February 2007
Performance Analysis

Tuning The Performance of ANSYS® FLUENT® on IBM® BladeCenter® HS21 and HS21 extended memory

Douglas M. Pase and Hari Reddy
IBM Systems and Technology Group

Abstract

In this paper we examine some of the factors that affect the performance of the ANSYS® FLUENT® benchmark[1] *running on IBM® BladeCenter® HS21*[2] *and its latest refresh, the IBM BladeCenter HS21 extended memory. Factors we examine include the choice of processor, BIOS configuration, memory configuration, choice of network, task layout and number of processors per blade. We also compare its performance against the IBM BladeCenter HS20*[3] *and the AMD Opteron LS21 for IBM BladeCenter*[4]. *Compared to the HS20 and LS21, the HS21 and HS21 extended memory perform well on this benchmark even in an untuned state, but tuning can provide significant additional performance.*

1. Introduction

The IBM BladeCenter HS21 is a low-power, high-density modular blade based on the Dual-Core Intel® Xeon 5100 series and Quad-Core Intel Xeon 5300 series processors. The design has been recently updated to better support new processors and to further increase performance. The original HS21 design supports four DIMMs of Fully-Buffered DIMM (FBD) technology. The new design supports eight. Both are powerful systems that are highly suitable for a wide variety of HPC workloads. We have used the FLUENT benchmark to measure both blades and find their performance to be excellent. In this paper we report the performance findings and what was done to tune the benchmark. Throughout the paper we refer to the new design as the HS21 XM to distinguish it from the original HS21 blade.

The FLUENT benchmark is based on the FLUENT application from ANSYS. FLUENT is a popular multiphysics code that includes Computational Fluid Dynamics (CFD) physics. FLUENT models many aspects of a system, including compressible and incompressible fluid flow, turbulence, heat transfer and acoustics. It is an extensive and widely used application. The benchmark is composed of nine separate tests, with three small, three medium and three large data sets. Each data set is run with some number of cores and the number of cores is reported along with the benchmark performance. Benchmark performance is reported as rate, and the rate for each data set is reported individually. No aggregate performance metric is computed. Parallelism is supported within the application through the MPI message passing library[5]. The benchmark does not distinguish between MPI tasks running on the same node and those running on different nodes. Only the number of “processors” is reported, which may mean MPI tasks or processor cores.

2. System and Processor Descriptions

For these tests we used a variety of systems, each of which will be described here. Our intent was to get a sense of the benchmark and understand what types of factors it is sensitive to.

The first system we used was the HS20 blade using two 3.2 GHz Intel Xeon EM64T processors and four DIMMs of 400 MHz, 1GB DDR2 main system memory. Each processor has 2MB L2 cache and supports 64-bit virtual addresses. It also supports SSE2 and SSE3 floating-point execution units, and is capable of completing two floating-point operations per clock. These are older processors that have only one core per socket. The memory bandwidth for this system was

6.4 GB/s and its peak floating-point performance was 12.8 gigaflops per second (GF/s). We used the default BIOS settings with HyperThreading disabled.

We also used two LS21 blades, one with two 2.0 GHz dual-core AMD Opteron processors, the other with two 2.2 GHz dual-core AMD Opteron processors. Each core has a 1MB L2 cache and each system was supplied with eight 1GB DIMMs of 667 MHz DDR2 main system memory. The default BIOS settings were used. Each processor core supports SSE2 and is capable of up to two floating-point operations per clock. As is typical of all Opteron processor-based systems each processor socket supports two memory channels, which gives four channels per system. Peak memory bandwidth is 10.6 GB/s per socket, or 21.2 GB/s per blade. Peak floating-point performance is 16 GF/s for the 2.0 GHz blade and 17.6 GF/s for the 2.2 GHz blade.

The next system we used was the original HS21 blade. This server was configured with two dual-core 3.0 GHz Intel Xeon 5160 processors. Each processor supports a 4MB L2 cache that is shared between both cores. The blade was supplied with four 667 MHz, 2GB DIMMs of Fully-Buffered DIMM (FBD) memory distributed across two memory channels. Each processor core was capable of completing four floating-point operations per clock and there were two cores per socket. Peak floating-point performance was 24 GF/s per core, or 48 GF/s per blade. Peak memory bandwidth was 10.6 GB/s. Default BIOS settings were used, which had hardware prefetch enabled.

The final system we used was the HS21 XM blade. We used both the dual-core 3.0 GHz Intel Xeon 5160 and quad-core 1.86 GHz E5320 processors. The system supports eight DIMMs of 667 MHz 2GB FBD memory spread over four channels, and we used both four and eight DIMM configurations. Peak memory bandwidth was 21.2 GB/s. The tuning of the BIOS will be described later in this paper.

The Intel Xeon E5320 processor is essentially two Xeon 5160 processors combined into a single package with a little extra glue logic and reductions in both the processor and front-side bus frequencies to make everything work within power, thermal and electrical constraints. In our configuration the front-side bus speed was reduced from 1333 MHz to 1067 MHz when the E5320 was used, and the processor clock frequency used was 1.86 GHz. This gives a peak floating-point performance of just under 60 GF/s and peak memory bandwidth of 17 GB/s.

3. System and Processor Selection

In this section we present the results for each of the systems and processors we tested. In some cases we tested only a single blade. Where the results were promising we also ran on two or four blades. So, for example, the HS20 with two single-core Xeon EM64T processors only has entries for one and two cores, while an HS21 XM with two quad-core Xeon E5320 processors has entries for up to 16 cores. Table 1 shows the results for all systems. For this table we used all cores in each blade before expanding to more blades. So, for example, eight cores in HS21 blades represents exactly two blades and not half of the cores in four blades. (Section 9 describes the effect of using a subset of the cores of each blade.)

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS20 Using 3.2 GHz Intel Xeon EM64T (SC, 2M L2) Processors and Gigabit Ethernet									
1	727.9	723.9	308.6	107.4	288.3		119.7	91.3	
2	138.6	554.4	217.5	96.9	225.3	238.5	97.9	74.9	
LS21 Using 2.0 GHz AMD Opteron (DC, 1M L2) Processors and Gigabit Ethernet									
1	1932.0	1941.6	1197.9	427.5	874.5	193.8	104.9	88.8	
2	3886.6	5877.6	2439.0	863.6	1240.9	382.0	215.5	180.5	
4	6597.9	10075.8	4958.4	1302.7	2782.6	590.6	430.3	335.5	
LS21 Using 2.2 GHz AMD Opteron (DC, 1M L2) Processors and Gigabit Ethernet									
1	2083.4	2074.4	1292.0	466.0	950.2	210.0	113.5	95.2	
2	4260.4	6378.7	2814.3	937.9	1355.8	412.8	231.8	193.2	
4	7111.1	10779.8	4847.1	1426.3	3018.3	649.6	465.1	336.1	
HS21 Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	3631.8	3260.4	2258.8	790.5	1519.1	320.8	204.5	145.4	
2	7146.4	8944.1	4250.9	1473.1	2736.3	623.3	379.9	275.1	
4	11636.4	12467.5	6215.8	2223.9	4261.4	869.2	625.2	404.2	
8	18285.7	18000.0	11675.7	4209.5	8597.0	1755.2	1165.2	774.4	132.0
16	21439.2	37978.0	17906.7	6803.1	16225.4	3061.1	2146.6	1555.4	281.4
HS21 XM Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	3631.8	3266.5	2214.0	813.0	1569.5	331.7	211.5	152.8	29.1
2	4997.1	8692.2	4523.6	1499.3	2858.6	655.8	393.9	295.2	54.8
4	11851.9	13200.9	6830.0	2300.9	4589.6	947.9	671.3	458.7	79.5
8	19393.9	19415.7	12943.8	4298.5	9265.4	1922.1	1292.0	923.3	157.8
16	22153.8	39183.7	19636.4	6898.2	17454.5	3223.9	2332.0	1792.5	326.7
HS21 XM Using 1.86 GHz Intel Xeon E5320 (QC, 4M shared L2) Processors and InfiniBand									
1	2260.6	2162.7	1453.3	499.9	961.3	209.1	130.6	97.3	18.5
2	4343.9	5767.7	2899.3	917.7	1762.4	417.1	242.8	188.1	35.2
4	7272.7	9994.2	4853.9	1528.5	3266.5	736.3	455.0	341.0	59.2
8	12075.5	11098.3	6870.8	2624.1	5260.3	998.6	737.5	449.8	75.9
16	17041.4	27428.6	10601.2	4374.7	10347.3	1936.1	1357.4	870.5	146.9

Table 1: FLUENT Performance on All Systems

When comparing benchmark performance across processor technologies, it is sometimes tempting to base comparisons on equal numbers of cores. But when the processors have different core counts, this can encourage misleading conclusions. For example, comparing two cores on a dual-core Xeon 5160 processor-based system against two cores on a quad-core Xeon E5320 processor-based system would be misleading from the standpoint of both price/performance and raw performance. It would compare the performance of only half of a Xeon E5320 processor against a full Xeon 5160 processor. In addition, the E5320 processor must use a slower clock frequency to support two extra cores it is not allowed to use. So, while such comparisons can be useful on occasion, in every comparison caution is advised.

Our first set of results in the table shows the performance of the HS20 using 3.2 GHz Intel Xeon EM64T processors. This system has the lowest peak memory and processor performance of all systems we tested, and it’s no surprise that it has the lowest FLUENT performance as well. It is also the oldest of the systems and has the fewest cores to work with. Both cases of FL5L3 failed due to insufficient system memory. This is a common thread through many of the tests. Through experimentation we learned that all cases, except FL5L3, can run in 4GB of memory or less, but FL5L3 requires between 8GB and 16GB of system memory. We were unable to obtain 16GB for all systems, but where we could we provide those results.

The next two sets of results are from AMD Opteron processor-based LS21 blades, and it is no surprise that the results are clearly superior to those of the HS20. In this case the LS21 provides superior processor and memory performance compared those of to the HS20, and the FLUENT results clearly show it.

That said, comparing single-core FL5L2 results shows some interesting effects. The 3.2 GHz HS20 is slightly faster than the 2.0 GHz LS21, but slightly slower than the 2.2 GHz LS21. This difference might be unexpected to some because the LS21 has faster memory and faster floating-point performance. The HS20 advantage is its larger L2 cache. In this case the 2MB L2 cache reduces the memory traffic sufficiently so that the HS20 is faster for this case. Of course, in almost all of the remaining cases the additional cores and higher memory performance of the LS21 give it far superior performance.

We ran the LS21 test cases with two different processor speeds to determine whether FLUENT was constrained at all by memory performance, or whether Opteron provided sufficient memory performance so that benchmark performance would scale linearly with processor frequency. It can be seen from comparing the two tables that while FLUENT scales fairly well with processor frequency, system memory throughput is clearly being saturated in many cases. It is also quite apparent that saturation increases as the number of threads are called into play, and as the test cases become larger. In all but one case (4 cores, FL5S3), the 2.2 GHz results are faster than the 2.0 GHz results, but the difference is usually smaller than the difference in the speed of the processors (10%).

The next two sets of results in the table show the performance of the 3.0 GHz Intel Xeon 5160 processor. The first set shows the performance when used in the HS21 blade; the second set shows its performance when used in the HS21 XM blade. The difference between the two systems is primarily in the memory structure. The original HS21 used two channels to memory in the main blade and two channels were reserved for extended memory in an extension blade. The HS21 XM uses all four channels to memory in a single-wide blade. Our tests used only a single-wide HS21 blade so the difference reflects whether two channels (HS21) or four (HS21 XM) were used. (In Section 10 we also examine the memory configuration of the HS21 XM blade.)

Comparing the Xeon 5160 results with the Opteron results clearly shows that the Xeon 5160 processor has a significant advantage. The peak performance of the Xeon 5160 (48 GF/s per blade) is much higher than Opteron (17.6 GF/s per blade), and that clearly had an effect. The memory performance was also sufficient to allow the Xeon 5160 to make use of its extra core performance. However, while the Xeon 5160 has more than twice the peak core performance of Opteron, its FLUENT performance is much less than twice Opteron's FLUENT performance.

Comparing the HS21 and HS21 XM results shows a small advantage to the HS21 XM. At first glance we expected greater performance from the HS21 XM blade because it offers four channels to memory instead of two. To test this hypothesis we ran a second test, Stream, and compared the results. The results showed the HS21 XM had only a modest advantage in memory performance, approximately what we saw in the FLUENT results. This was initially surprising because of our experience with DDR2 memory, but Fully-Buffered DIMMs (FBDs) represent a different memory architecture and have different performance characteristics than DDR2. FBD technology is based on point-to-point connections, whereas DDR2 is based on a shared bus. This allows FBD memory to bring more DIMMs into play more easily than DDR2 might be able to under similar conditions. Furthermore, we cannot discount the potential effect of how the operating system allocated physical memory in each of these tests. Some of these tests are small and may not have been spread across all of the DIMMs, as we might have wanted.

Our final set of results shows the performance of the HS21 XM blade with the quad-core Intel Xeon E5320 processors. The same systems and memory were used in both the Xeon 5160 and Xeon E5320 processor-based tests, with all details being identical except for the processors. The single-core Xeon E5320 results show less performance than the Xeon 5160 results, as expected, and by the expected amount. They differ by almost exactly the same proportion as the difference in their clock frequency. It is the multi-core results that are harder to gauge. In some cases the Xeon 5160 is slightly faster while in other cases the Xeon E5320 is slightly faster, even though the E5320 peak performance is 25% faster than the 5160 at these frequencies. Because there is no clear trend here, each case must be compared individually, but those cases in which the 5160 is faster are probably more sensitive to memory performance. We draw this conclusion because the 5160 has a faster front-side bus and less contention for memory than the E5320.

Notice also that we were able to obtain 16GB for the HS21 XM blade. As a result we were able to complete the FL5L3 tests. Those cases are reported for both the Xeon 5160 and Xeon E5320 processors and they show that the 5160 has a small advantage for those cases. Arguably the FL5L3 cases are the most relevant of the suite. They represent the largest data sets with more components in the model and therefore should be of more value to an engineer. So, as systems become faster the larger data sets become more accessible and therefore more relevant.

4. Floating-Point Efficiency

Perhaps it is more out of intellectual curiosity than practical analysis that we next examine the floating-point efficiency of FLUENT. Doing so doesn't help us tune the results directly, but it gives us insight into how much the processor is contributing to the problem. In later tests this helps us focus on the best processor family for this application.

To compute the floating-point efficiency we simply divide the FLUENT results by the peak floating-point performance of the cores used in the system. This levels the playing field, in a sense, and removes the consideration that some configurations have better raw floating-point performance than others. What it leaves exposed is whether considerations other than floating-point performance were important to benchmark performance.

For example, consider two systems, one based on AMD Opteron processors, the other based on Intel Xeon EM64T processors with 2MB L2 caches. To simplify matters, assume they both use 2.8 GHz single-core processors, which gives them essentially the same floating-point performance. If FLUENT is sensitive to memory performance, the Opteron processor-based system will have faster results. If it is sensitive to cache size, the Xeon EM64T processor-based system will be faster. If its performance is processor-core-limited and not sensitive to memory or cache performance, then the performance will be nearly the same. This comparison is simple and intuitive, but it is often difficult to find processors that are so evenly matched as our example. On the other hand, if FLUENT is processor-core-limited, then dividing by the peak floating-point performance of the processor will show similar values even for systems with different floating-point performance.

Table 2 shows the floating-point efficiency of FLUENT for all systems. It is interesting to note that the efficiency is similar for processor families. In other words, the efficiencies of the LS21 blades are similar to each other, the efficiencies of the HS21 and HS21 XM blades are similar to each other, and each group is very different from the other groups. The HS20 is by far the lowest, suggesting that memory performance is important to this benchmark. This suggestion is further reinforced by the fact that the LS21 efficiencies are much higher than both the HS20 and HS21, even though both have much larger caches. But while this is true, it is also clear that the greater floating-point performance and larger caches of the HS21 give it much better performance than the LS21. So, while the LS21 uses its resources more efficiently, the HS21 has more of the right resources to throw at the problem.

Another interesting observation is that the Xeon 5160 efficiencies increase both as the problem sizes get larger and as more cores are applied to the problem. Since the Xeon 5160 and Xeon E5320 processors have the same floating-point and cache architectures, the only real difference between the two is the memory performance. The E5320 front-side bus (FSB) is slower than that of the 5160, and using more cores increases pressure on the memory subsystem.

All of this has implications for the future. Future processors will increase both floating-point performance and memory performance, but they will only be able to use the increased performance to the degree that it maintains a balance similar to today's processors. In fact, to some degree the Xeon 5160, and to a larger degree the Xeon E5320, are already past the ideal balance for this workload. If future processors add more floating-point performance than memory performance, some of that floating-point performance will be wasted investment. Sometimes adding processor performance is less expensive than adding memory performance and it might help other workloads, so from the perspective of a processor designer it might still be the right thing to do, but it won't benefit FLUENT.

For the remaining sections we now concentrate primarily on the HS21 using Xeon 5160 processors, and the HS21 XM using both 5160 and E5320 processors. The 5160 and E5320 processors give the greatest performance and the least is known about them, so this is the focus for the remainder of this paper.

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS20 Using 3.2 GHz Intel Xeon EM64T (SC, 2M L2) Processors and Gigabit Ethernet									
1	113.7	113.1	48.2	16.8	45.0		18.7	14.3	
2	10.8	43.3	17.0	7.6	17.6	18.6	7.6	5.9	
LS21 Using 2.0 GHz AMD Opteron (DC, 1M L2) Processors and Gigabit Ethernet									
1	483.0	485.4	299.5	106.9	218.6	48.5	26.2	22.2	
2	485.8	734.7	304.9	108.0	155.1	47.8	26.9	22.6	
4	412.4	629.7	309.9	81.4	173.9	36.9	26.9	21.0	
LS21 Using 2.2 GHz AMD Opteron (DC, 1M L2) Processors and Gigabit Ethernet									
1	473.5	471.5	293.6	105.9	216.0	47.7	25.8	21.6	
2	484.1	724.9	319.8	106.6	154.1	46.9	26.3	22.0	
4	404.0	612.5	275.4	81.0	171.5	36.9	26.4	19.1	
HS21 Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	302.7	271.7	188.2	65.9	126.6	26.7	17.0	12.1	
2	297.8	372.7	177.1	61.4	114.0	26.0	15.8	11.5	
4	242.4	259.7	129.5	46.3	88.8	18.1	13.0	8.4	
8	190.5	187.5	121.6	43.8	89.6	18.3	12.1	8.1	1.4
16	111.7	197.8	93.3	35.4	84.5	15.9	11.2	8.1	1.5
HS21 XM Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	302.7	272.2	184.5	67.8	130.8	27.6	17.6	12.7	2.4
2	208.2	362.2	188.5	62.5	119.1	27.3	16.4	12.3	2.3
4	246.9	275.0	142.3	47.9	95.6	19.7	14.0	9.6	1.7
8	202.0	202.2	134.8	44.8	96.5	20.0	13.5	9.6	1.6
16	115.4	204.1	102.3	35.9	90.9	16.8	12.1	9.3	1.7
HS21 XM Using 1.86 GHz Intel Xeon E5320 (QC, 4M shared L2) Processors and InfiniBand									
1	303.8	290.7	195.3	67.2	129.2	28.1	17.6	13.1	2.5
2	291.9	387.6	194.8	61.7	118.4	28.0	16.3	12.6	2.4
4	244.4	335.8	163.1	51.4	109.8	24.7	15.3	11.5	2.0
8	202.9	186.5	115.4	44.1	88.4	16.8	12.4	7.6	1.3
16	143.2	230.4	89.1	36.7	86.9	16.3	11.4	7.3	1.2

Table 2: FLUENT Performance Divided by Peak FLOPS

5. BIOS Settings

It is well known that BIOS settings can have significant performance impact on almost any benchmark. The BIOS controls many aspects of the low-level structure of a system, such as how physical addresses are mapped to locations in memory, data caching policies, data pre-fetch policies, and so forth. Any of these settings could have a positive or negative impact on the performance. In our case we did not know in advance what settings would be most beneficial so we performed a few experiments to expose those that were most important.

Of the many BIOS settings available we identified Processor Adjacent Sector Prefetch and Processor Hardware Prefetcher as potential candidates. We performed two experiments to determine whether data prefetch would affect the performance. In the first experiment we disabled Processor Adjacent Sector Prefetch and enabled Processor Hardware Prefetcher. In the second experiment we enabled both Processor Adjacent Sector Prefetch and Processor Hardware

Prefetcher. In both cases we used the HS21 XM blade with 3.0 GHz Xeon 5160 processors. Table 3 shows the performance improvement from enabling this setting.

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 XM Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	2%	3%	8%	2%	4%	3%	6%	6%	
2	11%	0%	5%	2%	6%	2%	6%	5%	
4	0%	1%	3%	2%	5%	2%	6%	5%	

Table 3: Improvement Using Adjacent Sector Prefetch

Based on the observed improvement from enabling Adjacent Sector Prefetch we thought it unnecessary to explicitly test whether Processor Hardware Prefetcher would improve performance. We consider it a safe conclusion that performance is better with this switch enabled rather than disabled.

6. Network Selection

For this next set of tests two networks were available for our use. Our reference network was the Gigabit Ethernet (GbE) network that is standard within the chassis. We also had access to a high-performance InfiniBand network.

We ran four experiments in order to compare the effect of network type. Each experiment was run on a single node[1], so message passing takes place entirely within system memory. This gives the best possible performance for both networks. It also limits the impact of network selection to the overhead of the MPI stack and drivers, and doesn't actually make use of the network hardware. For these tests a pair of HS21 blades and a pair of HS21 XM blades were used. All four blades used Xeon 5160 processors. Table 4 shows the differences in performance.

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	1%	-4%	4%	2%	10%	5%	8%	6%	
2	10%	7%	11%	8%	9%	5%	12%	12%	
4	16%	3%	2%	3%	9%	7%	6%	9%	
HS21 XM Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	6%	0%	2%	3%	9%	4%	7%	6%	
2	-17%	2%	30%	3%	11%	6%	5%	7%	6%
4	15%	3%	1%	2%	8%	6%	4%	9%	4%

Table 4: Relative Speed-up of InfiniBand over Gigabit Ethernet

[1] Because of time constraints and availability of equipment, we were unable to gather data on multiple blades using both Gigabit Ethernet and InfiniBand. For this reason we compare single-blade results only, although we report InfiniBand results for multiple blades elsewhere in this paper.

While there are a few minor exceptions to this rule, the InfiniBand MPI stack provided a significant boost to FLUENT performance even when only a single core was used.

7. Processor Selection

Our next set of experiments compares the performance of the dual-core Xeon 5160 and quad-core Xeon E5320 processors. The experiments were performed on an HS21 XM blade. The configurations were identical in every way with the exception of the processors. Here we compare only those cases in which the full resources of each processor are brought to bear on the problem. Table 5 reports those results.

Blades	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 XM Using InfiniBand									
1	2%	-16%	1%	14%	15%	5%	10%	-2%	-5%
2	-12%	41%	-18%	2%	12%	1%	5%	-6%	-7%

Table 5: Relative Speed-up of Quad-Core Xeon E5320 over Dual-Core Xeon 5160

As we mentioned in a previous section, it would be misleading to provide a core-by-core comparison because the Xeon E5320 processor has twice as many cores as the Xeon 5160 processor. For this reason we present results by the number of blades rather than the number of cores that are used in the tests.

It is interesting to note that there are no clear trends here. At times the E5320 processor-based blade is much faster and at other times the 5160 processor-based blade is faster. The differences range from 41% in favor of the E5320 to 18% in favor of the 5160. Differences in the systems suggest that those cases in which the 5160 processor performs better are those cases that require more performance from the memory subsystem, but the data is not conclusive on that point. If nothing else, this clearly illustrates how different data sets can cause dramatic differences in benchmark performance, not just in the scale of the results, but also in which architectures perform best.

8. Processor Scaling

In this section we examine how performance changes as we increase the number of cores used in the benchmark. We divide the results for 2^n cores by the results for 2^{n-1} cores and report the excess over 1 as a percentage. The results are shown in Table 6.

This table shows how well the benchmark increases its performance each time the number of cores is doubled. We would expect that doubling the number of cores would result in doubling the benchmark performance. This would be reflected as 100% scaling in the table. Interestingly enough, there are cases that report much greater than 100% processor scaling. This can happen when a case with fewer cores is significantly underperforming. There are also cases where having more cores actually reduces performance, shown by negative processor scaling in the table. This can happen when system resources are oversubscribed and contention occurs, for example, in the memory subsystem. The HS20 is a clear example where performance decreases when more cores are added.

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS20 Using 3.2 GHz Intel Xeon EM64T (SC, 2M L2) Processors and Gigabit Ethernet									
2/1	-81%	-23%	-30%	-10%	-22%		-18%	-18%	
LS21 Using 2.0 GHz AMD Opteron (DC, 1M L2) Processors and Gigabit Ethernet									
2/1	101%	203%	104%	102%	42%	97%	105%	103%	
4/2	70%	71%	103%	51%	124%	55%	100%	86%	
LS21 Using 2.2 GHz AMD Opteron (DC, 1M L2) Processors and Gigabit Ethernet									
2/1	104%	207%	118%	101%	43%	97%	104%	103%	
4/2	67%	69%	72%	52%	123%	57%	101%	74%	
HS21 Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
2/1	97%	174%	88%	86%	80%	94%	86%	89%	
4/2	63%	39%	46%	51%	56%	39%	65%	47%	
8/4	57%	44%	88%	89%	102%	102%	86%	92%	
16/8	17%	111%	53%	62%	89%	74%	84%	101%	113%
HS21 XM Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
2/1	38%	166%	104%	84%	82%	98%	86%	93%	88%
4/2	137%	52%	51%	53%	61%	45%	70%	55%	45%
8/4	64%	47%	90%	87%	102%	103%	92%	101%	98%
16/8	14%	102%	52%	60%	88%	68%	80%	94%	107%
HS21 XM Using 1.86 GHz Intel Clovertown (QC, 4M shared L2) Processors and InfiniBand									
2/1	92%	167%	99%	84%	83%	99%	86%	93%	90%
4/2	67%	73%	67%	67%	85%	77%	87%	81%	68%
8/4	66%	11%	42%	72%	61%	36%	62%	32%	28%
16/8	41%	147%	54%	67%	97%	94%	84%	94%	94%

Table 6: Processor Scaling As Cores Are Doubled

Only single blades were used for the HS20 and LS21 tests so our ability to observe the scaling behavior for these systems was limited. Even so there are interesting observations to be made. For example, as mentioned before, the HS20 shows negative processor scaling for all cases reported.

There are also several cases where the one-core to two-core scaling is much higher than 100%, particularly for the FL5S2 case. For some reason this particular case shows very poor performance on the single-core experiments. At 16 cores this case also seems to get an additional boost that is independent of whether the test increases the number of blades being used. (If it were the fact that moving from 8 cores to 16 increases the number of blades in use that drives this boost, then we would see a similar boost on the Xeon 5160 blades when comparing 4 and 8 cores.)

Another common way to show processor scaling is to compare parallel performance against single-node performance, and these results are shown in Table 7. This table presents what is essentially the same information as Table 6, but cumulative effects are more easily seen in this form. For example, using Xeon E5320 processors, the scaling is only a little less than 2x for

FL5L3 comparing one node (8 cores) to two nodes (16 cores), a fact that is not readily apparent from the previous table.

What stands out in Table 7 is the fact that the costs of parallelizing FLUENT are fairly substantial for small and medium data sets, whether those costs come from partitioning and distributing the work, synchronizing results, message passing or other parallel overhead. Past experience with FLUENT running on scalable architectures shows that it can, in fact, achieve near linear speed-up if the architecture can provide adequate support and the problem size is large enough. We see that here with the L1, L2 and L3 data sets, with speed-ups of near 2x for two nodes, and near 4x for four nodes.

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
8/4	1.57	1.44	1.88	1.89	2.02	2.02	1.86	1.92	
16/4	1.84	3.05	2.88	3.06	3.81	3.52	3.43	3.85	
HS21 XM Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
8/4	1.64	1.47	1.90	1.87	2.02	2.03	1.92	2.01	1.98
16/4	1.87	2.97	2.88	3.00	3.80	3.40	3.47	3.91	4.11
HS21 XM Using 1.86 GHz Intel Xeon E5320 (QC, 4M shared L2) Processors and InfiniBand									
16/8	1.41	2.47	1.54	1.67	1.97	1.94	1.84	1.94	1.94

Table 7: Speed-Up Relative to One Node

9. Task Distribution

The next question we asked was whether the way in which tasks are distributed has an effect on FLUENT performance. The theory behind this question is that network performance seems to be a significant factor here, and interleaving MPI tasks among multiple blades can affect how much communication takes place. Furthermore, when there are fewer MPI tasks than there are cores in the system, the memory subsystem potentially has a lighter load. So if memory performance is a factor, then four MPI tasks distributed across four blades might perform better than four MPI tasks placed on a single blade. The network and memory effects are in opposition to each other; that is, distributing tasks increases memory performance but increases network traffic, so whether interleaving tasks helps or hurts performance depends on the balance of the two effects. If memory performance is more important to performance, interleaving will help. If interleaving tasks increases network traffic too much, doing so will hurt performance.

In Table 8 we see that interleaving tasks helps most cases as long as it makes more memory resources available to the benchmark. It is surprising that interleaving does not have a neutral effect on performance when the number of MPI tasks equals the number of processor cores. In the smaller tests the impact is, in fact, very high.

To some it may seem as if interleaving tasks in this manner is a “cheat,” that it benchmarks the system in ways it would probably not be used in a production cluster. On the other hand, a person designing a production cluster may want to know whether purchasing that second processor for each of a large number of blades is really going to be worth the cost.

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
1	0%	0%	0%	0%	0%	0%	0%	0%	
2	-2%	3%	1%	0%	3%	5%	-2%	4%	
4	10%	25%	34%	9%	24%	46%	16%	46%	
8	6%	25%	26%	6%	20%	15%	17%	42%	41%
16	-14%	-11%	-2%	-1%	1%	-2%	1%	2%	0%
HS21 XM Using 1.86 GHz Intel Xeon E5320 (QC, 4M shared L2) Processors and InfiniBand									
1	0%	0%	2%	0%	1%	1%	0%	-1%	1%
2	1%	7%	1%	0%	2%	5%	2%	5%	6%
4	9%	0%	16%	-1%	4%	11%	2%	12%	15%
8	0%	28%	34%	5%	22%	49%	20%	49%	52%
16	-22%	-16%	-2%	-2%	1%	-2%	0%	1%	1%

Table 8: Performance Improvement of Interleaving MPI Tasks

This question is addressed in Table 9. In each of the cases one can see the performance improvement from populating one socket per node in twice as many blades compared to two sockets per blade. To perform this experiment we used four HS21 blades with Xeon 5160 processors and ran FLUENT with two MPI tasks on each blade. This was labeled the “4x1” case (four blades with one processor each) and it approximates the performance of a blade populated with a single processor.[1] We then compared that against two HS21 blades running four MPI tasks on each (the “2x2” case). We also performed a similar experiment with two HS21 XM blades, populating them with Xeon E5320 processors. One can see that two single-socket servers are, in fact, faster than a two-socket server, a useful result for those planning future clusters.

Nodes	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 Using 3.0 GHz Intel Xeon 5160 (DC, 4M shared L2) Processors and InfiniBand									
4x1 vs 2x2	6%	25%	26%	6%	20%	15%	17%	42%	41%
HS21 XM Using 1.86 GHz Intel Xeon E5320 (QC, 4M shared L2) Processors and InfiniBand									
2x1 vs 1x2	0%	28%	34%	5%	22%	49%	20%	49%	52%

Table 9: Comparison of One-Socket and Two-Socket Servers

10. Memory Configuration

Our final set of experiments attempted to determine whether the memory configuration has a major impact on FLUENT performance. For this we used an HS21 XM blade populated with quad-core Xeon E5320 processors. This blade has four channels to memory and eight DIMM slots. DIMMs 1 and 2 are on the same channel, as are 3 and 4, 5 and 6, and DIMMs 7 and 8. We

[1] We say this approximates the performance rather than duplicating it exactly because the 5160 processor has a shared cache. If the operating system places both tasks on cores that reside in the same socket, the performance *is* duplicated exactly. But if the operating system places one task on each socket, each task would have more cache at its disposal than it would normally have on a single-socket server.

used 8 GB in both cases but we varied the placement of DIMMs to make use of either two or four channels. In order to use two channels of memory we placed DIMMs in slots 1, 2, 3 and 4. To use four channels we placed DIMMs in slots 2, 4, 5 and 7. A third configuration populated all eight slots with DIMMs. Table 10 shows the results of those tests.

HS21 XM using 1.86 GHz Intel Xeon E5320 (QC, 4M shared L2) Processors and InfiniBand									
Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 XM Populating DIMM Slots 1, 2, 3, 4									
1	2323.8	2179.1	1527.9	515.7	1018.3	218.6	136.7	101.5	
2	4401.4	5636.0	3028.9	948.9	1753.4	428.4	245.2	190.7	
4	7272.7	9757.2	4780.1	1525.2	3288.3	732.8	449.9	329.3	
8	11636.4	10909.1	6545.5	2618.2	5135.2	983.8	706.7	420.8	
HS21 XM Populating DIMM Slots 2, 4, 5, 7									
1	2252.9	2179.1	1511.1	501.5	979.6	212.4	131.3	97.7	
2	4128.0	5977.2	2736.3	920.4	1704.1	420.2	245.1	188.3	
4	7272.7	9953.9	4895.2	1531.2	3307.2	743.2	460.4	396.3	
8	12075.5	11162.8	6830.0	2630.1	5349.8	1000.6	734.4	447.7	
HS21 XM Populating DIMM Slots 1, 2, 3, 4, 5, 6, 7, 8									
1	2260.6	2162.7	1453.3	499.9	961.3	209.1	130.6	97.3	18.5
2	4343.9	5767.7	2899.3	917.7	1762.4	417.1	242.8	188.1	35.2
4	7272.7	9994.2	4853.9	1528.5	3266.5	736.3	455.0	341.0	59.2
8	12075.5	11098.3	6870.8	2624.1	5260.3	998.6	737.5	449.8	75.9

Table 10: Performance of Three Memory Configurations

The table shows that for the most part there is a small performance improvement gained by placing DIMMs on all channels. The size of this gain can be seen in Table 11. The most important case here is the one using eight threads because it fully loads the system in a manner that would be typical in a production environment. For that case there is a small but consistent improvement across all data sets over a configuration that only populates the first two channels. There is no consistent or significant difference between populating all channels with one DIMM per channel *vs*. two DIMMs per channel.

11. Conclusions

1. The dual-core Intel Xeon 5160 and quad-core Xeon E5320 were by far the fastest of the processors we tested. There was not a consistent difference between the two in spite of the E5320's higher peak performance. However, a faster clock for the E5320 or changes to the memory controller might change that. Opteron was slower, and the Xeon EM64T was slowest.
2. Opteron was by far the most efficient of the processors, which suggests memory performance is a significant factor in FLUENT performance. However, while it was more efficient, Opteron was still significantly slower than the Xeon 5160 or E5320.
3. Both Processor Adjacent Sector Prefetch and Processor Hardware Prefetcher should be enabled in the HS21 XM BIOS. Failing to enable Processor Adjacent Sector Prefetch gives away up to 6% of the performance. We also believe Processor Hardware Prefetcher should be enabled for best performance.

Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 XM Using DIMMs 1, 2, 3, 4, 5, 6, 7, 8 vs DIMMs 1, 2, 3, 4									
1	-3%	-1%	-5%	-3%	-6%	-4%	-4%	-4%	
2	-1%	2%	-4%	-3%	1%	-3%	-1%	-1%	
4	0%	2%	2%	0%	-1%	0%	1%	4%	
8	4%	2%	5%	0%	2%	2%	4%	7%	
Cores	FL5S1	FL5S2	FL5S3	FL5M1	FL5M2	FL5M3	FL5L1	FL5L2	FL5L3
HS21 XM Using DIMMs 1, 2, 3, 4, 5, 6, 7, 8 vs DIMMs 2, 4, 5, 7									
1	0%	-1%	-4%	0%	-2%	-2%	-1%	0%	
2	5%	-4%	6%	0%	3%	-1%	-1%	0%	
4	0%	0%	-1%	0%	-1%	-1%	-1%	-14%	
8	0%	-1%	1%	0%	-2%	0%	0%	0%	

Table 11: Performance Improvement of Populating All DIMM Slots

4. Using InfiniBand increases performance by up to 15% even when only a single node is used in the computation. We expect the difference to be higher when message traffic must move between blades.
5. In comparing the Xeon 5160 at 3.0 GHz and the Xeon E5320 at 1.86 GHz, there was no difference in performance that was both significant and consistent. For some tests the 5160 was faster, while the E5320 was faster for others. This could change in the future as the E5320 gets a faster clock or the memory controller is better tuned to it.
6. Processor scaling is modest for the HS21 and HS21 XM, achieving up to 12.7x (79%) on some of the smaller workloads using 16 cores, and much less than even that on the larger workloads. This suggests the speed-up and efficiency may taper off rapidly as more processors are added.
7. Interleaving tasks produces generally undesirable results, though it does show that better performance (per socket) is likely to be achieved using uniprocessor servers. In other words, eight uniprocessor servers is likely to be significantly faster than four dual-processor servers.
8. Performance is improved, albeit modestly, when all four memory channels are populated on the HS21 XM blade. There is no significant advantage to having all DIMM slots populated over simply having at least one DIMM on each channel.

12. References

[1] FLUENT Benchmarks - Overview, http://www.fluent.com/software/fluent/fl5bench/flbench_6.2/intro.htm.
[2] IBM BladeCenter HS21, http://www-03.ibm.com/systems/bladecenter/hs21/specs.html.
[3] IBM BladeCenter HS20, http://www-03.ibm.com/systems/bladecenter/hs20/specs.html.
[4] AMD Opteron LS21 for IBM BladeCenter, http://www-03.ibm.com/systems/opteron/ls21/specs.html.
[5] Marc Snir, Steve Otto, Steven Huss-Lederman, David Walker, and Jack Dongarra, MPI—The Complete Reference, Vol. 1, 2nd Ed., MIT Press, 1996.

IBM Systems and Technology Group

Department MX5A

Research Triangle Park NC 27709

Produced in the USA.

02-07

IBM reserves the right to change specifications or other product information without notice. References in this publication to IBM products or services do not imply that IBM intends to make them available in all countries in which IBM operates. IBM PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This publication may contain links to third party sites that are not under the control of or maintained by IBM. Access to any such third party site is at the user's own risk and IBM is not responsible for the accuracy or reliability of any information, data, opinions, advice or statements made on these sites. IBM provides these links merely as a convenience and the inclusion of such links does not imply an endorsement.

All performance information was determined in a controlled environment. Actual results may vary. Performance information is provided "AS IS" and no warranties or guarantees are expressed or implied by IBM. Buyers should consult other sources of information, including system benchmarks, to evaluate the performance of a system they are considering buying.