Performance testing of a Quad-Core Server for the Physics Database Services

CERN-IT/PSS, April 2007

Executive summary

This article reports on the tests performed at CERN-IT/PSS in March 2007 on a latest-generation server based on dual quad-core Xeon processors (quadcore) and FB-DIMM RAM. The tests are aimed at evaluating the use of quadcore servers for the Physics Database Services (PDB).

The quadcore server has been tested using Oracle-generated workload and focusing in particular on CPU and memory access performance. Results have been compared with a system representative of the infrastructure currently deployed in production, i.e. a 6-node RAC cluster of dual-CPU PIV Xeon servers with 4GB of DDR2 400 memory each. The quadcore server is a dual-CPU Core 2 Quad with 16 GB of FB-DIMM memory.

The results of the performance tests show that quadcore servers display approximately a 5 fold performance increase compared to the existing production, for memory-intensive workload (negligible physical IO in the workload). In other words for memory and CPU intensive jobs a single-node quadcore server performs as 5 nodes of our current production RACs.
Moreover tests have shown that the quadcore can also increase the overall performance of single-threaded CPU-intensive operations thanks to improved CPU-to-memory access.

This article reports the details of three tests were PIV and dual-core HW performances have been compared head-to-head: 1. measurement of concurrent memory access using Oracle workload, 2. throughput measurements of a representative Physics application, 3. throughput measurements of Oracle Streams replication.

Further details on the HW specifications and the tests performed are available in a wiki (access may be restricted):
http://twiki.cern.ch/twiki/bin/viewauth/PSSGroup/QuadCoreTests

Stress testing memory access

Response time measurements

The results reported below are measurements of CPU to memory access measured using Oracle workload. For this test a custom multi-user version of the Jonathan Lewis Oracle Computing Index query (JLOCI) has been developed. Each instance of the JLOCI query used here executes in memory (logical IO) using about 20000 x 8KB Oracle buffers of shared memory (buffer cache in the SGA) and 90 MB of per-process (private memory also known as PGA). The results of the throughput measurements for single instance, RAC and quad-core setups are show below in Figure A1.
From the data we conclude that:

- The response time of the quad-core machine falls between the response time of the 4–node RAC and 6-node RAC. Within the range of measured server loads (loads that make sense for DB production purposes) we can conclude that the quad-core server can sustain the same logical IO workload as 5 of the servers currently deployed in production.
- Single-threaded execution on the quad-core machine is 300% faster than with the existing HW for this type of workload.

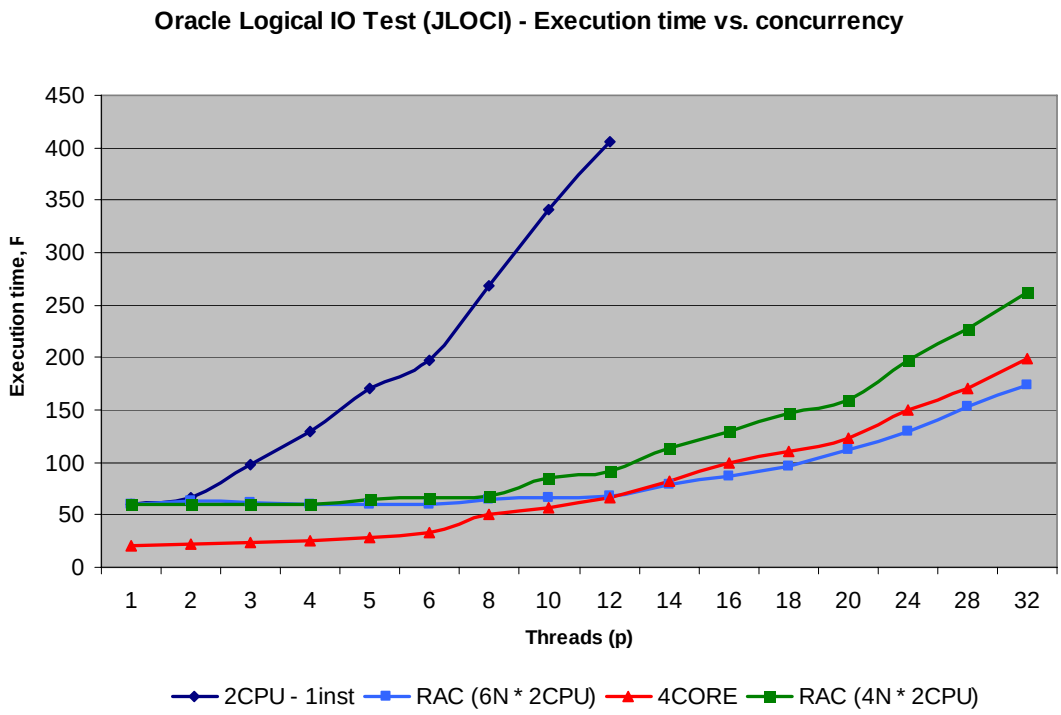

Figure A1: *Response time R(p) vs concurrency (p), of a memory-intensive workload (JLOCI test). The results show that the quad core (red line) has a throughput comparable to about 5 nodes of a RAC of PIV machines.*

Scalability and saturation effects for highly concurrent memory access

This paragraph looks into more details at the scalability of memory access as measured with the JLOCI test discussed in the above paragraph. Scalability analysis highlights a different behavior between PIV and quad core in the case of highly concurrent memory access workloads. The scalability factor C(p) is the normalized response time and can be calculated from the response time R(p) time data reported in Figure A1 above, as $C(p) = p / R(p)$ where p is the degree of parallelism (i.e. N# parallel threads).

The resulting scalability data can be modeled and interpreted using Amdahl's law:

$$C(p) = C(1)\frac{p}{1+\sigma(p-1)}$$

where σ is a 'fit parameter' representing a measure of serialization.

CPU-only workload are expected to have $\sigma^{-1} \sim$ /N# CPUs. Measurements taken on a single PIV server show indeed σ^{-1} approximately equal to 2, the number of CPUs deployed. Results on RAC also show $\sigma^{-1} \sim$ /Total_N# CPUs, incidentally showing that the multi-user JLOCI test used here scales with RAC.

However, the measured value of σ^{-1} for the quad-core server is ~3.5, which is < 8, the expected maximum value of σ^{-1} for 8 cores.

The deviation of the measured valued of σ^{-1} from the maximum value of 8 can be explained as the outcome of serialization effects induced by the type of workload tested: highly concurrent access to memory (see also next paragraph).

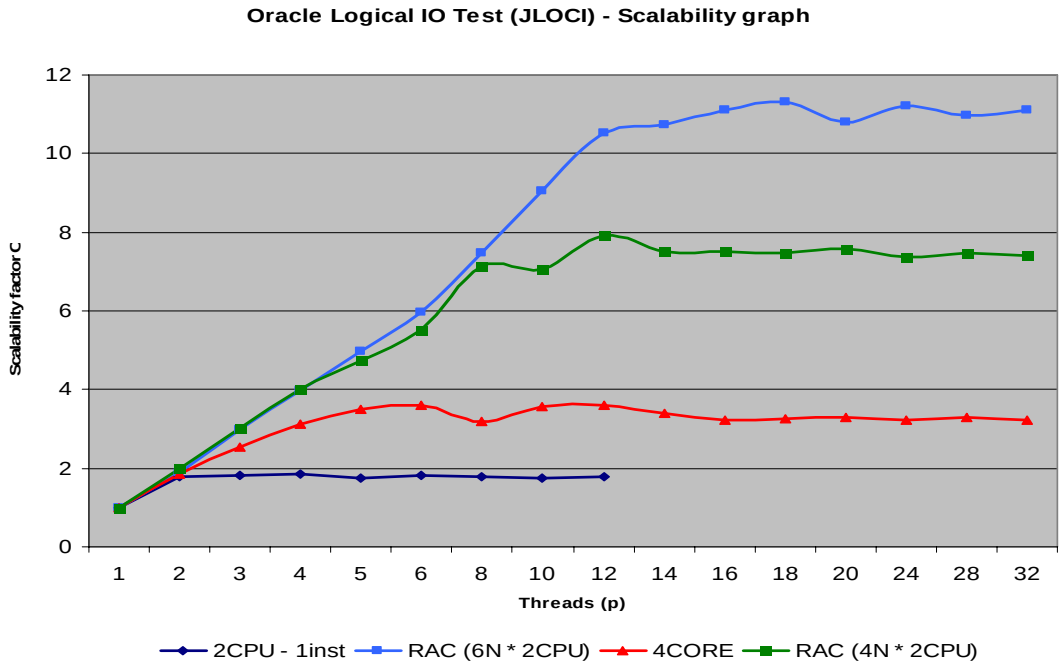

Figure A2: *Scalability factor Cp) for vs. concurrency (p), of a memory-intensive workload (JLOCI test). The results show that the quad core (red line) has different scalability behavior than PIV in the case of highly concurrent access to memory.*

Scalability of a simpler workload – linear scalability

As a sanity check of the scalability behavior of the processors under test, a simpler workload is examined in this paragraph. The scalability graph of a workload consisting of a 'for loop' that runs completely in CPU and CPU cache is reported in Figure A3 below. Comparing this case with the memory test reported in the paragraph above we note that the quad core server in the case of this simpler workload scales well and can be fitted with Amdahl's law with $\sigma^{-1} = 8$ (see Amdahl's law above).

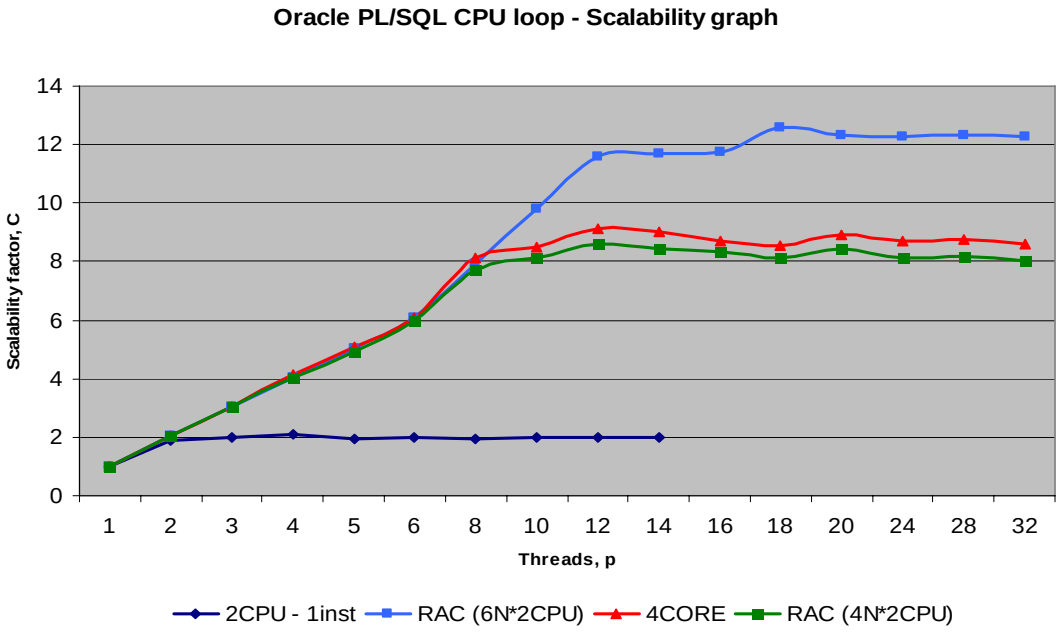

Figure A3: *Scalability factor Cp) for vs concurrency (p), of CPU intensive workload (simple for loop). The results show that the quad core (red line) scales as the PIV architecture to the maximum value:* σ^{-1} *=N#CPUs*

Quad-Core Performance Testing with the PhEDEx Application

PhEDEx (Physics Experiment Data Export) is a highly distributed and transaction oriented application used to control file replication between LCG GRID sites. Due to its interesting characteristics and importance for the CMS experiment it has been chosen for the comparison tests of the quad core server.

Goals

The main goal of this test was to check whether the quad-core servers can be successfully used as nodes for RAC clusters handling OLTP-like load coming from a highly distributed application. The extra goal was to compare performance of a database running on the quad core server with a 6-node RAC database (Xeon PIV architecture) while stressed with an OLTP-like workload.

Test runs

The test has consisted of two independent runs performed using the same, dedicated client hardware. During first run the 6 node RAC was being used as database backend. During second one PhEDEx data was being stored in the database running on the quad-core server. Each test run was being kept running till maximum possible transfer rate has been reached and stabilized.

Test results

The figures below show transfer rates measured during the test runs.

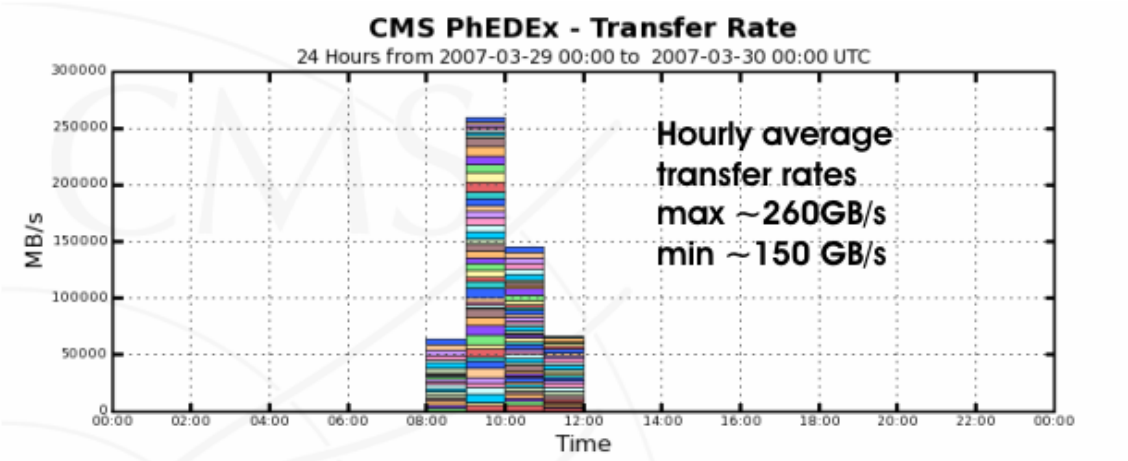

Figure B1: *PhEDEx Transfer Rate (6-node RAC). During the test the software was managing artificial file transfers using a 6-node RAC as the database backend. After ramp up phase PhEDEx was able to handle transfers with average rate of ~200 GB/s.*

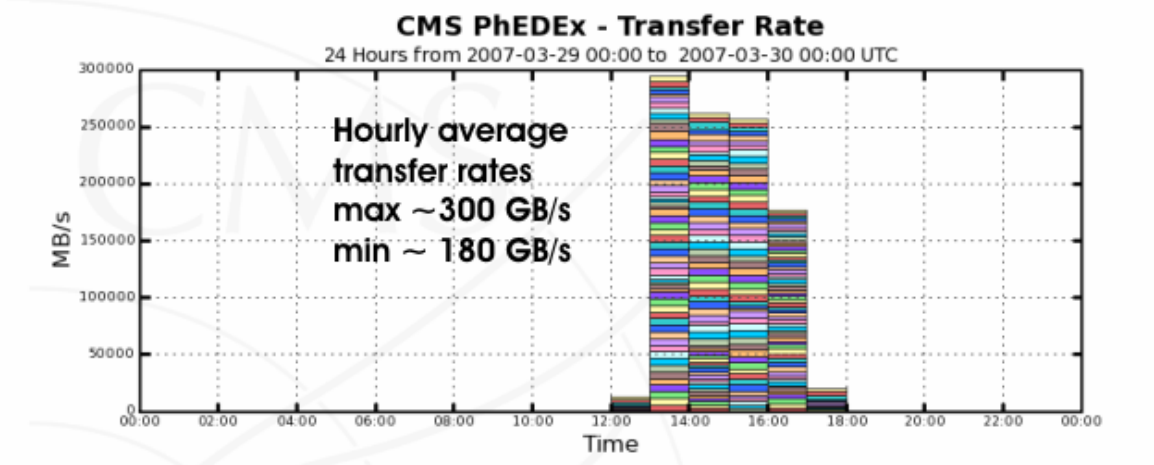

Figure B2: *PhEDEx Transfer Rate (quad core server). During the test the software was managing artificial file transfers using a database running on the quad-core server as the database backend. After ramp up phase PhEDEx was able to handle transfers with average rate of ~250 GB/s.*

The test results indicate that a single quad core server is able to handle PhEDEx-like workload even more efficiently then a 6-node RAC (~250GB/s vs. ~200GB/s average after reaching the maximum possible speed). Although during the test (due to limited client hardware) in neither case it was possible to saturate CPU resources, it was still very relevant and useful as the transfer rates achieved were by orders of magnitude higher then the one observed for the real PhEDEx production (~550 MB/s average in March 2007).

Quad-Core Performance and Oracle Streams

Test description

The purpose of the test was to measure the performance of a quad-core machine running the Oracle RDBMS and Oracle Streams replication. The source database was a dual-node RAC based on Pentium IV, whereas the destination (apply) machine was a quad-core single node DB server.
In our case, the replication performance can be considered as the time, after which the data is accessible on the destination database or the rate of logical change records (LCRs) applied/propagated per second. The test case consisted in the replication of 7 EcalPedestals objects in a single transaction. Each object consists of 61200 rows (2 Numbers and 6 Binary Floats each), which imposes that commit follows 428400 DML statements. Only one transaction was being run at a time.

Pentium 4 apply performance

Although the destination database was the 6-node RAC, apply process can only utilize the resources of a single machine.
The tests indicated that the average replication time was 300 seconds. The apply writer phase took on average 160 seconds, the remaining time (propagation and transaction assembly) was 140 seconds. This corresponds to 3000 LCRs per second (propagation and apply).
Low level performance analysis indicated that the CPU load was roughly 50% for both processors in an apply server (writer) phase. That suggests the performance issue was I/O or memory bound. The CPU usage on the destination during apply reader phase was 40-70% for both processors at a time, so the average total load was 120% per 2 CPUs. AWR report revealed 120 seconds of “Streams AQ: enqueue blocked due to flow control” wait events.

Quad-core apply performance

The tests have resulted in 70% performance improvement. The total replication time was 180 seconds, of which 85 seconds falls on apply writer. That gives the rate of 5300 LCRs per second.
Due to the nature of the test (1 transaction), the apply server utilizes only one task (thread). The CPU load in the apply writer phase confirmed that only one (of eight) cores was utilized (100%). Changing the apply process parallelism parameter from 1 to 8 didn't change the load distribution as well (no parallel transactions). During the propagation-apply reader (transaction assembly) phase, the load was evenly distributed between all of the cores (30-40%), with one core usually being loaded more than others (70-80%).

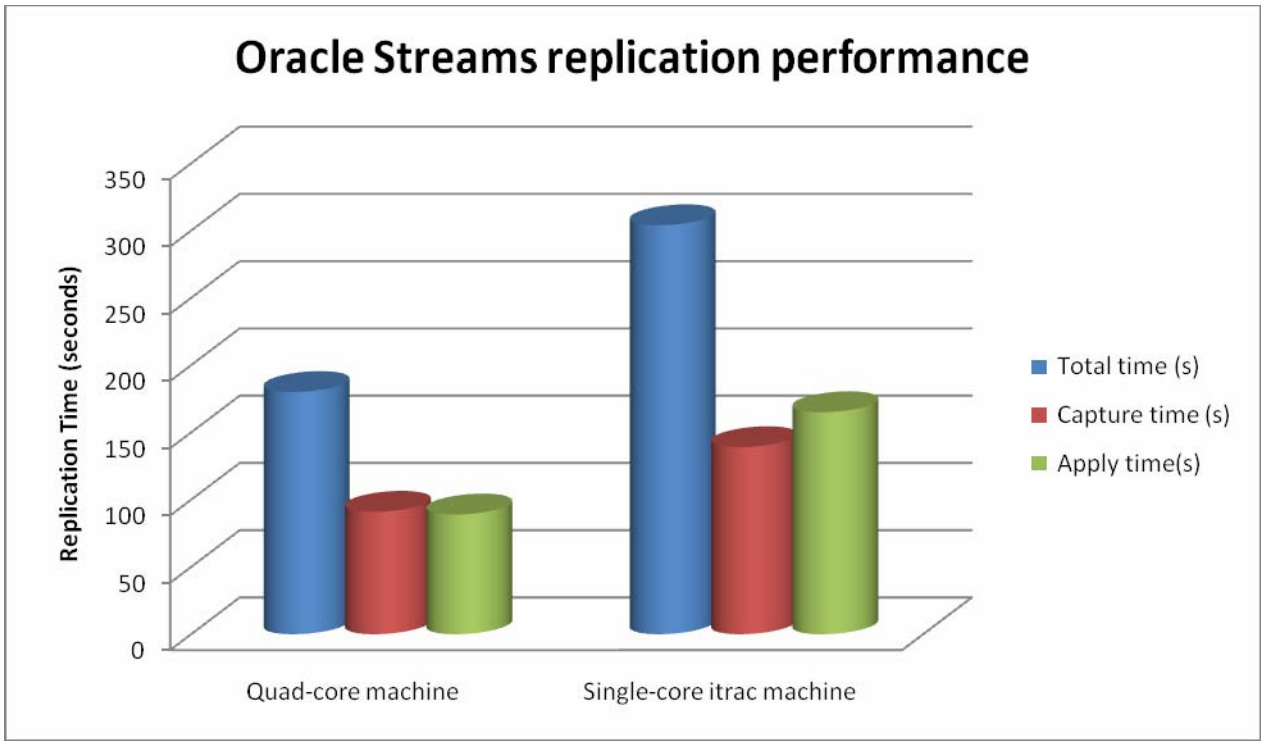

Figure C1: *Total and partial replication time (in seconds) on quad-core and single-core machines. The quadcore results show a 70% increase in performance compared to the current production HW (PIV Xeon). The performance increase is due to faster memory access and the deployment of a larger buffer cache.*

Comparison of AWR reports has shown the following differences in setup and test results:

- The amount of buffer cache increased from 768MB to over 5GB.
- No physical reads in quad-core setup
- "Streams AQ: enqueue blocked …" wait event takes 60 seconds

The performance of Streams replication depends on the number of LCRs in a transaction. If it exceeds the threshold, the messages (LCRs) are spilled to the database tables by apply process. The overhead related to the fact of reading and writing the LCR from the tables is very high.

The quad-core database makes use of a lot more Buffer Cache memory so the overhead of spilling the messages to the database tables seems to be reduced and therefore it is possible to saturate the CPU core responsible for running the apply writer process. The performance improvement is a result of faster memory subsystem, rather than SMP architecture.

Conclusions

In conclusion, the performance of streams replication test described above has shown a speedup of about 70% (compared to the current production HW), when quad-core based servers are used at the apply side of the replication chain.