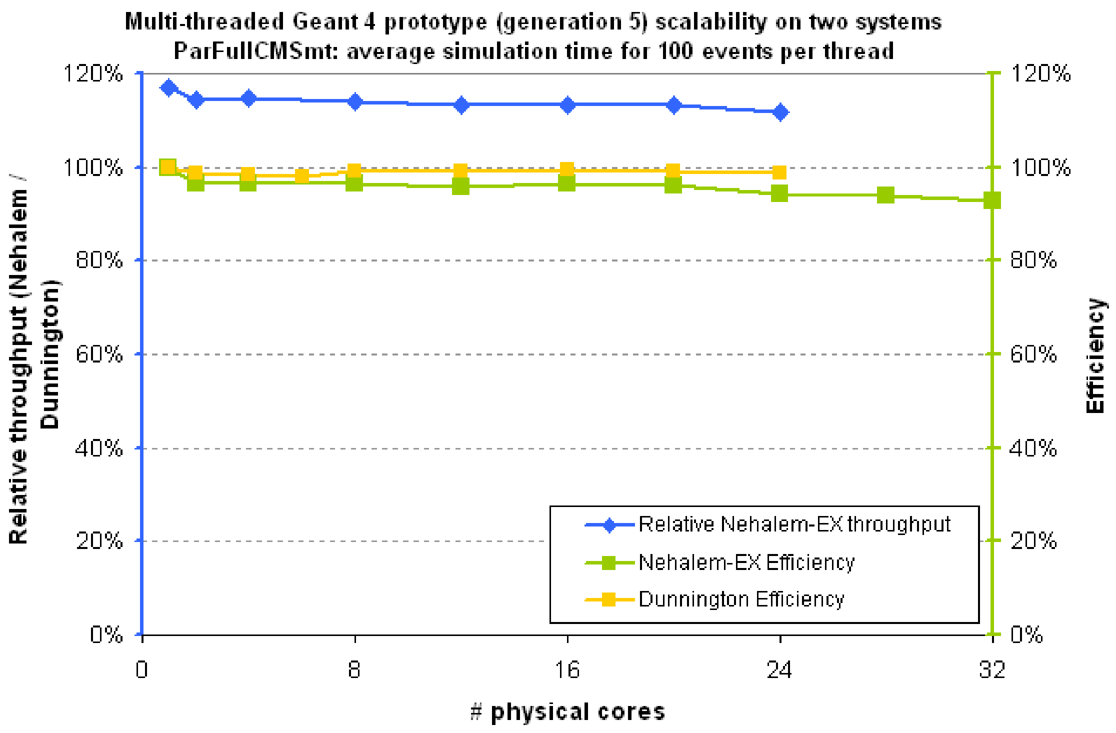

Figure 6: Nehalem-EX throughput advantage over Dunnington (frequency scaled)

The tested Nehalem-EX platform also delivers a 33% increase in core count, and adds Hyper Threading functionality in comparison to the previous solution. A Nehalem-EX machine with 32 cores fully loaded provides 47% more throughput than the equivalent Dunnington solution, and 87% more when loaded with 64 threads on 32 cores.

In essence, a Nehalem-EX core can be up to 17% faster than an equivalent Core 2 based one, while the overall system provides up to 87% more throughput when using Hyper Threading.

However, if we consider that both the X7560 and its Core 2 counterpart are “top of the line” parts with the highest frequency bins available, we can also make a realistic, direct comparison of the two. In this case, the Nehalem-EX performs slightly (1-5%) worse in terms of absolute performance, but the increased core count allows for a 25% advantage with 32 cores loaded, and the addition of SMT increases this advantage to 60% with 64 threads. This comparison is illustrated on Figure 7.

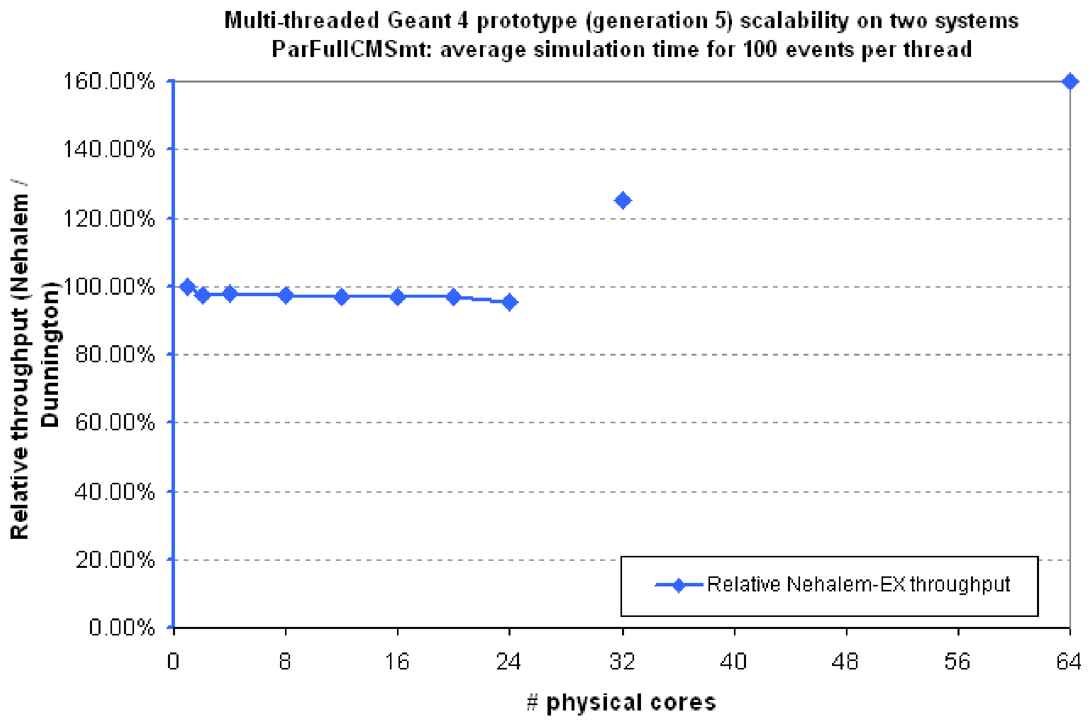

Figure 7: Nehalem-EX throughput advantage over Dunnington (direct comparison)

In essence, considering non-scaled performance in this benchmark, the tested Nehalem-EX can yield up to 60% more throughput.

Multi-threaded Test40

The multi-threaded test40 benchmark is a fusion of the classic test40 test used at openlab for quick performance studies, and the prototype of the parallel Geant4 processing framework. It focuses on a particle gun inside a simple detector geometry, still representing a substantial part of a physics simulation. It could be considered as a simplified version of a full-fledged processing framework, nevertheless bearing key characteristics of commonly used HEP software. The parallelization technology used is identical to the case described in the previous section, and has been carried out by Xin Dong. The test was executed with 200 events per thread. Other software and hardware conditions also match those from the previous test. As previously, this scenario tests the wall clock runtime per event and efficiency.

The test was run 10 times and the results were averaged over each core count. The standard deviation did not exceed 6% of the average sample value for up to 16 cores or less, and was maintained below 2.3% for all other data points.

Scalability testing

Quite similarly to the previous benchmark, this test yielded a 93% efficiency figure for full physical core load, which corresponds to a scaling factor of 29.9x. Other noteworthy scalability data points include:

- 1.90x for 2 processes (98% efficiency)
- 3.69x for 4 processes (92% efficiency)

- 7.54x for 8 processes (94% efficiency)
- 15.27x for 16 processes (95% efficiency)
- 22.88x for 24 processes (95% efficiency)

A “ditch” between 2 and 12 cores has been noticed and remains to be investigated, however this peculiarity might be an effect of the software rather than the hardware. The graph below (Figure 8) shows the total simulation time curve in blue and the efficiency (scalability) curve in green. Regardless, the efficiency is maintained at a relatively flat level.

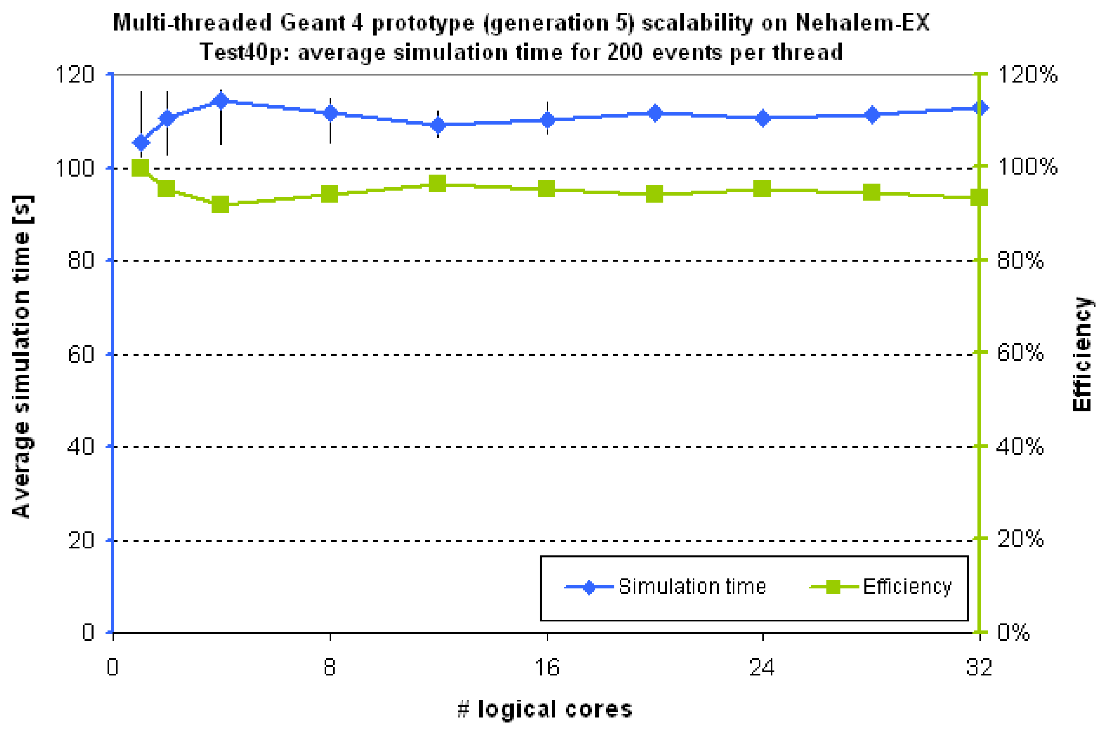

Figure 8: Test40p scalability on Nehalem-EX (without SMT)

The second graph (Figure 9) shows scalability plotted all the way up to 64 threads. The efficiency curve surpasses 100%, since for thread counts higher than 32, expected scalability is fixed to 32x. Thus a final value of 115% indicates that the system loaded with 64 threads of the benchmark yields 15% more throughput than a perfectly scaled serial version on 32 physical cores.

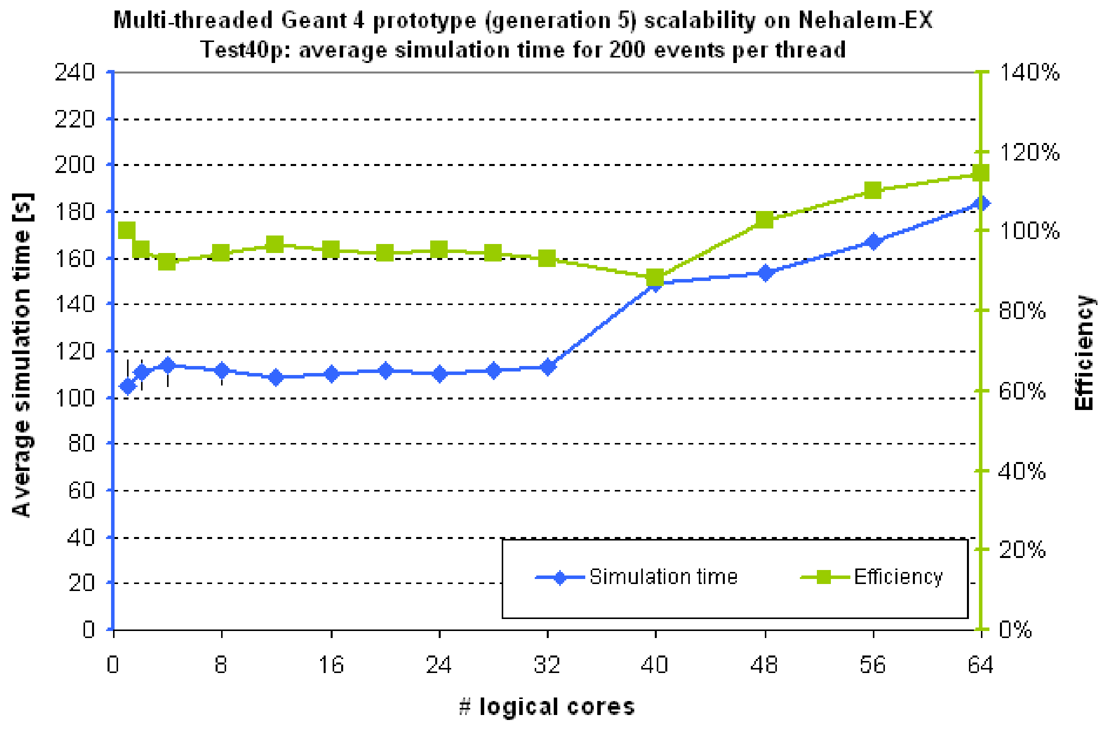

Figure 9: Test40p scalability on Nehalem-EX (with SMT)

Hyper-threading advantage

In the case of multi-threaded test40, a slight 5% disadvantage was observed when running with 40 threads. This is expected, and matches previous experience of cases where the competition for execution resources in a core provide a disadvantage which is bigger than the total benefit of engaging in-core hardware multi-threading for relatively low hardware thread counts. Increasing the thread count improved this figure, yielding a 10% advantage with 48 threads, 18% with 56 threads and a 23% advantage with all 64 threads. Similarly to the previous test, an increased number of threads implies higher memory usage.

X7460 based Dunnington comparison

The first set of results in this section is frequency scaled. When compared to a Xeon X7460 based Dunnington platform tested earlier, the Nehalem-EX performs 19% better with one thread on one core and 16% better with 24 threads on 24 cores. Another interpretation of this figure is that in the context of a many-core x86 system, the new multi-socket Nehalem microarchitecture and platform provide 19% more throughput per core than the previous Core 2 design. The advantage decrease is comparable to the one seen in the previous test.

Considering the increased core count and the addition of Hyper Threading, the tested Nehalem-EX machine with 32 cores fully loaded provided 51% more throughput than the equivalent Dunnington solution, and 86% more when loaded with 64 threads on 32 cores.

In essence, the Nehalem-EX core can be up to 19% faster than an equivalent Core 2 based one, while the overall system provides up to 86% more throughput when using Hyper Threading.

However, if we consider that both the X7560 and its Core 2 counterpart are “top of the line” parts with the highest frequency bins available, we can also make a realistic, direct comparison of the two. In this case, the Nehalem-EX performs slightly (1-5%) worse in terms of absolute performance, but the increased core count allows for a 29% advantage with 32 cores loaded, and the addition of SMT increases this advantage to 59% with 64 threads.

In essence, considering non-scaled performance in this benchmark, the tested Nehalem-EX can yield up to 59% more throughput.

Multi-threaded ROOT minimization

In general all the methods used in data analysis are based on mathematical optimizations. Depending on the particular method, the evaluation of a function is required, like the likelihood function or the expected prediction error function, which has to be optimized as function of several free parameters to be determined [Num07]. The High Energy Physics (HEP) community makes large use of many complex data analysis techniques, like maximum likelihood fits, neural networks, and boosted decision trees [Stat01]. Upon the startup of the data analysis activity at Large Hadron Collider (LHC) experiments, such techniques will be used for a better discrimination between signal and background events in order to discover possible new physics phenomena [Phys08].

The increase of the sample sizes and use of complex algorithms for data analyses require high CPU performance. Traditionally all software for data analysis developed in HEP does not use parallelism. However, with the introduction of multi-core CPUs and the usage of more and more complex algorithms, an effort for parallelization of the software has been started in recent years. The MINUIT package is the most common package used for optimization of a function in the HEP community [Min72]. The main algorithm in this package, MIGRAD, searches for the minimum of a function using the gradient information. For each minimization iteration, MIGRAD requires the calculation of the derivative for each free parameter of the function to be minimized. A. Lazzaro and L. Moneta have developed a parallel version of MIGRAD, where the calculation of the derivatives has been spread out on different processes using a Message Passing Interface (MPI) parallelization [Laz09]. Particularly interesting is the minimization procedure applied to the maximum likelihood technique [Cow98]. In this case we minimize $-\ln L$, where L is the likelihood function. This function L is the sum of different terms calculated for each event, identified by several variables, for a given data sample. Since each event is independent from the others, it is possible to distribute the calculation of each event over different processes and then collect all results for the final calculation of L for each process (using an MPI::Allreduce routine). We use the RooFit package (package inside the ROOT software framework developed at CERN) for the L calculation, where we have implemented a parallelization using MPI. A corresponding parallel version based on OpenMP is under development and is expected for the next releases of the software.

We base our tests on an analysis performed at the BaBar experiment (an experiment which was located at the SLAC National Accelerator Laboratory, California), namely the measurement of Time Dependent CP violation for the decays of neutral B mesons to η'K final state. This analysis was published in the journal Physics Review D 79, 052003 (2009). Thus, this is an example of a real-world application in use in the HEP community.

We look at the execution time spent by the application running in parallel with a certain number of processes (with Turbo mode enabled). The parallelization implemented guarantees that the results are the same in all configurations. The workload is not entirely balanced amongst the parallel processes since one of the processes does the calculation of an additional term of the likelihood function (the extended term). We look at the efficiency, defined as the scaling of the software relative to the serial runtime (scalability) confronted with ideal scaling determined by the process count. We take as reference for serial runtime the test running without Turbo mode. Given the fact that the application is not fully parallelized and the processes are not fully balanced in the workload, we expect a limit of the scalability for a high number of requested parallel processes. The fraction of execution time spend in code we can parallelize is 98.7%. Another known limitation is the increase on the time spent for MPI functions. So we also perform a profile, looking at the performance for different parts of the application, to have a better understanding of the performance. Since we are interested in strong scaling of the application (i.e. considering the same application with different number of parallel processes), there is no benefit from hardware threads. However, the system was SMT-enabled, so the hardware threading feature was activated, but not used during the tests since we run our tests up to 32 processes.

Technical setup

We compiled all the code (ROOT v5.26, RooFit, MINUIT) using GCC 4.1.2. As MPI implementation we used OpenMPI v1.4.1 compiled with GCC 4.1.2. For the profile of the application we use the TAU suite (v2.18). We have checked that the profiler does not introduce significant overhead in the execution time of the application.

Results

The application takes about 43 minutes when running it as a single process (serial) with Turbo mode enabled (47 minutes with Turbo mode disabled). We do not use any shared memory, and the resident memory footprint is about 1.2 GB for each required parallel process. Since the system has 128 GB of memory, so the application is not memory constrained.

The total execution time (wall-clock time) and efficiency for 1 to 32 processes are shown in Table 6. The efficiency is greater than 100% in some cases, since we take as reference for serial runtime the test running without Turbo mode enabled. The same quantities can be seen on Figure 10. In the same table we show the theoretical efficiency values calculated using the Amdahl's law and the fraction 98.7% of parallel portion of the code.

# Processes	*Wall-clock time [seconds]*	*Efficiency [%]*	*Theoretical Efficiency [%]*
1	2566	110	110
2	1301	108	109
4	664	106	106
8	350	101	101
12	243	97	96
16	192	92	92
20	169	84	88
24	149	79	85
28	134	75	81
32	126	70	78

Table 6: Wall-clock time and efficiency for the fit application requiring different number of processes. We also show the theoretical efficiency calculated using the Amdahl's law and the fraction 98.7% of parallel portion of the code.

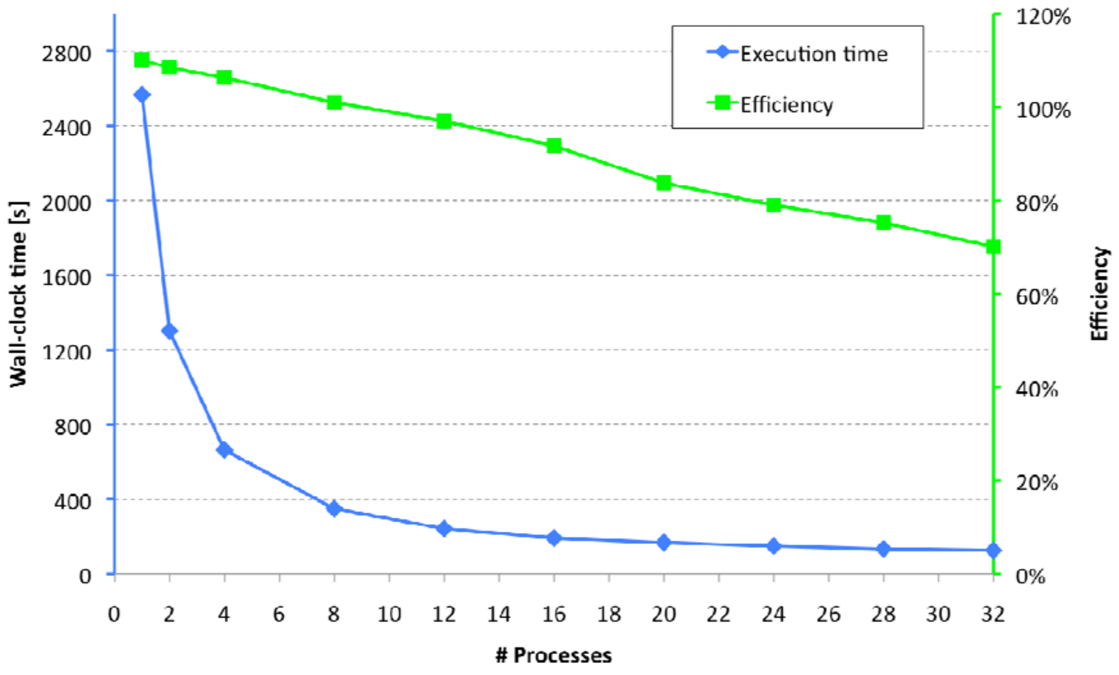

Figure 10: Time spent for the calculation of the fit application (blue line) and efficiency (green line).

The efficiency is between 70% and 110%, with a negative slope versus the increase of number of processes. This constant decrease in the efficiency is expected since we know that the application is not fully parallelized (the sequential part of the code limits the scalability, as we can also realize from the comparison with the theoretical efficiency numbers).

We also extract the contribution of the Turbo mode. We do a comparison of the results of wall-clock time obtained from tests with and without Turbo mode enabled, as shown in Figure 11. Clearly Turbo mode helps to speed up the application and there is an average benefit of 10%. In conclusion the results are in accord with expectations.

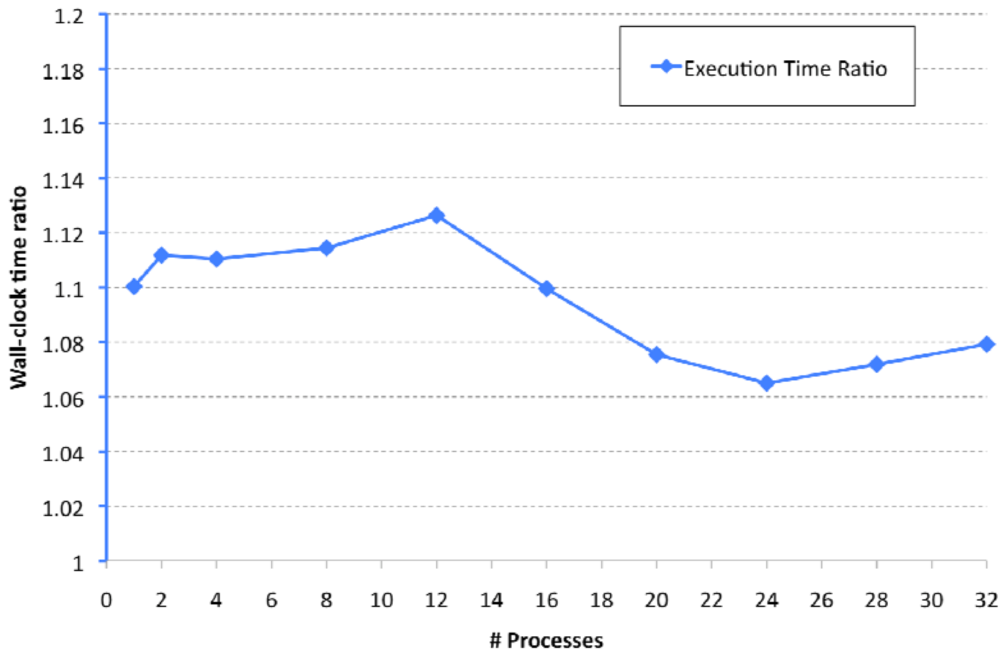

Figure 11: Ratio of the Wall-clock time results for the configurations with Turbo mode on and Turbo mode off.

X7460 based Dunnington comparison

We perform a direct comparison of the Nehalem-EX results with X7460 based Dunnington platform, with Turbo mode disabled on Nehalem-EX. We also perform the comparison scaling the results at the common Nehalem-EX frequency of 2.27 GHz.

The execution time results are shown on Figure 12. Up to 16 processes the Dunnington system provides an average improvement of 10% in performance from the direct comparison (see plot for this information). The improvement becomes smaller for 20 processes (6%) and 24 processes (5%), which is the number of physical cores on the Dunnington system. The tested Nehalem-EX platform also delivers a 33% increase in core count. This system with 32 cores fully loaded provides 11% of increase in performance with respect to the equivalent Dunnington solution. If we consider the comparison using frequency scaled results, then the new Nehalem microarchitecture and platform provide an average increase of performance of 7.5% per each core and 47% with 32 cores fully loaded with respect to the previous Core 2 design.

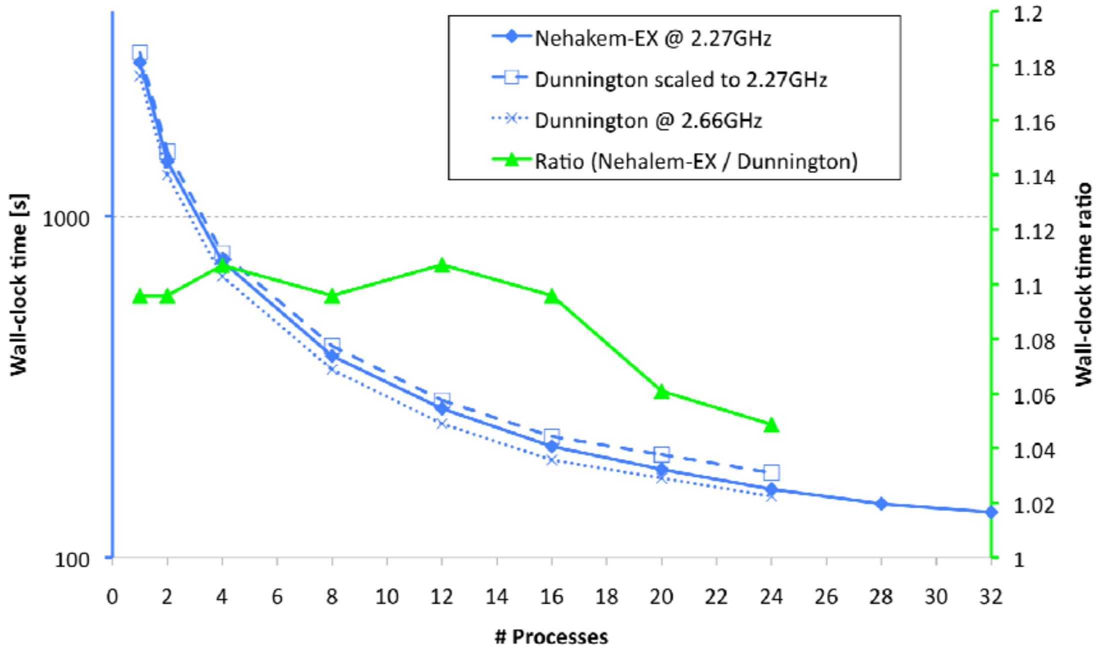

Figure 12: Execution time for the Nehalem-EX (solid blue line) and Dunnington (dashed blue line). Both results are scaled to 2.27GHz. The dotted blue line shows non-scaled results. The green line represents the non-scaled ratio between the Dunnington and Nehalem-EX results.

Conclusions and summary

The new Xeon 7500 platform provides a significant jump in performance and efficiency compared to the previous Xeon 7400 Dunnington generation. Part of the improvements can be credited to the architectural jump – "Nehalem" represents a significant improvement over "Core 2". Another part of the improvements is the effect of switching from FSB to QPI – a much more scalable bus implementation.

Core increase and architectural changes

The new parts represent a 33% core increase over the previous generation, and each chip had 8MB more of cache, bringing the total amount per chip to 24MB. The performance figures thoroughly represent these additions. If we consider clock for clock performance, it is roughly 15% to 20% better than the tested Dunnington system.

Hyper Threading

As far as Hyper Threading (SMT) is concerned, on average the performance advantage seems to be equivalent to the one obtained on Nehalem-EP. All improvement figures fall within the 19% and 28% range.

Overall platform performance

To sum up, if we consider the overall frequency-scaled throughput of the whole dual-processor system, a significant increase can be seen in comparison to Dunnington based servers. The measured throughput slightly exceeded 3.5x that of the Dunnington for the HEPSPEC06 benchmark (SMT included), and has increased between 47% and 87% for in-house applications (SMT off and on respectively). The noteworthy result for HEPSPEC06 could be credited in a large part to weak performance on the Dunnington, possibly stemming from an FSB bottleneck.

If we consider that both the X7560 and its Core 2 counterpart are “top of the line” parts with the highest frequency bins available, we can also make a realistic, direct comparison of the two. In the case of the HEPSPEC06 benchmark, throughput is approximately tripled. Other applications that were tested provide between 11% and 60% more throughput.

The overall power consumption of the system was quite significant. Although Nehalem-EX is unlikely to become the prevalent architecture in the CERN computing center, power consumption improvements are always welcome, as the power consumption of a system can contribute a significant portion of its lifetime cost. As already mentioned, the power-performance of the Dunnington system is not directly comparable and was not evaluated in this context.

References

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)
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)
Stat01	J. Friedman and T. Hastie and R. Tibshirani, *The Elements of Statistical Learning*, Springer (2001)
Phys08	G. Kane, A. Pierce, *Perspectives on LHC Physics*, World Scientific (2008)
Min72	F. James, *MINUIT - Function Minimization and Error Analysis*, CERN Program Library Long Writeup D506 (1972)
Laz09	A. Lazzaro and L. Moneta, *MINUIT Package Parallelization and applications using the RooFit Package*, Proceedings of Science, PoS(ACAT08)083 (2009)
Cow98	G. Cowan, *Statistical Data Analysis*, Clarendon Press, Oxford (1998)

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 13.

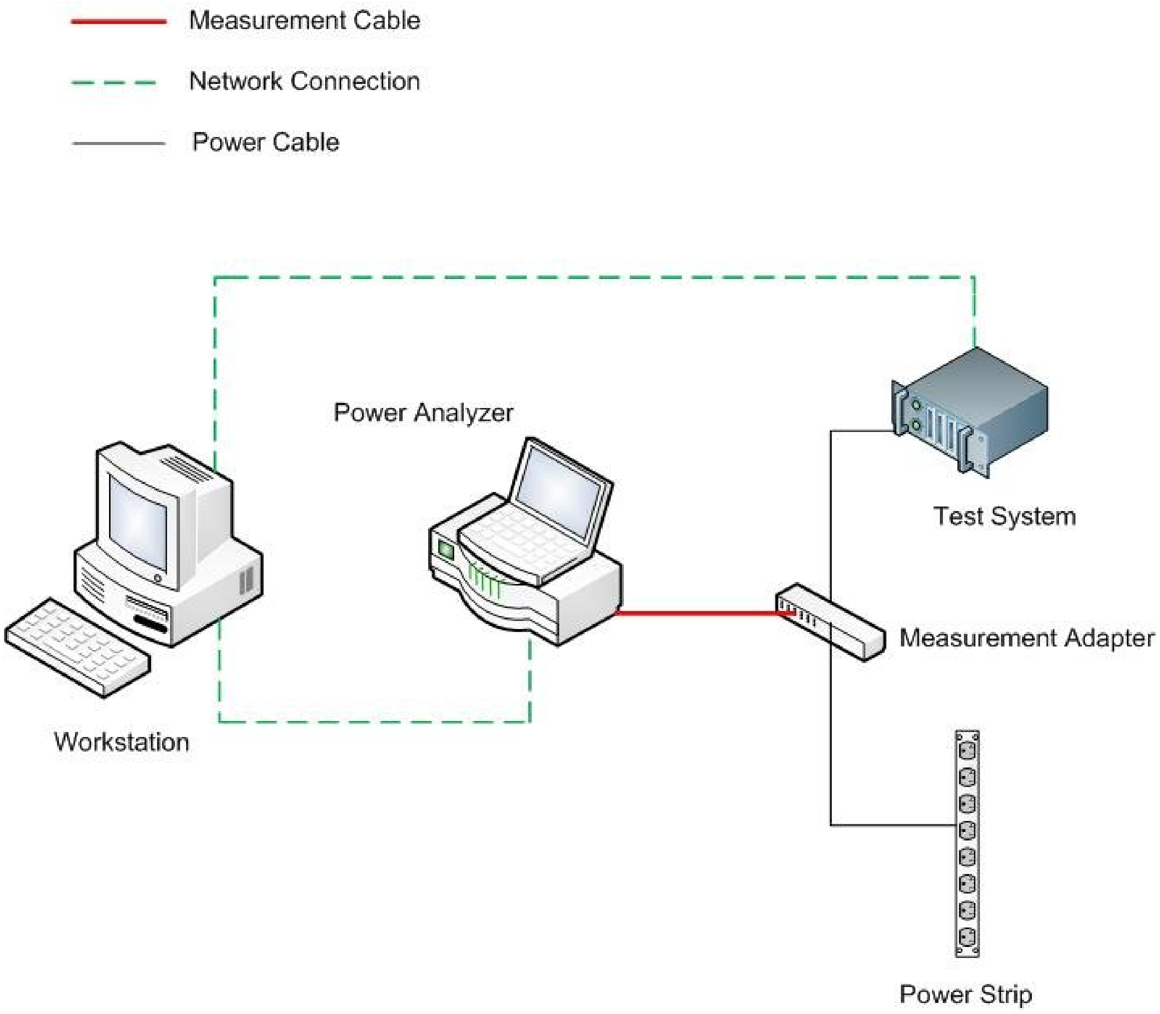

Figure 13: Power test setup

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%.